Mechatronic Systems

El-Kébir Boukas and Fouad M. AL-Sunni

Mechatronic Systems

Analysis, Design and Implementation

Springer

Authors

Prof. El-Kébir Boukas
Mechanical Engineering Department
Ecole Polytechnique de Montreal
P.O. Box 6079, Station "centre-ville"
Montreal, Quebec, H3C 3A7
Canada
Email: el-kebir.boukas@polymtl.ca

Prof. Fouad M. AL-Sunni
Department of Systems Engineering
King Fahd University of Petroleum
and Minerals
Dhahran, 31261
Saudi Arabia
E-mail: alsunni@kfupm.edu.sa

Preface

Nowadays most of the systems are computer controlled among them we quote mechatronic systems where the intelligence is implemented in microcontrollers. The discipline that deals with such systems is mechatronics that we define as the synergistic combination of mechanical engineering, electronic engineering, and software engineering. The purpose of this interdisciplinary engineering field is to control complex systems by providing hardware and software solutions. The engineers working in this field must master concepts in electronics, control and programming. Examples of such systems can be found in different industrial areas ranging from aerospace to automobile industries.

In the mechanical part, the engineer must follow a rigorous procedure to design the mechatronic system. He must build the mechanical part of the system and choose the appropriate sensors and actuators that have to be used in the functioning of the mechatronic system. At this phase we must think about the place where the electronic circuit will be integrated.

In the electronics part, the engineer must design the electronic circuit around microcontrollers that will assure the functioning of the mechatronics systems. It covers the integration of the required electronics components such as resistors, capacitors, integrated circuits, sensors and the chosen microcontrollers. The required regulated voltage for the different components is also part of this step.

In the control part, the engineer must analyze the system under study and design the appropriate controller to get the desired performances. In the analysis part, we should start by establishing an acceptable model that gives the relationship between the inputs and the outputs. Once the dynamics is mastered a sampling period is chosen and the model is converted to a discrete-time form and an appropriate controller can be chosen among the classical proportional integral and derivative (PID)

controller or the state feedback controller or any other controller that can give the desired performances.

In the programming part, the engineer must develop the code of the appropriate algorithms and then upload it in the memory of the chosen microcontroller. Many languages can be used for this purpose. In the rest of this volume, the C language is used to implement the developed algorithms.

The field of mechatronics is blooming and due to its interdisciplinarity many universities around the world have introduced complete programs on mechatronics in their curriculum. Also the number of students that are attracted by this field is also blooming and many research directions related to this have emerged recently. Huge efforts have been done to structure research in this discipline and we have seen recently many international conferences totally dedicated to this. Also some journals have been created to report interesting results on the subject. Unfortunately the number of book dealing with such discipline is limited and sometimes inappropriate for some courses in the different programs around the world.

This book provides some tools that engineers working on the mechatronics discipline can use. It can be considered as a reference for a second course in mechatronics curriculum where the students are supposed to have a prerequisite course in which the structure and the different components on mechatronics systems have been presented. It focuses only on the analysis, design and implementation of continuous-time systems controlled by microcontrollers using advanced algorithms to get the desired performances.

The hardware design of the mechatronic systems represents the hearth of the mechatronics field. It consists of designing the different parts of the mechatronic systems. Mainly beside the electronic circuit, we should select the appropriate sensors and actuators that we can use for our mechatronic system. The choice of the microcontroller is also important for the success of the desired system.

In the modeling part a model to describe the behavior of the system is developed either using the transfer function or the state space representation. In the transfer function approach part, the model of the continuous-time systems is converted to a discrete-time system and different techniques for analysis and synthesis of controllers to guarantee some desired performances are developed. In the state space approach part, the model of the continuous-time systems is converted to a discrete-time state space representation and different techniques for analysis and synthesis of controllers to assure some desired performances are developed.

The part on implementation will focus on how we can implement the control algorithm we developed either using the transfer function tools or the ones based on state space. Both the hardware and software parts will be covered to give an idea for the reader on how to deal with such problems. Mainly the selection of the sensors and the actuators that may be used in the mechatronic system will be covered.

In the advance control part, a flavor of how to design controllers that handle uncertainties and external disturbances in the dynamics is presented. This will give an idea to the reader on robust control technique and get familiar with implementation of these techniques. Stability and stabilization problems and their robustness are covered. Different controllers (state feedback, static output feedback and dynamic

output feedback) are used and linear matrix inequality (LMI) condition is developed to design such controllers.

In the case studies part, a certain number of practical examples are presented to show how the concepts we presented earlier are implemented to obtain a functional mechatronics systems. More detail is given to help the reader to design his own mechatronic system in the future.

The rest of this book is organized in seven parts and divided in eleven chapters and one appendix. In the introduction, a general overview of the mechatronics fields is given and the main concepts are recalled to make the book self-contained. In Chapter 2, the structure of mechatronic systems are detailed and some examples are given. Chapter 3 which is a part of the modeling part, deals with the modeling problem of the class of linear continuous-time systems. Both the physical laws and identification approaches are covered. The concepts of transfer function and state space representations are presented. Chapter 4 treats the $\mathscr{Z}$-transform and its properties and how the transfer function is obtained from a model that is given in a set of differential equations. Other techniques for analysis of such systems are also covered. In Chapter 5, some design approaches based on transfer function are developed. Chapter 6 deals with the state space approach for analyzing linear discrete-time systems. The concepts of stability, controllability and observability are covered. In Chapter 7, the state feedback, static output and dynamic output stabilization techniques are tackled. Chapter 8 deals with the implementation problem of the control algorithm we may develop for controlling a given continuous-time system. The focus will be made on all the steps. Mainly the hardware and software parts are covered in detail to help the reader to develop his own expertise. Chapter 9 presents some ideas on robust control. Stability and stabilization problems for systems with uncertainties and external disturbances are tackled. Chapter 10 covers the guaranteed cost control problem. Different types of controllers are used for this purpose. In Chapter 11 some selected systems are considered and all the concepts we developed in this book are applied to give the whole picture for the reader. An appendix that contains some relevant tools is also provided to try to make the book self-contained.

El-Kébir Boukas
Fouad M. AL-Sunni

In Memory of Prof. El-Kébir Boukas

Missing a very dear friend

Born in Morocco in 1954, Prof. Boukas obtained his BS Electrical Engineering degree from Ecole Mohammadia des Ingenieurs with excellent standing and with an early focus on control and application on large scale systems. Since then, he was fascinated by the area of control and its application. To fulfil his design of knowing more about it, he moved to Canada to pursue his higher studies. A decision which proved rewarding, he finished his MS and PhD in Electrical Engineering from Ecole Polytechnique of Montreal, and established himself as an authority in his area of specialization of control and automation with specialization in the use of control tools in manufacturing , maintenance and inventory control.

In his mid- fifties, he left us while still active in his research and very productive. In fact, the manuscript of this book was with him while in hospital during the last few weeks of his life. He left behind an excellent profile of accomplishments in the form of 167 High caliper International Journals, more than 8 books and many educational software and materials, and very visible presence in international conferences with more than 125 papers and presentations in conferences and involvements in organizations, and international technical committee of several of conferences over the years.

After fighting for his life, he passed away peacefully and he left behind his loyal wife , two daughters (A dentist, and an MD) and one son (soon to-be physical therapist).

I have known him since 1996, and since his visit to us in King Fahd University of Petroleum and Minerals, I have known him to be a kind, nice, helpful, and dear friend to all. He has been one of my best friends that I will always remember. He left me with the job of completing this manuscripts and then to translate it to Arabic to be the first textbook on the subject. The English version is now out, and the Arabic version is being scheduled at a later time.

Fouad M. AL-Sunni

Contents

List of Figures

List of Tables

1
Introduction

After reading this chapter the reader will:

1. have an idea on how we design mechatronic systems

2. know what are the phases of the design of such systems

3. have a clear idea on how to deal with each phase of the design of the mechatronic systems

The progress and the miniaturization we have seen in electronics during the last decades have allowed engineers to come up with new products and new engineering disciplines. Early in the eighteens we have seen the introduction of new products that combines mechanical parts with electronics parts. Another factor that gives a booming to mechatronics applications is the continuously decreasing prices of the electronic parts and the challenges to design very small systems. Today, for instance microprocessors with high performances are becoming very cheap which encourages their uses in computer controlled systems.

A microprocessor is an integrated circuit that contains the entire central processing unit of a computer on a single chip. The microprocessor is the main part in our nowadays computers. It does all the necessary computations and treats the data. The microprocessors have the following components:

- control unit

- arithmetic and logic unit

- input/output (I/O) data bus

- address bus

- internal registers

- clock

- etc.

To construct the computers, other peripherals and components are added to the main part which is the microprocessor. Screens, hard disk, floppies, memory, etc. are examples of such peripherals that we can have in our computers. For the computer controlled systems, we need appropriate cards known as data acquisition cards. These devices come with analog to digital (ADC) and digital to analog (DAC) converters and other necessary components real-time control applications. For some mechatronic systems, the use of computers and data acquisition cards are not appropriate and more often we use instead electronic circuit built around microcontrollers that can be considered as small microprocessor with their own peripherals.

A microcontroller is an integrated circuit as it is the case of the microprocessor and consisting of:

- a relatively simple central processing unit (CPU)

- memory

- a crystal oscillator

- timers,

- watchdog,

- serial and analog I/O

- pulse-width modulation (PWM) modules

- etc.

Microcontrollers are designed for small applications, while the microprocessors are used in high performance applications and personal computers. The Intel microprocessors that run in our laptops are examples of these microprocessors and the PICs of Microchip[1] are examples of microcontrollers. These machines are used in almost all the products that we use in our daily life. As examples that use microcontrollers, we quote:

- cars

- airplanes

---

[1] Microchip is a trademark, see www.microchip.com

- cellular phones

- digital cameras

- etc.

Nowadays most of the systems are computer controlled where the intelligence of these mechatronic systems is implemented in microcontrollers. The discipline that deals with such systems is mechatronics that we define as the synergistic combination of mechanical engineering, electronic engineering, and software engineering. The purpose of this interdisciplinary engineering field is to build and control complex systems by providing hardware and software solutions. The engineers working in this field must master concepts in electronics, control and programming. Examples of such systems can be found in industrial areas ranging from aerospace industry to car industry.

The design of mechatronic systems is a task that requires engineers from different disciplines like mechanical engineering, electrical engineering, control engineering, computer engineering, etc. The knowledge of these engineers are combined to produce the best mechatronic system. Most of these mechatronic systems are composed of:

- a mechanical part including the actuators and sensrors

- an electronic circuit that is built around a microcontroller or a set of microcontrollers

- a real-time implementation that represents the intelligence of the system

As example of mechatronic system, let us consider a laboratory setup for real-time implementation of control algorithms. This setup must have all the functionalities that allow learning real-time control. More specifically,

- the mechanical part must allow the user to check the output of the control algorithm

- an electronic circuit must be simple and easy to reproduce by the user in case

- the implementation must be easy to do and well documented.

In the rest of this chapter we will describe briefly each phase of the design of the whole mechatronic systems.

1.1   Mechanical Part Design

The mechanical part is a principle part in the mechatronic system. In the phase design of this part, we will conceive and manufacture the parts that compose the mechatronic system. We will also choose the actuators and the sensors we will use for this mechatronic system. Either the design of the mechanical part or the choice of the actuators and sensors are done by respecting some design rules that will be presented in a forthcoming chapter of the volume. It is also important to keep in mind

that the recycling of the mechatronic system once it becomes obsolete to respect our environment is an important matter that we must consider during the design phase. The assembly and disassembly of the system either for maintenance or any other purpose should be considered also during the design phase.

For the design of the mechanical part, the steps of the mechanical design such as definition of the problem, research of solution using brainstorming or any equivalent approach, practicability study, prototyping, etc. are used. The choice of the actuators and the sensors are also done by following the guidelines and the norms that are in use. As an example, if the mechatronic system is designed to operate in mines, electrical actuators are avoided since they may cause fires, while for food industries hydraulic actuators are excluded also.

For the setup of the real-time implementation that we are considering as example, the mechanical part in this case is only a small graduated disk (in degree) that will be attach solidly to the shaft of the actuators. This mechanical part is made from aluminium. The actuator is a dc motor that is equipped with a gearbox and an encoder. The role of the gearbox is to reduce the velocity of the mechanical part and also to apply a high torque. The encoder is used to measure the disk position and therefore, use this information for feedback. The whole is mounted on a plexiglass as it is shown in Fig. 1.1. More details on the conception of this mechanical part will be given in a forthcoming chapter of this volume.

1.2   Electronic Circuit Design

In the electronics part, the engineers must design the circuit that will assure the functioning of the mechatronics systems. It covers the integration of the required electronics parts such as resistors, capacitors, integrated circuits and the chosen microcontroller or microcontrollers. The required regulated voltages for the different components are also part of this step. The main part of the electronic circuit is the microcontroller or a set of microcontrollers. In this volume we decided to use one type of microcontroller which is the dsPIC30F4011 manufactured by Microchip. There is no real justification that we can give but only our desire is to adopt one microcontroller for all the examples we will cover in this volume. This choice will also make the real-time implementation easy for the reader since we will use the same structure for all the examples.

The regulated voltages will depend on the components we will use beside the microcontroller that requires following its datasheet a voltage between 2.5 V and 5 V. Since most of the examples use dc actuators and to drive them we need an analog signal that we can get either using a DAC or just PWM and an integrated circuit named L293D (a H-bridge). This integrated circuit needs a regulated voltage of 5 V and it will deliver a signal output that will feed the dc motor between 0V and 24V. We are also using many sensors that need regulated voltages to operate properly. Most of these devices need 5V exception made for the accelerometers and gyroscopes that requires a less regulated voltages (see the two wheels robot). For the
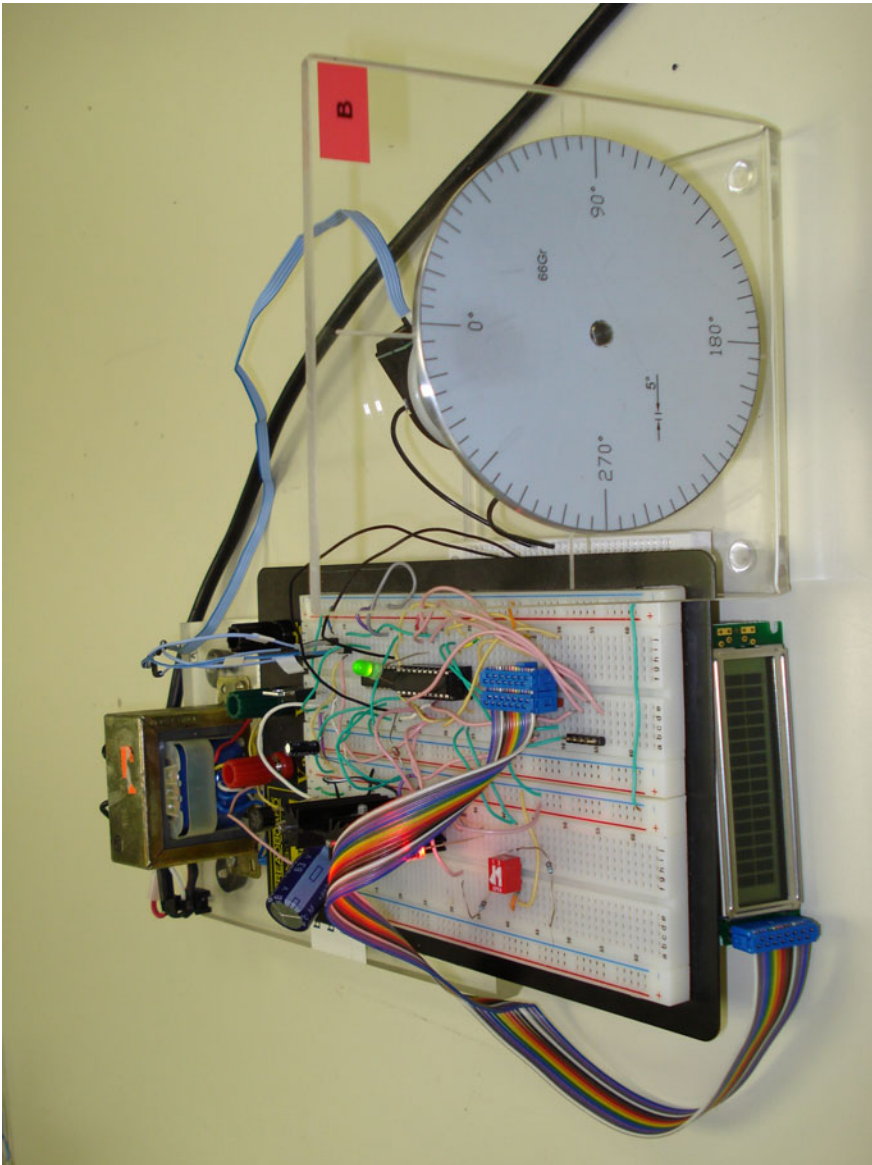
Fig. 1.1 Load driven by a dc motor kit

dc motor kit Figs. (1.1)-(1.2) give an idea of the electronic circuit of the dc motor kit that we will use in this volume.

To control the mechanical part two structures are possible. These structures are illustrated by Figs. 1.3-1.4.
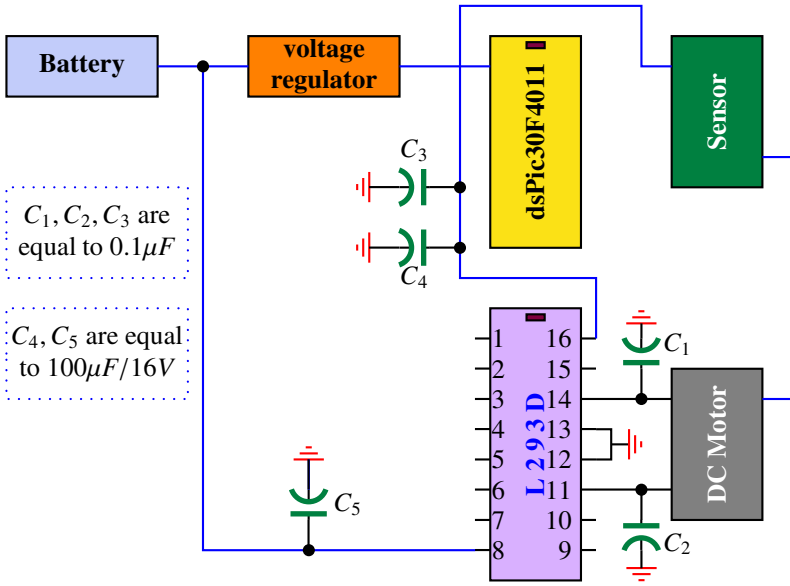
Fig. 1.2 Electronic circuit of the dc motor kit
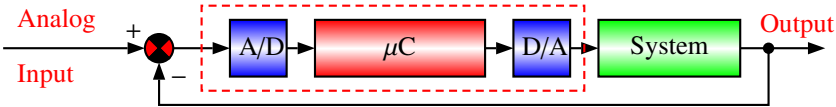
Fig. 1.3 Signal conversion made in the forward path
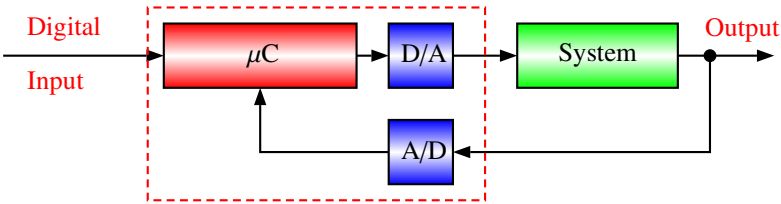
Fig. 1.4 Signal conversion made in the feedback path

If we compare these two structures, we remark that in the first one the references are analog while in the second one, they are digital. The second structure has the advantage that we can eliminate the noises. In the rest of this volume, we will adopt this structure.

The functioning of this structure is simple and it can be explained as follows. The microcontroller runs in indefinite loop and at each interrupt, the microcontroller reads the value of the output using the sensor and the ADC, then using the control

algorithm a control action is computed and sent to the system via the DAC. All these steps are done inside the interrupt routine. To avoid error calculation and error quantization the choice of number of bits either for the microcontroller or the ADC is an important issue. For the micrcontroller, a choice of 16 bits is done and this gives a good precision while for the ADC, a 10 bits will be used for all the examples we are presenting. This will not give a good precision but the results are acceptable.

If we go back to our real-time implementation setup, its electronic circuit is built around the dsPIC30F4011. The PWM module is used to deliver the voltage to the L293D integrated circuit that is in turn delivers the necessary power to drive the actuator. An encoder is used to measure the position of the small disk and also the velocity by simple calculations.

1.3   Real-Time Implementation

In the control part, the engineer must analyzes the system under study and design the appropriate controller to get the desired performances. In the analysis part, we should start by establishing an acceptable model that gives the relationship between the inputs and the outputs. Once the dynamics is mastered a sampling period is chosen and the model is converted to a discrete-time form and an appropriate controller can be chosen among the classical proportional integral and derivative (PID) controller or the state feedback controller or any other controller that can give the desired performances. To respond to the control specifications, a controller structure and its parameters are computed, then a recurrent equation is established for the determination of the control action that we must send at each sampling period to the system.

In the programming part, the engineer enters the algorithms of the chosen algorithm in the memory of the microcontroller. Many languages can be used for this purpose. In the rest of this volume, the C language is used to implement the developed algorithms.

Again if we go back to our real-time implementation setup and consider the case of two simple algorithms the PID controller and the state feedback controller. For these controllers the control action is computed using the measurement, the references, etc. In all the cases, the expression of the control law is simple and should not take a time that exceeds the sampling period (see Fig. 1.5). The implementation is done using the interrupt concept. The following example shows how the position of the load is controlled.

```
//
// A C program for the dsPic4011 for control the position of a
// dc motor driving a small disk
//

//
//   Includes and defines
//
#include <p30f4011.h>
```

```
#include <pwm.h>
#include <stdio.h>
#include <stdlib.h>
#include "xlcd.h"

#define ENABLETRIS  TRISEbits.TRISE2
#define ENABLE      LATEbits.LATE2

#define ENCODER_PRIORITY      7
#define CONTROLLER_PRIORITY   5
#define DISPLAY_PRIORITY      2

#define    Ts    0.005; // 1.0/200;
#define    Fs    200.0;

typedef struct {
     float KP;      // Proportional gain
     float KI;       // Integral gain
     float KD;     // Derivative gain
} PIDstruct;

PIDstruct thePID;

typedef struct {
     long Position;      // Shaft position
     long error[3];       // the errors
     long ref;               // the reference

     double u[2];       // control (actual and past)
}motorData;

motorData themotorData;

//
// dsPic configuration
//
_FOSC(CSW_FSCM_OFF & FRC_PLL16);;
_FWDT(WDT_OFF);
_FBORPOR(PBOR_OFF & MCLR_DIS);
_FGS(CODE_PROT_OFF);
_FICD( ICS_NONE );

//
//  Variables
//

typedef enum _BOOL { FALSE = 0, TRUE } BOOL;

BOOL    A;
```

```
BOOL     B;
BOOL prevA;
BOOL prevB;

unsigned int dutycycle;


//
//  Functions
//

// Initialization function
void Initialize(void);

// Interrupt functions
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void);
void __attribute__((__interrupt__)) _T1Interrupt(void);

//
//  Main function
//
int main(void)
{
    Initialize();

    themotorData.ref = 600;      // (90 deg)

  while(1);
}

//
//  Initialize  function
//
void Initialize(void)
{
    // variables initialization

    thePID.KA = 70.14;
    thePID.KI = -128.62;
    thePID.KD = 58.54;

    themotorData.u[0] = 0.0;
    themotorData.u[1] = 0.0;
    themotorData.error[0] = 0;
    themotorData.error[1] = 0;
    themotorData.error[2] = 0;

    // Activation of the interrupts priority
    INTCON1bits.NSTDIS = 0;
```

```
    // Digital pins
    ADPCFG = 0b11111111;

    // I/O
    TRISEbits.TRISE0 = 0;   // PWM1H
    TRISEbits.TRISE1 = 0;   // PWM1L
    TRISBbits.TRISB2 = 1;   // Encoder Chanal A : RB2 -- CN4
    TRISBbits.TRISB3 = 1;   // Encoder Chanal B : RB3 -- CN5
    ENABLETRIS = 0;

    /* start-up LCD */
    OpenXLCD(FOUR_BIT & LINES_5X7);


    //
    // initialize variables for the encoder
    //
    prevA = PORTBbits.RB2;
    prevB = PORTBbits.RB3;
//
// Initialize CN interrupts *
//

    CNEN1bits.CN0IE=0;                // CN0 interrupt disable
    CNEN1bits.CN1IE=0;                // CN1 interrupt disable
    CNEN1bits.CN2IE=0;                // CN2 interrupt ENABLE
    CNEN1bits.CN3IE=0;                // CN3 interrupt ENABLE
    CNEN1bits.CN4IE=1;                // CN4 interrupt disable
    CNEN1bits.CN5IE=1;                // CN5 interrupt disable
    CNEN1bits.CN6IE=0;                // CN6 interrupt disable
    CNEN1bits.CN7IE=0;                // CN7 interrupt disable
    CNEN2bits.CN17IE=0;               // CN17 interrupt disable
    CNEN2bits.CN18IE=0;               // CN18 interrupt disable

    IFS0bits.CNIF = 0;          // clear CN interrupt flag
    IPC3bits.CNIP = ENCODER_PRIORITY;  // CN interrupt max priority (7)
    IEC0bits.CNIE = 1;          // CN interrupt enable


//
// Configure  PWM
//
    ConfigIntMCPWM(PWM_INT_DIS & PWM_FLTA_DIS_INT);

    SetDCMCPWM(1, 1024, 0);

    OpenMCPWM (0x3FF, 0x0, PWM_EN & PWM_IDLE_CON & PWM_OP_SCALE1
                            & PWM_IPCLK_SCALE1 & PWM_MOD_FREE,
                  PWM_MOD1_COMP & PWM_PDIS3H & PWM_PDIS2H & PWM_PEN1H
```

```
                     & PWM_PDIS3L & PWM_PDIS2L & PWM_PEN1L,
                        PWM_SEVOPS1 & PWM_OSYNC_TCY & PWM_UEN);


//
// Initialize Timer 1 interrupt
//

    T1CONbits.TON=1;            // turn timer 1 on
    T1CONbits.TGATE=0;
    T1CONbits.TSIDL=0;          // stop timer in idle mode (0=non)
    T1CONbits.TCKPS=1;          // prescaler (0=1:1, 1=1:8, 2=1:64)
    T1CONbits.TCS=0;            // clock source (0=FOSC/4)
    PR1 = 18424;               // 200Hz
    IFS0bits.T1IF = 0;          // clear timer 1 interrupt flag
    IPC0bits.T1IP = CONTROLLER_PRIORITY;
    IEC0bits.T1IE=1;            // enable timer 1 interrupt


//
// Initialize Timer 2 interrupt
//

    T2CONbits.TON=1;            // turn timer 2 on
    T2CONbits.TGATE=0;
    T2CONbits.TSIDL=1;          // stop timer in idle mode (0=non)
    T2CONbits.TCKPS=2;          // prescaler (0=1:1, 1=1:8, 2=1:64)
    T2CONbits.TCS=0;            // clock source (0=FOSC/4)
    PR2 = 0xFFFF;           // slower possible
    IFS0bits.T2IF = 0;         // clear timer 2 interrupt flag
    IPC1bits.T2IP = DISPLAY_PRIORITY;
    IEC0bits.T2IE = 1;          // timer 2 interrupt enable
}

//
//     C N  Interrupt routine
//

// Decode of the position
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)
{
    if(IFS0bits.CNIF)
    {
      CNLED = !CNLED;

      // Get current Encoder signals
      // Must read port before clearing flag!!
          A = PORTBbits.RB2;
          B = PORTBbits.RB3;

      // Compare current signals with previous ones to see which
```

```
      // one has changed

      // Change occurs on A
      if(A != prevA){
        if(A == B){
                          themotorData.Position++;
        }else{
                          themotorData.Position--;
        }
      // Change occurs on B
      }else if(B != prevB){
        if(A == B){
                            themotorData.Position--;
        }else{
                           themotorData.Position++;
        }
      }

      // Save current signals for next time
      prevA = A;
      prevB = B;

      IFS0bits.CNIF=0;    // clear interrupt flag
    }

}    //end of CN_interrupt function

//
// T 1  Interrupt service routine
//

// Sampling period
void __attribute__((__interrupt__)) _T1Interrupt(void)
{
 if (IFS0bits.T1IF)
{
// Error
themotorData.error[0] = themotorData.ref - themotorData.Position;

// Control equation
themotorData.u[0] = themotorData.u[1] + thePID.KA*themotorData.error[0];
themotorData.u[0] += thePID.KI*themotorData.error[1];
themotorData.u[0] += tthePID.KD*themotorData.error[2];

// send control
SetDCMCPWM(1, 1024 + (int)(themotorData.u[0]), 0);

// save the actual data
themotorData.u[1] = themotorData.u[0];
```

```
themotorData.error[2] = themotorData.error[1];
themotorData.error[1] = themotorData.error[0];

IFS0bits.T1IF = 0;    // Clear Timer interrupt flag
  }
}

//
// T 2  I N T E R R U P T service routine
//

// LCD
void __attribute__((interrupt, auto_psv)) _T2Interrupt(void)
{
    if (IFS0bits.T2IF)
    {
        while(BusyXLCD());
        XLCDLine1();
        printf("e: %ld", themotorData.error[0]);
        while(BusyXLCD());
        XLCDLine2();
        printf("u: %8.3f ", themotorData.u[0]);

        IFS0bits.T2IF = 0;
    }
}
```
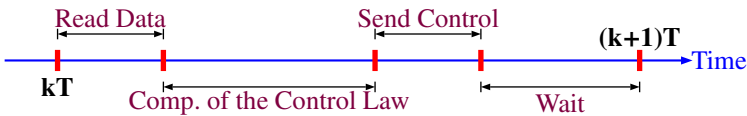

Fig. 1.5 Partition of the sampling period *T*

Example 1.3.1 *As a second example of mechatronic system, let us consider the design of a traffic light control system. We suppose that we have two streets, a main one with 80 % of the traffic while the other one has 20 % of the traffic. Fig. 1.6 illustrates the traffic light system we are dealing with and for which we should design the mechatronic system. Our goal is to design a mechatronic system that controls the traffic flow for these two streets. More specifically, we must control the lights (red, yellow and green) in each street. Most of the common traffic lights around the world consists of three lights, red, yellow and green. Fig. 1.7 gives an idea of the light used in our traffic system. In each corner of the traffic system we place a light in order that the pedestrian and the driver can see the light and take the appropriate action.*
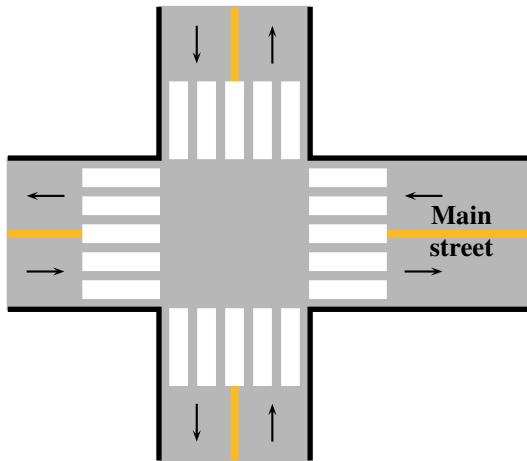
Fig. 1.6 Traffic system

When the light turns to red, the drivers must stop their car, while when it turns to green, the drivers have the right to move their car. The yellow light is used as a cautious step indicating either that the light is about to turn to green or to red and the drivers must take the appropriate actions either move or stop their cars. More often the yellow is used when the light is about to switch from green to red as an intermediate step that takes short time.

Each street is divided into two ways for two directions and each way has two lanes. The cars can either go straight or turn left or right in each way. We have also in each intersection to control the requests of the pedestrians. These requests are random and must be taken into account in a short time with a certain priority.

The mechatronic system for the traffic light is a simple system and it is composed of:

- lights that are located at each corner of the streets with some push buttons for pedestrians to request permission to cross the street

- an electronic circuit built around a dsPIC30F4011

- an algorithm in C language for control

The lights that control the traffic are placed at each corner of the street. The type of these lights is shown in Fig. 1.7. The push bottoms are also placed to help the pedestrians to cross the street when it is needed in safe way.

To simulate our traffic light we represent lights by colored light-emitting diode (LED) using the same colors as in the traffic light control system. For pedestrian we use the blue color.

The algorithm we will use for the control of the flow traffic is very simple and it is executed in a sequential manner except for the requests of pedestrians that are treated as interrupts routines. If we denote by Gmain, Ymain, Rmain, Gsec,

Pedestrian Head
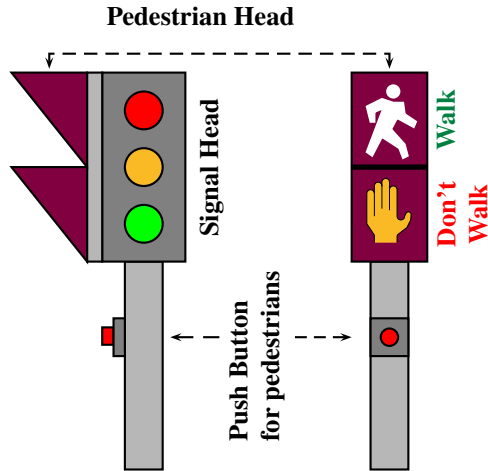
Fig. 1.7 Type of light used in the traffic light system

Ysec, Rsec the light green, yellow and red respectively for the main street and the secondary streets.The algorithm is as follows:

> *Begin loop*
>
> > – *put Gmain on, Ymain off, Rmain off, Gsec off, Ysec off and Rsec on, and wait for a time tmain*
> > – *put Gmain off, Ymain on, Rmain off, Gsec off, Ysec off and Rsec on, and wait for a time tswitch*
> > – *put Gmain off, Ymain off, Rmain on, Gsec on, Ysec off and Rsec off, and wait for a time tsec*
> > – *put Gmain off, Ymain off, Rmain on, Gsec off, Ysec on and Rsec off, and wait for a time tswitch*
>
> *End loop*

When an interrupt occurs, we identify on which corner the pedestrian pushed the button and act in consequence by stopping the traffic of the cars to allow the pedestrian to cross the street in a safe way.

The structure of the program used for the control light system is given by:

```
// Include here the headers
#include <dspic30f4011.h>

// Define variables

unsigned int i;
unsigned int Tmax = 65535;
unsigned int tmain = 8;
unsigned int tsec = 4;
unsigned int  tswitch = 1;
```

```
#define delaytmain() {for i=0;i<tmain*Tmax;i++) Nop(); }
#define delaytsec() {for i=0;i<tsec*Tmax;;i++) Nop(); }
#define delaytswitch() {for i=0;i<tswitch*Tmax;;i++) Nop(); }

#define greenMain RE0     // green light of the main street
#define yellowMain RE1    // yellow light of the main street
#define redMain RE2         // red light of the main street

#define greenSecondary RE3  // green light of the secondary street
#define yellowSecondary RE4  // yellow light of the secondary street
#define redSecondary RE5  // red light of the secondary street

typedef enum _BOOL { FALSE = 0, TRUE } BOOL;
BOOL A;
BOOL prevA;

// Initialization of the streets

// Main street
MainStreet.green = TRUE;
MainStreet.orange = FALSE;
MainStreet.rouge = FALSE;

// Secondary street
SecondaryStreet.green = FALSE;
SecondaryStreet.orange = FALSE;
SecondaryStreet.rouge = TRUE;


//  Assign the dsPic ports to the lights

//
// Functions
//

void Initialize(void);

void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)

//
// main function
//
int main (void)
{
Initialize();
while (1)
{
//            tmain
```

```
// Main Street during the tmain
greenMain = 1;
yellowMain = 0;
redMain = 0;

// Secondary street during the tmain
greenSecondary = 0;
yellowSecondary = 0;
redSecondary = 1;

delaytmin();
//              tswitch
// Main Street during the tswitch
greenMain = 0;
yellowMain = 1;
redMain = 0;

// Secondary street during the tswitch
greenSecondary = 0;
yellowSecondary = 0;
redSecondary = 1;

delaytswitch();
//               tsec
// Main Street during the tsec
greenMain = 0;
yellowMain = 0;
redMain = 1
;
// Secondary street during the tsec
greenSecondary = 1;
yellowSecondary = 0;
redSecondary = 0;

delaytsec();

//              tswitch
// Main Street during the tswitch
greenMain = 0;
yellowMain = 0;
redMain = 1;

// Secondary street during the tswitch
greenSecondary = 0;
yellowSecondary = 1;
redSecondary = 0;
delaytswitch();
}
}
```

```c
void Initialize(void)
{

TRISE = 0x00      // configure the port E  as output

//
// initialize variables for the encoder
//
prevA = PORTBbits.RB2;
//
// Initialize CN interrupts *
//

CNEN1bits.CN0IE=0;              // CN0 interrupt disable
CNEN1bits.CN1IE=0;              // CN1 interrupt disable
CNEN1bits.CN2IE=0;              // CN2 interrupt ENABLE
CNEN1bits.CN3IE=0;              // CN3 interrupt ENABLE
CNEN1bits.CN4IE=1;              // CN4 interrupt disable
CNEN1bits.CN5IE=1;              // CN5 interrupt disable
CNEN1bits.CN6IE=0;              // CN6 interrupt disable
CNEN1bits.CN7IE=0;              // CN7 interrupt disable
CNEN2bits.CN17IE=0;            // CN17 interrupt disable
CNEN2bits.CN18IE=0;            // CN18 interrupt disable

IFS0bits.CNIF = 0;             // clear CN interrupt flag

IPC3bits.CNIP = ENCODER_PRIORITY;     // CN interrupt max priority (7)

IEC0bits.CNIE = 1;          // CN interrupt enable
}

//
//     C N   Interrupt routine
//

// Pedestrian ask to cross
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)
{
if(IFS0bits.CNIF)
{
CNLED = !CNLED;

// Get the switch signal
// Must read port before clearing flag!!
A = PORTBbits.RB2;

// Compare the current signal with the previous signal to see the change
// Change occurs on A
```

```
if(A != prevA){
// put all the red lights on
}

// Save current signal for next time
prevA = A;

IFS0bits.CNIF=0;    // clear interrupt flag
}

}    //end of CN_interrupt function
```

The program starts by initializing all the variables and also configure the inputs and outputs of the dsPIC30F4011. After this, the program enters in indefinite loop in which we execute the sequence that controls the light for the intersections. If a pedestrian asks for the permission to cross the street, we shorten the time for actual activity since we can not stop abruptly the activity to prevent accidents. A given time is allocated for the pedestrian to cross the street. Once this time is finished, the sequence in the loop is resumed.

Remark 1.3.1 *For pedestrians, there is also the possibility to include right to cross the streets in the sequences that we have to execute in the program. Also late in the night, we can eliminate the rights for pedestrians since there is a small probability that a pedestrian will be at the corner and he will cross the street. But with the interrupts solution, it is possible to keep the same algorithm for all the time and we don't have to change it.*

We can improve our algorithm to make it more intelligent by adding appropriate sensors that memorize the queues in each street and act appropriately by adjusting the time of the lights in each street to reduce the waiting time of the drivers in the traffic light.

These two examples give an idea on mechatronic systems and how they can be difficult and complex to design. It is important to notice that the solution for a given mechatronic system is not unique and it varies with the knowledge of the design team. It is also important to keep in mind that the optimization should be used during the phases to obtain a competitive system.

1.4 Organization of the Book

This book can be considered as second course in mechatronics curriculum where the students are supposed to have a prerequisite course in which the structure and the different components on mechatronic systems have been presented. It focuses only on the analysis, design and implementation of continuous-time systems controlled by microcontrollers using advanced algorithms to get the desired performances.

In the modeling part a model to describe the behavior of the systems is developed either using the transfer function or the state space representation.

In the transfer function approach part, the model of the continuous-time systems is converted to a discrete-time system and different techniques for analysis and synthesize of controllers to guarantee some desired performances are developed.

In the state space approach part, the model of the continuous-time systems is converted to a discrete-time state space representation and different techniques for analysis and synthesize of controllers to guarantee some desired performances are developed.

The part on implementation will focus on how we can implement the control algorithms we developed either in the part on transfer function approach or the one based on state space. Both the hardware and software parts will be covered to give an idea on the reader on how to deal with such problems.

In the part of advance control, some algorithms that can be used to control systems with uncertainties and/or external disturbances are presented to give a flavor to the reader on the robust control theory and introduce him to the research in this field.

In the case studies part, a certain number of practical examples are presented to show how the concepts we presented earlier are implemented to obtain a functional mechatronic systems.

Part I

Mechatronic Systems

2
Mechatronic Systems

After reading this chapter the reader will:

1. master the concepts of Mechatronics and Mechatronic systems

2. be able to execute each phase in the design of mechatronic systems

3. be capable to design the mechanical part, the electronic circuit and to compute the control law and implement it in real time

4. be able to write a program in C language and how to insert it in the dsPIC30F4011

2.1   Mechatronics

Let us examine the design of an autonomous car which may be used for navigating in the floor of a building, and to move from different offices in the same floor. There are two main approaches for achieving the design of this autonomous car. The first design approach follows the classical design method. In this approach, the mechanical design is done first. After getting a satisfactory mechanical design, the electronic system is designed. In the final stage, the control system is designed.

The second design approach is to design the AC while observing the effects of each system on the overall design. In this approach, the design of the mechanical, electronic and control systems of the autonomous car are designed while taking the interaction of the design of each system and its effects on the other two systems. So, the mechanical system is finalized only after studying the effects of such design on the electronic design, and on the control system design. In this approach, the interaction between the three systems (mechanical, electronic, and control) and their effects on the final design and performance are taken into consideration at every step of the design of each system. The benefits of the second design approach are very obvious. One of the main benefits is the possibility for making the best design of each system that will make the best overall performance. This is not possible in the classical design approach because once a mechanical system is designed, it will be the final mechanical design. Also, once a mechanical design (such as the materials used, the size of the design) is decided, it will dictate and may limit the available alternatives in the electronic system (the size of the motors used, the location of the motors etc.) design which in its turn will also limit the alternatives for the instrumentation and the control system used for the overall system. The interactions between the design of the three systems (mechanical, electronics, and control) is what mechatronics offers for a better design.

Design of the mechanical system

Design of the electronic system

Design of the control and instrumentation system
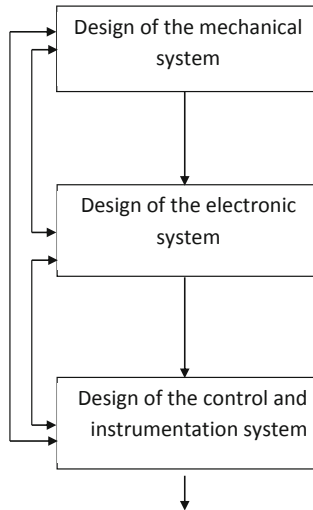
Fig. 2.1 Mechatronic design approach

Figure 2.1 presents the mehcatronic design approach in a simple way. The one-sided arrows indicates the flow of the design process, while the two-sided arrows represent the interactions between the designs of the different systems of the final product. The two-sided arrows represents the need to think about the overall design

at any point in time in the design process and at any system being designed. They mean that at any step, the design could impact the design of the other systems.

If the two-sided arrows are removed, we get the classical design approach where no interactions exist.

In the literature, there exist many definitions that have been proposed for mechatronics. These definitions depend mainly on the vision and the research interests on the field by a person or a group of persons working in some directions of mechatronics. In our vision mechatronics can be defined as an interdisciplinary field of engineering that deals with the design of products whose function relies on the integration of mechanical, electrical, and electronic components connected by a control scheme.

Nowadays, the word mechatronics is worldwide known and many mechatronic systems where designed either for personal or commercial uses. All the gadgets we are using in our daily life are in fact mechatronic systems. As an example of these gadgets, we mention our laptop and our car where many mechatronic systems are used.

A mechatronic system can be seen as a device that is able to perceive the surrounding environment and take the appropriate decisions based on the collected information. To perceive the surrounding environment sensors are needed and that without these devices the mechatronic system can not perform their tasks for which it was designed. Nowadays, cars possess many mechatronic systems to assist the driver in a safe drive among then we quote:

- airbag

- ABS brake

- speed control

- etc.

Also, to take the appropriate action, the mechatronic system needs a smart algorithm that gives actions to some appropriate actuators which can be simple switches, dc motors, stepper motors, ac motors, hydraulic motors or pneumatic motors to position for instance the mechanical part that we would like to control.

The intelligence of the mechatronic system is programmed as an efficient algorithm that coordinates all the task of the used devices. This algorithm runs in general on a powerful microcontroller.

The design of a mechatronic system is a hard task that needs interdisciplinary engineers that can understand the different elements of the system. The main components of each mechatronic system are:

- the mechanical part

- the sensors

- the actuators

- the electronic circuit

- and the program

As examples of mechatronic systems we quote

- the position control of a dc motor

- the balancing robot

- the mobile robot

- and the magnetic levitation

These systems will be used extensively in this volume to illustrate the important concepts we will cover. Before presenting these examples, let us focus on the main parts of the mechatronic systems and give some guidelines on how to design or to select them.

2.2   Mechanical Part

The mechanical part represents the main component in the mechatronic system we are trying to design. It can either be manufactured or built from existing components. In the phase design of this part more care should be paid to the following points:

- the price

- the shape

- the weight

- the size

- etc.

It is also important to pay attention to the environment in which the mechatronic system will operate. This will help us to select the appropriate material from which the mechanical part should be made. The maintenance of the mechatronic system is also a critical point, it is why we should pay attention during the design phase to the accessibility of all critical parts of the system. It is also important at this stage to think about the recycling of all the mechatronic system once it will be useless to respect the environment that we need to protect for our new generations.

The mechanical part can be made from iron, aluminium, plastic, composite or any other material. The choice of one of these materials or a combination of them will depend on many factors such the environment in which the system will operate, the weight, the task for which the mechanical part is designed for, etc.

During the design of this part care should paid also to the look of the pieces and to the fact that other parts of the mechatronic system have to be integrated later such as sensors, actuators, electronic circuit, etc. The assembly or the disassembly of all the

system should be simplified such that everybody can assemble of the disassemble the system when it is needed.

2.3   Sensors

The sensors are the key points in each mechatronic system. There are in some sense the eyes of the system through which all the type of variations are detected and the appropriate actions are taken. A sensor can be seen as a device that converts a physical phenomena like position, velocity, acceleration, temperature, flow, etc. into an electrical signal that can be easily measured or processed. A sensor is composed of a transducer and a signal conditioning unit. Nowadays, for some phenomena there exist many sensors that can be used to sense them which makes the choice very hard. Selecting a sensor for an appropriate application is always a difficult task even for experienced person in the field. The engineer must take into consideration the following points during the choice of any sensor:

- the error/precision/resolution
- the range or span
- the nonlinearity
- the repeatability
- the hysteresis
- the stability/drift
- the bandwidth
- the reliability
- the cost
- the ease of utilization

Nowadays there are a lot of type of sensors that can be categorized based on their applications or their theory of operations. Among the most used sensors in the mechatronic systems we quote:

- the encoders
- the accelerometers
- the gyroscopes
- and the cameras (image sensors)

An encoder can be defined as a device that assigns a code to represent some data. More specifically, it can be seen as a sensor or a transducer for converting rotary motion or angular position to series of electronic pulses that are appropriate for computer use. The pulses are counted and the value of the measured input is

deduced. The existing encoders in the market are either absolute or incremental. They are used in many applications among them we quote:

- the position control

- robots

- CNC machines

- medical equipment such as MRI, CT-Scan and PET-Scan machines

- etc.

The absolute encoder is mainly composed of an optical disk that has a number of tracks that gives a digital word depending of the position of the shaft. As an example, if we consider a disk with 8 tracks, in this case the encoder has 256 distinct positions, which gives an angular resolution of 1.4 degrees. The Gray and the binary codes are commonly used in the absolute encoder.

The incremental encoder is simpler compared to the absolute encoder and it consists of two tracks and two sensors that give two channels A and B. When the shaft of the sensor rotates pulse trains appear on the two channels that are quadrature signals. These signals can be used to determine the angular position and the rotation sense. A third output channel referred to as *Index* that produces a pulse by revolution and it is used to count the number of revolutions.

The accelerometer is a device that can be used to detect the acceleration and tilt. Nowadays accelerometers are used in cars for passenger security. Their role is to detect the impact and deploy the car airbag when it is necessary to save the life of the passengers. These type of sensors found use in digital cameras where their role is to guarantee the stability of the image.

Nowadays, the accelerometer comes in MEMS technology. The MEMS accelerometer usually comes in the smallest surface mount package and can detect acceleration in up to 3 axis. The data from this accelerometer can directly be used by the microcontroller and therefore take the appropriate action when it is required. The accelerometer can be used to measure the acceleration of the object or measure the tilt of the object to which the sensor is attached to.

The gyroscope can be seen as a device consisting of a rotating wheel mounted on a base so that its axis can turn freely in certain or all directions, and capable of maintaining the same absolute direction in space regardless of any movement of the base. This device is used in airplanes, satellites, robots, etc. Nowadays, the gyroscopes come in MEMS technology which facilitates their application in mechatronic systems.

The cameras (image sensors) can be seen as complementary metal oxide semiconductor (CMOS) or charge-coupled device (CCD)-based chips that record the intensities of light as variable charges. The cameras contains millions of pixels arranged in a matrix which catches and records light when a picture of an element is taken. The cameras are used extensively in image processing for quality control, supervision, etc.

2.4 Actuators

Actuators are defined as devices that convert some kind of power, such as hydraulic or electric power, into linear or rotary motion. They represent the arms of the mechatronic systems. In practice different type of actuators are used, among them we quote:

- electric actuators

- hydraulic actuators

- and pneumatic actuators

An electric actuator is a device that convert electric power into a linear or a rotary motion. They are used to position or to give the speed to the mechanical part of the mechatronic system. The common electric actuators are:

- the dc motors

- the ac motors

- the stepper motors

- and the switches

These actuators own the following advantages:

- high speeds

- self contained

- low cost

- simple design

- reliable operation (less maintenance)

- high efficiency

- long life

The dc motors beside being cheap and simple are easy to control in speed, position and torque. While their homolog ac motors are in general expensive in speed control, show some instability in operating at low speeds and own poor positioning control.

Electric actuators are in general precise and flexible. They are ideal to position mechanical part precisely or to develop forces quickly when it is required. Their major disadvantage is that they need cooling systems during their operation. When they are well designed and well protected, their maintenance is reduced to the changes of the sliding contacts or the commutators. Large load may burn the winding of the electric actuators if the protection is not installed properly.

Stepper motors are more appropriate to control mechanical parts that don't require feedback. Mostly these type of actuators are used in open loop control and to position the mechanical part. For this purpose a certain number of pulses are sent

by the microcontroller. These actuators are used in laser printers, faxes, and most of the appliances for computers.

A hydraulic actuator can be defined as a cylinder or fluid motor that converts hydraulic power provided by a pump into a useful mechanical work. The mechanical motion that results may be linear, rotary, or oscillatory. This type of actuator provides the following advantages:

- high dynamic response

- high force capability

- high power per unit weight and volume

- good mechanical stiffness

while the disadvantages are:

- leakage

- need more maintenance (filers)

- need external hydraulics pump

These features lead to wide use in precision control systems and in heavy-duty machine tool, mobile, marine, and aerospace applications.

The pneumatic actuator is defined as a device that uses pressurized air to create mechanical motion (linear or rotary). Similarly to the hydraulic actuator, this one also requires a compressor for air to operate. It is also important to mention that the efficiency of this kind of actuators is low. The pneumatic actuators are in general inexpensive and their operations are not affected by difficult environmental factors such as dust, etc. and they are easy to install and operate. They have less precision compared to the other actuators due the compressibility of the air. Pneumatic actuators are appropriate for use in potentially explosive environments. Contrary to the electric actuators, the pneumatic ones can support large loads and don't require the cooling system.

Selecting an actuator for an appropriate application is always a difficult task even for experienced engineers in the field, meanwhile main guidelines should be kept in mind. In fact the power, the environment of operation are main points to be considered and can help in choosing the type of actuators. For instance, if the mechatronic system is designed to operate in mining where sparks may cause fire, the electric actuators are excluded and the hydraulic actuators are possible solutions. In food industry, the hydraulic actuators are excluded and electric or pneumatic actuators are the possible solutions.

2.5   Electronic Circuit

The electronic circuit is the brain of the mechatronic systems. It regroups passive and active components beside integrated circuits. Its role is to manage and coordinate in a desired way the functioning of all the components that compose the system.

The passive components include resistors and capacitors, while the active ones can be a simple diode or a transistor or any integrated circuit that performs the desired task. The electronic circuit manages and orchestrates a variety of functions that the mechatronic system allows beside providing the desired regulated voltage for the different integrated circuit, the sensors, the actuators and the micrcocontroller.

When designing the electronic circuit we must keep in mind that the size of the circuit and its consumption in power should be minimized. The safety of the circuit and its cooling are also of importance. In case of manipulating high voltage security rules should be followed seriously.

2.6   Real-Time Implementation

Once the hardware part of the mechatronic system is built, the next step is to design the control algorithm that we should implement to guarantee that the system will perform properly the tasks for which it was designed for. The design of such algorithm is done into two steps. The first one consists of establishing the mathematical model that describes properly the relationship between the inputs and the outputs of the system. This model can be determined either analytically with some limited experiments to the values for some parameters, or experimentally using the identification techniques. In the second step, the desired performances are fixed and the controller is designed using the appropriate techniques. The results of this step is the determination of the recurrent equation that will compute the decision at each interrupt. This equation represents the algorithm that we have to implement in the microcontroller.

The microcontroller is used to provide real time response to the different events for which the system is designed for. In general is running in a loop and when an event occurs, the associated interrupt system alerts the processor to suspend processing of its current instruction and to start an interrupt service routine. This interrupt routine executes the main steps of the control algorithm that we are using. Once the task of the interrupt service routine is completed, the processor returns to the place where the execution were suspended.

The implementation is done following the following structure:

```
//
// Put here title and comments
//
#include "p30F4011.h"        // proc specific header

//
// Define gobal variables in RAM
//
float  Reference;   // simple variable
int variable0;         // (16 bits)
char myVariable; // (8 bits)
```

```
#define n1 10   /* sample constant definition */
#define n2 20;
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
                // array with dsPIC30F attributes
int array5[n2];    // simple array
int variable3 __attribute__((__space__(xmemory)));
                 // variable with attributes
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
                 // array with dsPIC30F attributes
int array5[n2];       // simple array
int variable4 __attribute__((__space__(xmemory)));
                 // variable with attributes

 //
 // Define a struct
 //
  typedef struct {
       // PID Gains
       float KP;      // Propotional gain
       float KI;       // Integral gain
       float KD;      // Derivative gain

    //
    // PID Constants
    //
    float Const1_pid;  // KP + T KI + KD/T
    float Const2_pid;  // KP + 2KD/T
    float Const3_pid;  // Kd/T
    float Const4_pid;  // KP + KD/T
    float Const5_pid;  // T KI

    //
    // System variables
    //
    float y_c;      // y_c[k] -> controlled output
    float y_m;     // y_m[k] -> measured output
    float u_k;      // u[k]   -> control at time k
    float e_k;      // e[k]   -> error at time k

    //
    // System past variables
    //
    float u_km1;     // u[k-1] -> output at time k-1
    float e_km1;     // e[k-1] -> error at time k-1
    float e_km2;     // e[k-2] -> error at time k-2
    float y_mkm1;  // y_m[k-1] -> measured output at time k-1
    float y_mkm2;  // y_m[k-2] -> measured output at time k-2

  }PIDStruct;
```

```
  PIDStruct thePID;

//
// Constants in ROM
//
const char Variable_Rom[] = {1,2,3,4};
const int myConstant = 100;

//
// Non memorized constants
//
#define var1 0x1234;
#define var2 "ma chaine";

//
// Functions
//
float my_Function(float a, float b)
    {
        int local_var;

        local_var = a - b;
        return local_var;
    }



//
// Interrupt program here using Timer 1 (overflow of counter Timer 1)
//
void __ISR _T1Interrupt(void)     // interrupt routine code
  {
      // Interrupt Service Routine code goes here
      float Position_error;

      // get the actual position from the encoder
      // ThePID.y_m

      Position_error = my_Function(Reference, ThePID.y_m);
      .......

      IFS0bits.T1IF=0;    // Disable the interrupt
  }


int main ( void )                 // start of main application code
{
// Application code goes here
int i;
```

```
// Initialize the variables Reference and ThePID.y_m
(it can be read from inputs) Reference = 0x8000; // Hexadecimal number
(0b...  Binary number) ThePID = 0x8000;

// Initialize the registers
TRISC=0x9fff;  // RC13 and RC14 (pins 15 and 16) are configured as
outputs IEC0bits.T1IE=1;   // Enable the interrupt on Timer 1

// Infinite loop
while (1)
{
}
return 0
}
```

2.7   Examples of Mechatronic Systems

The aim of this section is to present some mechatronic systems that may be used in the rest of this volume to show the different concepts we will develop. We will try to present all the parts of these mechatronic systems to help the reader to make a clear idea on the design of mechatronic systems and hope that this will help him to design his own system in the future.

We will restrict ourself to mechatronic systems that use common components like electric actuators, encoders, accelerometers, gyroscopes, etc.

2.7.1   Dc Motor Control

As a basic mechatronic system, let us design a setup that can be used either for speed or position control. This system will be the basis of almost all the coming mechatronic systems. The system we will present here consists of a dc motor that drives via a gear a small disk. In order to control it properly either in speed or in position an incremental encoder is used.

The mechanical part of this system is a small disk that is manufactured in our mechatronics laboratory. Graduations are indicated on the disk to help us to position it at any desired position we want. The disk is made from aluminium and attached solidly to the motor shaft using a screw.

The actuator is a small dc motor that we bought from a surplus store. It has already a gear (ratio is 1 : 6) and an incremental encoder (100 pulses/rev). The electronic circuit of this system is too simple and it can be summarized to:

- a transformer

- two voltage regulators (T78012 and T7805)

- resistors (2 resistors of 10 KΩ, 2 resistor of 220 Ω and a variable resistor of 20 KΩ and capacitors (3 of 0.1 μF)

- diodes

- an H bridge

- a liquid crystal display (LCD)

- switch (to put the system on or off)

- a microcontroller

This setup is designed to operate in a fixed place. Therefore, we do not need to use batteries to deliver the necessary power to the different components. The necessary regulated voltages are obtained from the ac current. Firstly, the ac voltage (115 V) is changed to a lower level 36V using a transformer in our case. A Graetz bridge is combined with a low pass filter to rectify the voltage and smooth it for proper use in the components.

To drive the dc motor, a 24 V is needed and therefore an integrated circuit (IC) motor driver named L293D (dual H-bridge) is used. The presence of the letter "D", means that it is built in flyback diodes to minimize inductive voltage spikes. The L293D chip gives the desired power to the dc motor to move the load to the desired position if it is the case. This IC has an output current of 600 *mA* and a peak output current of 1.2 *A* per channel. It is important to notice this limitation since if the motor requires more current, the IC L293D will burn each time we exceed 1.2 A and a protection such as a fuse is needed in this case.

For the speed or the position control, we use the Microchip dsPIC30F4011. The intelligence that we will implement in the system is programmed in C language and after compilation, it is downloaded in the memory of the microcontroller.

Fig. 2.2 gives an idea of the whole mechatronic systems. The dc motor we use in this setup is manufactured by Maxon and it has a gear of 1: 6 ratio. An incremental encoder attached to shaft of the motor is also used to measure the position of the disk. With this setup we get 600 pulses per revolution. Our incremental encoder uses two output channels (A and B) like most of the incremental encoders to sense position. Based on the two code tracks on the disk of the encoder (positioned 90 degrees out of phase), the two output channels of the quadrature encoder indicate both position and direction of rotation. Therefore, if A leads B, for example, the disk is rotating in a clockwise direction, meanwhile if B leads A, then the disk is rotating in a counterclockwise direction. Another benefit of the quadrature signal scheme is its ability to electronically multiply the counts during one encoder cycle. Mostly the following is used for this purpose:

- all counts are generated on the rising edges of channel A

- both the rising and falling edges of channel A are used to generate counts

- the rising and falling edges of channel A and the channel B are used to generate counts
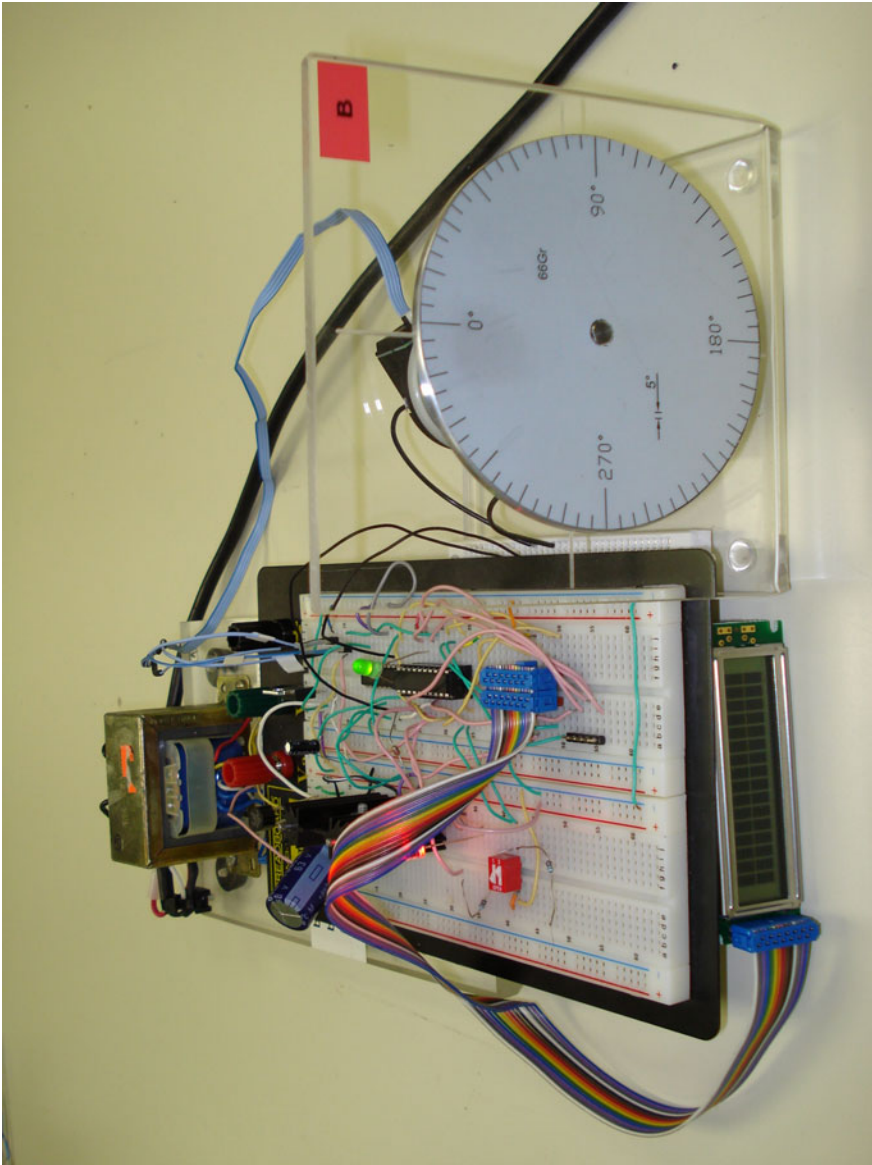
Fig. 2.2 Real-time implementation setup

Using the second or the third options we can increase the resolution and conse-
quently improve the control precision. For instance, if the third option is used the
resolution is increased by a factor of four and therefore we get 2400 pulses/rev.

For the speed control if the controller is chosen as proportional controller with
a gain K_p, the system will work as follow. Firstly a speed reference is selected let
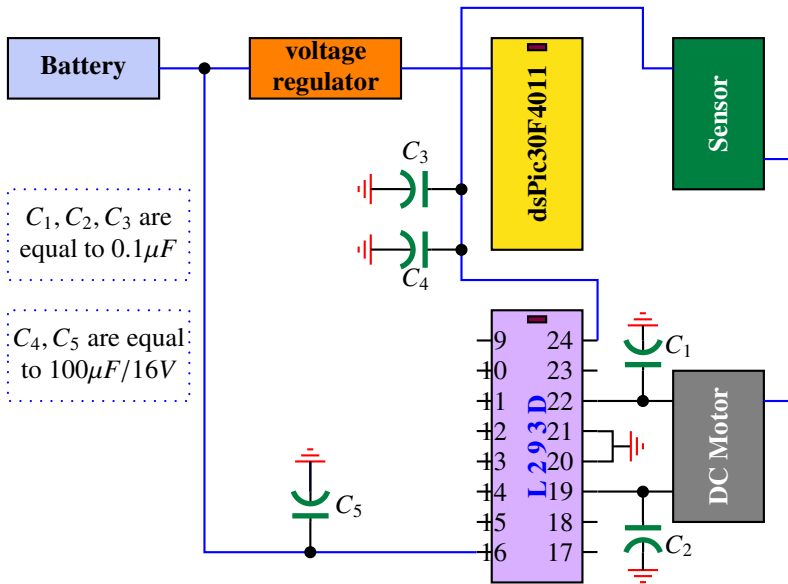
Fig. 2.3 Electronic circuit of the dc motor kit

say 100 rev/s. At each interrupt, the microcontroller will read the speed of the disk, compare it to the reference and compute the error. This error is multiplied by the gain K_p to generate the voltage to be delivered to the dc motor. Since the microcontroller can deliver a voltage between 0 and 5 V, the integrated circuit motor driver L293D will do the necessary to deliver only a voltage between 0 and 24 V with sufficient power to move the motor. The pulse width modulation (PWM) technique is used in this case. This technique is used to generate variable-width pulses to represent the amplitude of an analog input voltage that we should deliver to the dc motor. The PWM technique is characterized by its duty cycle which gives an indication of the fraction of time when the signal is on. The control of the voltage or the speed of the small disk is obtained by adjusting the duty cycle. The PWM works by making a square wave with a variable on-to-off ratio, the average on time may be varied from 0 to 100 percent. Fig. 2.3 gives an idea of the electronic circuit.

2.7.2   Two Wheels Robot

The idea of the two wheels robot has attracted a lot of researchers for the challenges it offers either in the modeling or in control. Different types of robots have been developed in research laboratories around the world. In our mechatronics laboratory, we have designed an experimental one that we use in our research to experiment our control algorithms. This robot has a compact structure and can be assembled or disassembled easily and quickly. It is composed of a platform on which a rod is attached at its middle. The whole is mounted on two wheels that are solidly attached

to the platform and are driven by two independent dc motors of the same type we used in the previous mechatronic system. The major parts of this robot are made from aluminium to reduce the robot weight. The electronic circuit which is a little bit more complicated compared to the previous system. This circuit is mounted on a breadboard and fixed to the platform. A set of batteries to obtain 24 V is used to deliver the different regulated voltages we need in this system. The batteries are put between the electronics and the platform.

The electronic circuit of this system is in some sense similar to the previous mechatronic system except for this system we need more components since we have two dc motors. The electronic circuit is built around the dsPIC30F4011 that orchestrates and manages all the tasks of the different parts of this system. For this electronic circuit we need more voltages since the LCD and the L293D need 5 V

Fig. 2.4 Balancing robot

to operate, while dsPIC30F4011, the wireless, the accelerometer and the gyroscope need 3.5 V.

Beside the two encoders that are used to measure the positions of the wheels and therefore the one of the robot, an accelerometer and a gyroscope are used to measure the tilt of the robot. The goal is to keep the robot in the vertical position while moving along a desired trajectory. All this is done by controlling the dc motors. The PWM technique is also used here to deliver the desired voltages that are generated by the control algorithm we implement in the dsPIC.

The references to the robot can be either entered by program or sent wireless using a telecommunication system. Different control algorithms are experimented on this system. Some of these algorithms will be developed in the rest of this book.

Fig. 2.4 gives an idea of the whole mechatronic system, while the Fig. 2.5 gives an idea on the electronic circuit. The program is similar to the one of the dc motor kit except that more complex and too long to be presented here.

Fig. 2.5 Electronic circuit of the balancing robot

2.7.3   Magnetic Levitation

Magnetic levitation is a technology that has a lot applications which attracted a lot of researchers to this field. As an example where this technology is extensively used is in fast magnetic levitation trains since it permits to reduce the friction and therefore eliminates energy losses. In this section we will develop a system that used this technology and show that the principle works. The mechatronic system developed here is composed of two parts: a fixed one that represents the coil that generates the electromagnetic force and a ferromagnetic object that we would like to place at a certain position by acting on the electromagnetic force generated by the coil. The objective of the system is to control the vertical position of the moving object by adjusting the current in the electromagnet through the input voltage. The object position is measured using a Hall effect sensor. An elecgronic circuit build around a dsPIC30F4011 supplies the coil through an L298, an integrate circuit, with a current that is proportional to the command voltage of the actuator. Fig. 2.6 gives an idea of the whole mechatronic system.

2.8   Conclusions

In this chapter, we have presented the different components of mechatronic systems and we gave some mechatronic systems that we will use here to show the concepts developed in this volume. Some guidelines that can be used during the design phase of mechatronic systems are developed to give an idea to the reader and help him to design his own system.

2.9   Problems

1. In this problem we would like to design a one leg robot that can move using one wheel while remaining in a vertical position. Provide the design of such mechatronic system.

2. Solar energy is an alternate source of power that can be used. In this problem we ask you to design a solar system that maximizes the energy generated by the solar panel.

3. In this problem we ask you to design an insect with four legs that can walk and at the same time avoid obstacles.

4. Make the design of a small car that may use magnetic levitation to move. Give the different parts of such system.

5. In this problem we ask you to design a small airplane that may be used as a drone to give information of a certain region when flying over such region. Enumerate the different parts that may be used in such system.
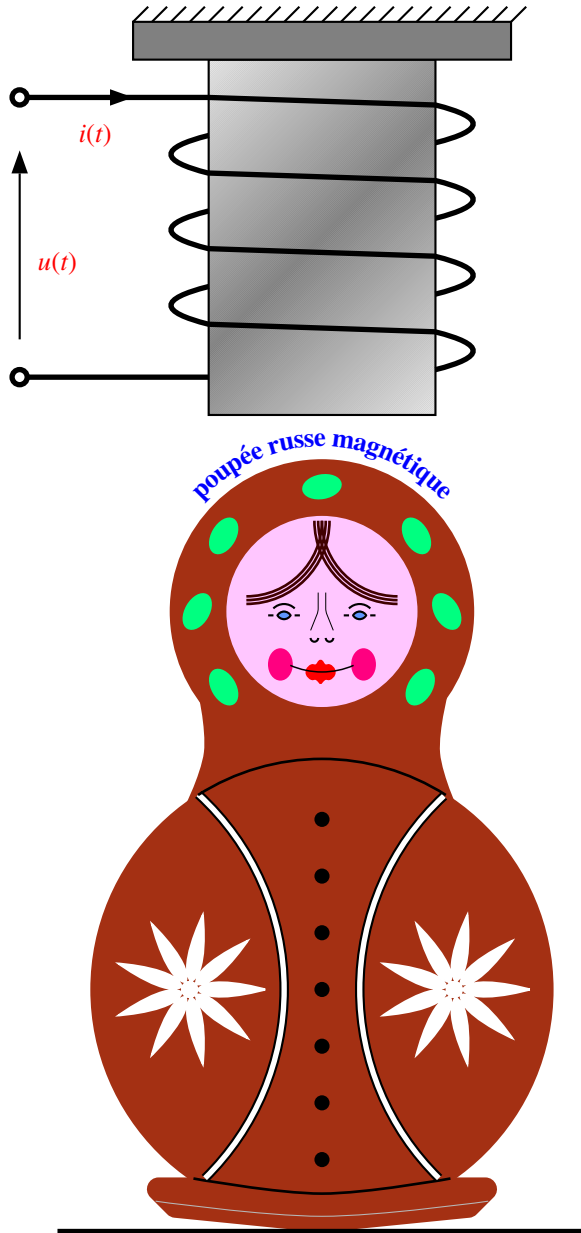
Fig. 2.6 Magnetic levitatios system

6. Design a small boat that we move on a small lake using a joystick. Enumerate the different components of this system.

7. In this problem we ask to design a hoover that can be controlled to seal on water via a emitter and a receiver using a joystick.

Part II

Modeling

In this modeling part we will cover different representations that may be used to describe a dynamical system that we would like to control in order to improve its performances. As it was said earlier, the focus is made on the control of continuous-time systems by microcontrollers that we can represent using one of the following representation:

1. transfer function

2. state space representation

More often, the relationship between the inputs and the outputs is described by differential equations that may be linear or nonlinear. For single input single output linear time invariant system, the transfer function, $G(s)$ is defined as follows:

$$G(s) = \frac{Y(s)}{R(s)} \tag{2.1}$$

where s is a complex variable that belongs to the set of complex number $\mathbb{C}$, $Y(s)$ and $R(s)$ represent respectively the Laplace transform [1] of the output, $y(t)$ and the input, $r(t)$ respectively, i.e.:

$$\begin{aligned} Y(s) &= \mathscr{L}\left[y(t)\right] \\ R(s) &= \mathscr{L}\left[r(t)\right]. \end{aligned}$$

The relation between the input and the output is then given by:

$$Y(s) = G(s)R(s) \tag{2.2}$$

For the multi-inputs multi-outputs case, we get similarly the following relation:

$$Y(s) = G(s)R(s) \tag{2.3}$$

with

$$R(s) = \begin{bmatrix} R_1(s) \\ \vdots \\ R_m(s) \end{bmatrix}$$

$$Y(s) = \begin{bmatrix} Y_1(s) \\ \vdots \\ Y_p(s) \end{bmatrix}$$

$$G(s) = \begin{bmatrix} G_{11}(s) & \cdots & G_{1m}(s) \\ \vdots & \ddots & \vdots \\ G_{p1}(s) & \cdots & G_{pm}(s) \end{bmatrix}$$

where $R_i(s)$, $Y_j(s)$ and G_{ji} represent respectively the ith input, the jth output and the transfer function between them when the other inputs are fixed to zero.

---

[1] The Laplace transform of a function $f(.)$ that satisfies the appropriate assumptions is defined by $F(s) = \int_0^\infty f(v)dv$

Notice that the jth output is given by the following expression:

$$Y_j(s) = G_{j1}(s)R_1(s) + G_{j2}(s)R_2(s) + \cdots + G_{jm}(s)R_m(s) \qquad (2.4)$$

which implies the dependence of the outputs on the different inputs.

Usually, we use also the block diagram of Fig. 2.7 to represent dynamical systems.
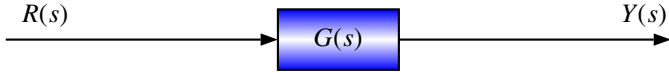
Fig. 2.7 Block diagram of continuous-time system

The state space representation is another way of representing the relationship between the input $u(t) \in \mathbb{R}^m$ and the output $y(t) \in \mathbb{R}^p$ of a given system and we can obtain it by proceeding with some mathematical transformation either of the differential equations or its corresponding transfer function. Its general structure is given by:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0 \\ y(t) = Cx(t) + Du(t) \end{cases} \qquad (2.5)$$

where A, B, C and D are constant real matrices with appropriate dimensions; and $x(t) \in \mathbb{R}^n$ and x_0 represent respectively the state of the system and its initial condition.

Usually the following block diagram (see Fig. 2.8) is used to represent dynamical systems in state space description:
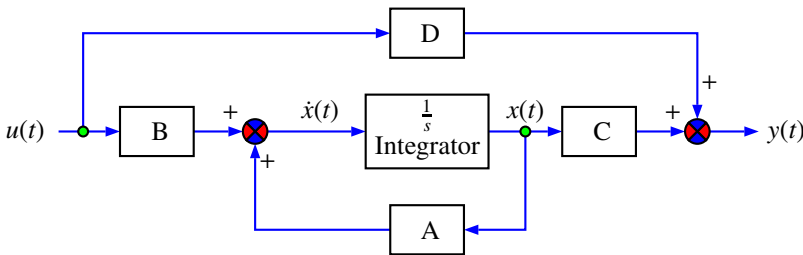
Fig. 2.8 Block diagram of continuous-time linear system

The goal of this part is to show to the reader how we can establish the mathematical model of a given dynamical system. The model can either be obtained through experiment or using the physics law with some specific experiments that may be used to determine the appropriate parameters that enter in the mathematical model obtained by this approach.

3

Mathematical Modeling

After reading this chapter the reader will:

1. be able to establish a mathematical model for any mechatronic system either analytically based on physics law or experimentally using the identification techniques

2. be able to build mathematical models for the mechatronic system using the transfer function concept

3. be able to build the state space representation for any given mechatronic system

It is well known that the mathematical modeling is a hard problem in control engineering. Most of the engineers working in this field agree on that. Any practical system has inputs and outputs. The outputs are variables that we would like to control or keep at certain levels, while, some of the inputs are variables on which we can act to change the outputs of the dynamical system. The rest of the inputs are referred to as external disturbances that are beyond our control.

A mathematical model is a representation that uses mathematical language, more often differential equations or difference equations, to describe the behavior of a dynamical system. Mathematical models are extensively used in engineering

disciplines to describe the relationship between inputs and outputs and the dynamical system parameters.

Mathematical models of dynamical system can be split into two categories depending on how the time variable is to be treated. A continuous-time mathematical model is based on a set of differential equations that are valid for any value of the time variable, whereas a discrete-time mathematical model provides information about the state of the physics system only at a selected set of distinct times.

The development of an appropriate model to describe the behavior of a given dynamical system can be done in different steps. At the first step, the inputs and the outputs variables are chosen. Then, at a second one the appropriate assumptions are made and the mathematical relationships between these variables are established using physics laws. Some experiments are required to determine the system's parameters.

In some circumstances, this approach is too complex and an another alternate is adopted to avoid this complexity. This approach consists of considering the dynamical system as a black box and recourse to the identification techniques. In the rest of this chapter we will cover these techniques and show to the reader how we can handle the mathematical modeling of some dynamical systems. In both cases we will be looking for the simplest accurate model we can get since this will facilitate the analysis and the design phases.

3.1 Mathematical Modeling Based on Physics Laws

To show how this technique can be applied let us consider a certain number of dynamical systems. As a first example let us consider a dc motor with a mechanical load that we like to control either in speed or in position. The dc motor represents the actuator that is mostly used in the position control servomechanism. It is the means by which the electrical energy is converted to mechanical energy. The block diagram of the dc motor driving the load of our example is illustrated by Fig. 3.1. If we let $u(t)$, $i(t)$ and $\omega(t)$ denote respectively the voltage of the armature, the current in the armature and the speed of the shaft at time t, based on the basic electrical and mechanics laws we have the following:

$$\begin{cases} u(t) = Ri(t) + L_m \frac{di}{dt}(t) + K_w w(t) \\ J \frac{dw}{dt}(t) = K_t i(t) - b\omega(t) \end{cases} \tag{3.1}$$

where R, L_m, K_w, K_t represent respectively the electric resistor of the armature, the inductance of the armature, the electromotive force constant, the torque constant (in the international system (IS) these both constants are equal), J and b are defined by:

$$J = J_m + \frac{J_c}{n^2}$$
$$b = b_m + \frac{b_c}{n^2}$$

with J_m and J_c are the moments of inertia of the rotor and the load respectively, and b_m and b_c are the damping ratios of the motor and the load, and n is the gear ratio.
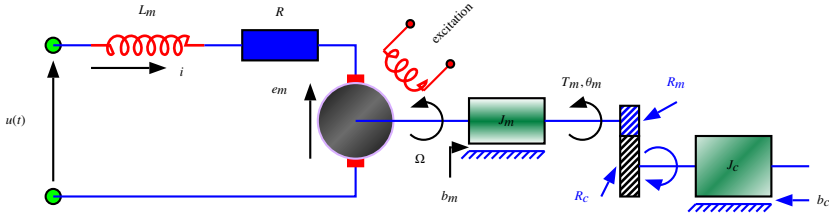
Fig. 3.1 Block diagram of a dc motor

3.1.1   Concept of Transfer Function

If we use the Laplace transform with the initial conditions equal to zero, we get:

$$\begin{cases} U(s) = RI(s) + L_m sI(s) + K_w \Omega(s) \\ Js\Omega(s) = K_t I(s) - b\Omega(s) \end{cases} \tag{3.2}$$

where $U(s)$, $I(s)$, and $\Omega(s)$ are respectively the Laplace transform of $u(t)$, $i(t)$ and $\omega(t)$.

Combining these relations and the definition of the transfer function between the velocity $\Omega(s)$ and the voltage $U(s)$, we get:

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{K_t}{(Js + b)(L_m s + R) + K_t K_w} \tag{3.3}$$

If the armature inductance L_m can be neglected, the transfer function becomes:

$$G(s) = \frac{K}{\tau s + 1} \tag{3.4}$$

with

$$K = \frac{K_t}{Rb + K_t K_w}$$

$$\tau = \frac{JR}{Rb + K_t K_w}$$

Remark 3.1.1 *When the armature inductance L_m can be neglected the mathematical model can be simplified to a first order system, otherwise we have a second order one. It may happen in some applications that the dynamics of the driven load is too slow compared to the actuator one and in this case, the dynamics of the actuator is reduced to a simple gain.*

Remark 3.1.2 *The parameters of the dc motor are in general available in its data sheet. Once the inertia of the load and the gear ratio are known, all the data of the model are then known. It is also important to mention that the data sheet contains the average data for a sample that has been tested to get these parameters. Therefore, it may happen that the considered actuator may have uncertainties in its model that can be compensated by the choice of the appropriate controller.*

Notice also that the position, $\theta(t)$, of the dc motor is obtained from the velocity, $\omega(t)$, by using:

$$\Theta(s) = \frac{\Omega(s)}{s}$$

where $\Theta(s) = \mathcal{L}[\omega(t)]$.

Using this and the simplified model between the voltage and the velocity, we get the following relationship between the voltage and position:

$$G(s) = \frac{\Theta(s)}{U(s)} = \frac{K}{s(\tau s + 1)}$$

where K and τ are defined previously.

Notice that the previous relations of the mathematical model between the voltage and the velocity can be rewritten as follows:

$$\begin{cases} \frac{di}{dt}(t) = -\frac{R}{L_m}i(t) - \frac{K_w}{L_m}w(t) + \frac{1}{L_m}u(t) \\ \frac{dw}{dt}(t) = \frac{K_t}{J}i(t) - \frac{b}{J}\omega(t) \end{cases} \tag{3.5}$$

3.1.2 State Space Description

Now if we let $x_1(t) = i(t)$, $x_2(t) = \omega(t)$ and $y(t) = x_2(t)$ we get:

$$\begin{cases} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} \\ \frac{K_t}{J} & -\frac{b}{J} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_m} \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \end{cases} \tag{3.6}$$

that gives the following standard form:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{3.7}$$

where

$$A = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} \\ \frac{K_t}{J} & -\frac{b}{J} \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{L_m} \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix}.$$

This mathematical form of the system is known in the literature as the state space representation.

Remark 3.1.3 *In this example we assumed that we have access only to the velocity which implies that that* $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$. *If we have access only access to the current or to the two variables the corresponding output matrices become respectively* $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 1 \end{bmatrix}$.

For the state space representation that gives the position, notice that the previous relations of the mathematical model between the voltage and the velocity and the relation that links the velocity and position can be rewritten as follows:

$$\begin{cases} \frac{di}{dt}(t) = -\frac{R}{L_m}i(t) - \frac{K_w}{L_m}w(t) + \frac{1}{L_m}u(t) \\ \frac{dw}{dt}(t) = \frac{K_t}{J}i(t) - \frac{b}{J}\omega(t) \\ \frac{d\theta}{dt}(t) = \omega(t) \end{cases} \tag{3.8}$$

Now if we let $x_1(t) = i(t)$, $x_2(t) = \omega(t)$, $x_3(t) = \theta(t)$ and $y(t) = x_3(t)$ we get:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} & 0 \\ \frac{K_t}{J} & -\frac{b}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_m} \\ 0 \\ 0 \end{bmatrix} u(t) \tag{3.9}$$

$$y(t) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} \tag{3.10}$$

that gives the standard form (3.7) with:

$$A = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} & 0 \\ \frac{K_t}{J} & -\frac{b}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{L_m} \\ 0 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}.$$

To use these models, we need to know the different parameters in each one. This may be in some circumstances difficult to measure and therefore another alternate is required. In the next section, the situation will overcome by using the identification techniques.

Remark 3.1.4 *It is important to mention that the state space description is not unique, which means that for a given system, we can find many state space description. This matter will be explained later in this volume.*

Remark 3.1.5 *Notice that in general, the manufacturer of dc motors provides the data sheet in which we can find all these parameters that correspond to a sample that was chosen for test. These parameters may be not identical to those of the dc motor we are using and this may cause some error in modeling. The feedback control will cope with such errors.*

As a second example, we consider the model of the Segway (see [6]). The dynamics of this system is composed of two models that will be decoupled under some appropriate assumptions. We assume that the Segway remains close to its vertical

position when moving with small speed and the wheels remain in touch with the ground and don't slip.

Under these assumptions the dynamics of our Segway will be partitioned into two parts. The first one gives the behavior of the tilt and linear displacement dynamics while the second one governs the heading angle dynamics. Now if we define the variables of the Table 3.1.

Table 3.1 Variables definition

Variable	definition
$\psi(t)$	tilt angle
$x(t)$	linear position
$\theta_i(t)$	motors' shaft angle
$\theta_o(t)$	gear box shaft angle
$\theta(t)$	wheels' angle
$T_i(t)$	torque delivered to a gear box by one of the dc motors
$T(t)$	torque delivered to a wheel by one of the dc motors
$F(t)$	resultant force between the ground and each of the wheels
$u_x(t)$	motors' voltage input controlling tilt and linear displacement
r_w	wheels' radius
M	mass of the half robot including one wheel
m_b	mass of half the body of the robot
m_w	mass of one of the wheels
J_b	moment of inertia of half the body of the robot
J_w	moment of inertia of one of the wheels
d	distance between motors' shafts and center of gravity of the body
K_t	motors' torque constant
K_e	motors' back emf constant
r_a	motors' armature resistance
r_g	gear boxes' ratio
η	gear boxes' Efficiency
C_f	rotational damping constant

and noticing that the following relations hold always:

$$
\begin{aligned}
M &= m_b + m_w \\
K_t &= K_e = K \\
x(t) &= r_w\theta(t) \\
\theta_i(t) &= r_g\theta_o(t) \\
\theta_o(t) &= \theta(t) + \psi(t) \\
F(t) &= M\ddot{x}(t)
\end{aligned}
$$

we have the following relations:

• motor' dynamics

$$
\begin{aligned}
T_i(t) &= K_t i(t) \\
i(t) &= \frac{u_x(t)}{r_a} - \frac{K_e \dot{\theta}_i(t)}{r_a} \\
T_i(t) &= K_t \left[\frac{u_x(t)}{r_a} - \frac{K_e \dot{\theta}_i(t)}{r_a} \right]
\end{aligned}
\tag{3.11}
$$

$$J_b \ddot{\psi} = \Sigma M$$
$$J_b \ddot{\psi} = m_b g d \sin \psi + T - C_f \dot{\psi} - C_f \dot{\theta}$$
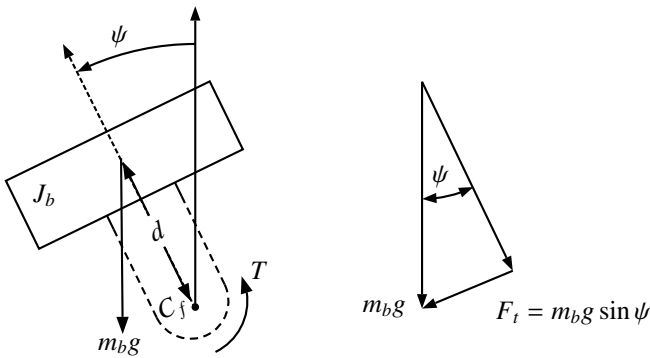
Fig. 3.2 Tilt dynamics free body diagram

$$J_w \ddot{\theta} = \Sigma M$$
$$J_w \ddot{\theta} = T - F r_w - C_f \dot{\theta} - C_f \dot{\psi}$$
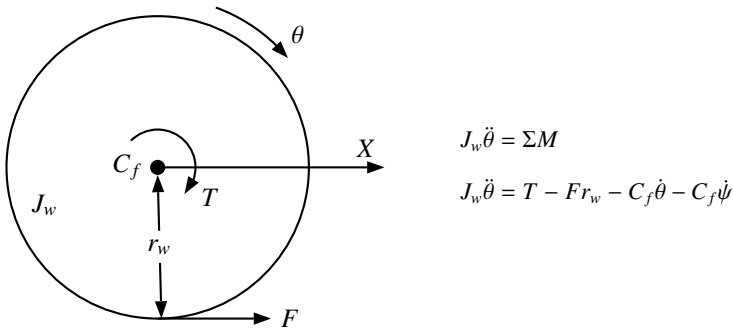
Fig. 3.3 Wheels and linear displacement free body diagram

- torque applied to the wheels

$$
\begin{aligned}
T(t) &= \eta r_g T_i(t) \\
&= \frac{\eta r_g K u_x(t)}{r_a} - \frac{\eta r_g K^2}{r_a}\dot{\theta}_i(t) \\
T(t) &= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\theta}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) \\
&= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t)
\end{aligned}
\tag{3.12}
$$

- robot tilt dynamics, referring to Fig. 3.2 we have:

$$
\begin{aligned}
J_b\ddot{\psi}(t) &= m_b g d \sin(\psi(t)) + T(t) - C_f\dot{\psi}(t) - C_f\dot{\theta}(t) \\
&= m_b g d \sin(\psi(t)) + \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) - C_f\dot{\psi}(t) - C_f\frac{\dot{x}(t)}{r_w} \\
\ddot{\psi}(t) &= \frac{m_b g d \sin(\psi(t))}{J_b} - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b}\right]\dot{x}(t) + \frac{\eta r_g K}{r_a J_b}u_x(t)
\end{aligned}
$$

If we assume that $\psi(t)$ is small we get $\sin(\psi(t)) \approx \psi(t)$ which implies in turn:

$$
\begin{aligned}
\ddot{\psi}(t) &= \frac{\eta r_g K}{r_a J_b}u_x(t) + \frac{m_b g d}{J_b}\psi(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b}\right]\dot{\psi}(t) \\
&\quad - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b}\right]\dot{x}(t)
\end{aligned}
\tag{3.13}
$$

- robot wheels and linear displacement dynamics, referring to Fig. 3.3

$$
\begin{aligned}
J_w\ddot{\theta}(t) &= T(t) - F(t)r_w - C_f\dot{\theta}(t) - C_f\dot{\psi}(t) \tag{3.14} \\
&= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) - r_w M\ddot{x}(t) - \frac{C_f}{r_w}\dot{x}(t) - C_f\dot{\psi}(t)
\end{aligned}
$$

which in turn gives:

$$
\left[\frac{J_w}{r_w} + Mr_w\right]\ddot{x}(t) = \frac{\eta r_g K}{r_a}u_x(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w}\right]\dot{x}(t)
$$

and finally, we obtain:

$$
\begin{aligned}
\ddot{x}(t) &= \left[\frac{\eta r_w r_g K}{r_a(J_w + Mr_w^2)}\right]u_x(t) - \left[\frac{\eta r_w r_g^2 K^2 + C_f r_w r_a}{r_a(J_w + Mr_w^2)}\right]\dot{\psi}(t) \\
&\quad - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a(J_w + Mr_w^2)}\right]\dot{x}(t)
\end{aligned}
\tag{3.15}
$$

If we define $\mathbf{x}^\top(t) = \begin{bmatrix} \psi(t) & \dot{\psi}(t) & x(t) & \dot{x}(t) \end{bmatrix}$ and $\mathbf{y}^\top(t) = \begin{bmatrix} \psi(t) & x(t) \end{bmatrix}$, we get the following state space representation:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu_x(t)$$
$$\mathbf{y}(t) = C\mathbf{x}(t)$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{m_b g d}{J_b} & -\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b} & 0 & -\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\eta r_w r_g^2 K^2 + C_f r_w r_a}{r_a(J_w + Mr_w^2)} & 0 & \frac{\eta r_g^2 K^2 + C_f r_a}{r_a(J_w + Mr_w^2)} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \frac{\eta r_g K}{r_a J_b} \\ 0 \\ \frac{\eta r_w r_g K}{r_a(J_w + Mr_w^2)} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

We will now establish the model representing the heading angle dynamics of the robot taking into consideration that an equal but opposite torque has to be applied by the two motors in order to induce a purely rotational motion on the robot without affecting its tilt and linear position. Therefore, an equal but opposite voltage has to be applied to the two motors and this voltage is taken as the input of this system. Here, we are taking the assumption that the robot is staying in the vertical position and that its moment of inertia around the vertical axis is not changing. If we introduce the additional variables of the Table 3.2 and noticing again that the following hold:

Table 3.2 Variables definition

Variable	definition
$\delta(t)$	heading angle
$x_r(t)$	linear position of the right wheel
$x_l(t)$	linear position of the left wheel
$\theta_r(t)$	right wheel angle
$\theta_l(t)$	left wheel angle
$T_r(t)$	torque delivered by the right dc motor
$T_l(t)$	torque delivered by the left dc motor
$F_r(t)$	driving force of right wheel
$F_l(t)$	driving force of left wheel
$u_r(t)$	right motor voltage input
$u_l(t)$	left motor voltage input
$u_h(t)$	motors' voltage input controlling heading
J_d	moment of inertia of the robot around the vertical axis
S	wheel span

$$x_r(t) = r_w \theta_r(t)$$

$$x_l(t) = r_w \theta_l(t)$$

$$\delta(t) = \left[\frac{x_l(t) - x_r(t)}{S} \right]$$

$$u_l(t) = -u_r(t) = u_h(t)$$

$$u_l(t) \quad -u_r(t) = 2u_h(t)$$

we have the following relations:

- from (3.14), solving for $F(t)$, we have:

$$
\begin{aligned}
F(t) &= \frac{T(t) - J_w \ddot\theta(t) - C_f \dot\theta(t) - C_f \dot\psi(t)}{r_w} \\
&= \frac{\eta r_g K}{r_a r_w} u(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w} \right] \dot\psi(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2} \right] \dot x(t) - \frac{J_w}{r_w} \ddot\theta(t)
\end{aligned}
$$

now making reference to left and right, we get:

$$F_l(t) = \frac{\eta r_g K}{r_a r_w} u_l(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w} \right] \dot\psi(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2} \right] \dot x_l(t) - \frac{J_w}{r_w} \ddot\theta_l(t)$$

$$F_r(t) = \frac{\eta r_g K}{r_a r_w} u_r(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w} \right] \dot\psi(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2} \right] \dot x_r(t) - \frac{J_w}{r_w} \ddot\theta_r(t)$$
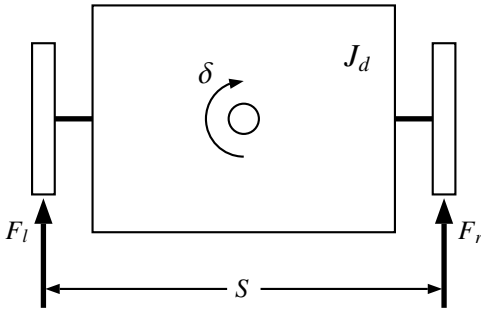
$$J_d \ddot\delta = \Sigma M$$

$$J_d \ddot\delta = F_l \frac{S}{2} - F_r \frac{S}{2}$$

$$J_d \ddot\delta = (F_l - F_r)\frac{S}{2}$$

Fig. 3.4 Heading dynamics free body diagram

- referring to Fig. 3.4 we get:

$$
\begin{aligned}
J_d \ddot{\delta}(t) &= [F_l(t) - F_r(t)] \frac{S}{2} \\
&= \frac{\eta r_g K S}{2 r_a r_w} [u_l(t) - u_r(t)] + \left[\frac{\eta S r_g^2 K^2 + S C_f r_a}{2 r_a r_w^2} \right] [\dot{x}_r(t) - \dot{x}_l(t)] + \frac{J_w S}{2 r_w} \left[\ddot{\theta}_r(t) - \ddot{\theta}_l(t) \right] \\
&= \frac{\eta r_g K S}{r_a r_w} u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{2 r_a r_w^2} \right] \dot{\delta}(t) - \frac{J_w S^2}{2 r_w^2} \ddot{\delta}(t)
\end{aligned}
$$

which in turn gives:

$$
\left[J_d + \frac{J_w S^2}{2 r_w^2} \right] \ddot{\delta}(t) = \frac{\eta r_g K S}{r_a r_w} u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{2 r_a r_w^2} \right] \dot{\delta}(t)
$$

and finally, we obtain:

$$
\ddot{\delta}(t) = \left[\frac{2 \eta r_w r_g K S}{r_a (2 J_d r_w^2 + J_w S^2)} \right] u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{r_a (2 J_d r_w^2 + J_w S^2)} \right] \dot{\delta}(t) \quad (3.16)
$$

If we define $\mathbf{x}_h^\top(t) = \left[\delta(t) \; \dot{\delta}(t) \right]$ and $\mathbf{y}_h(t) = \delta(t)$, we get the following state space representation:

$$
\begin{aligned}
\dot{\mathbf{x}}_h(t) &= A_h \mathbf{x}_h(t) + B_h u_h(t) \\
\mathbf{y}_h(t) &= C_h \mathbf{x}_h(t)
\end{aligned}
$$

where

$$
A_h = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{r_a (2 J_d r_w^2 + J_w S^2)} \end{bmatrix}
$$

$$
B_h = \begin{bmatrix} 0 \\ \frac{2 \eta r_w r_g K S}{r_a (2 J_d r_w^2 + J_w S^2)} \end{bmatrix}, C_h = \begin{bmatrix} 1 & 0 \end{bmatrix}
$$

The last example is the magnetic levitation system. This system is represented by the Fig. 4.10. The data of this system are summarized in the Table 3.3.

Table 3.3 Data of the magnetic levitation system

Variable	value
R	62.7 Ω
L	60 mH
m (object mass)	7.64 g
diameter of the permanent magnet	9 mm

Let $x(t)$ denote the position of the object at time t measured from the bottom of the coil. The dynamics of the moving object is described by the following differential equation:

$$m\ddot{x}(t) = mg - F_c - F_p \tag{3.17}$$

where g is the gravity, F_c and F_p are the magnetic forces generated respectively by the coil and the permanent magnet.

Remark 3.1.6 *It is important to notice that the direction of the magnetic force F_c is linked to the direction of the current in the coil.*

If we denote by $i(t)$ the current at time t that give a force F_c pointing down at time t with the following expression:

$$F_c(t) = k_c \frac{i^2(t)}{x^2(t)}$$

The permanent force F_p is given by the following expression:

$$F_p(t) = k_p \frac{1}{x^2(t)}$$

Using these expressions we get:

$$m\ddot{x}(t) = mg - k_c \frac{i^2(t)}{x^2(t)} - k_p \frac{1}{x^2(t)} \tag{3.18}$$

From the other side, we have the following relation between the current $i(t)$ and the applied voltage $u(t)$:

$$u(t) = Ri(t) + L\frac{di(t)}{dt}$$

If we neglect the effect of the coil, this relation becomes:

$$u(t) = Ri(t)$$

and the dynamics become:

$$m\ddot{x}(t) = mg - k_c \frac{u^2(t)}{x^2(t)} - k_p \frac{1}{x^2(t)} \tag{3.19}$$

For the output equation notice that we have a Hall effect sensor that generates a voltage that is function of the position, $x(t)$ of the object and therefore it is function of the magnetic field B (sum of the two fields (the one due to the coil and the one due to permanent magnet)). If we denote by $y(t)$ this voltage and using the data sheet of this sensor, we get:

$$y(t) = 0.003125B + 2.5$$

where B is measured in Gauss (1 Telta = 1000 Gauss).

This gives the following one:

$$y(t) = 31.25B + 2.5$$

where B is measured in Tesla.

It can be shown that the expression of the total magnetic field is given by:

$$B = C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2$$

where $C_p = -1.9446 \ 10^{-8}$, $C_b = -0.1671$, $C_1 = -0.011027$ and $C_2 = 0.003568$.

In conclusion the output of the sensor is then given by:

$$y(t) = \left[\frac{1}{0.032} \left[C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2 \right] \right] + 2.5$$

It can seen that the model is nonlinear and the theory we will present in this volume will not help. Therefore a linearization around an equilibrium point is required. At the equilibrium point the speed and the acceleration of the object are equal to zero and the current is constant in time and the total force is equal to the gravitational force. Using this and the previous dynamics, we get:

$$\begin{cases} x^2(t) = sign(u(t)) \frac{k_c}{mgR^2} u^2(t) + \frac{k_p}{mg} \\ y(t) = \left[\frac{1}{0.032} \left[C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2 \right] \right] + 2.5 \end{cases}$$

Using these conditions and some appropriate experiments, we can determine the values for k_c and k_p and these values are given by:

$$k_c = 5.9218 \ 10^{-4}$$
$$k_p = 4.0477 \ 10^{-6}.$$

At the equilibrium point, the object occupies a fixed position x_e that corresponds to the voltage u_e ($u_e = Ri_e$). The corresponding voltage delivered by the sensor is y_e. In the neighborhood of this equilibrium point (x_e, u_e, i_e, y_e), the system has a linear behavior. The linearized model is given by ([1]):

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

where

$$x(t) = \begin{bmatrix} x_1(t)(position) \\ x_2(t)(velocity) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ \frac{2\left[sign(u_e)k_c u_e^2 + k_p R^2 \right]}{mR^2 x_e^3} & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \frac{-2 sign(u_e)k_c u_e}{mR^2 x_e^2} \end{bmatrix}$$

$$C = \begin{bmatrix} \frac{-3C_p}{0.032 x_e^4} & 0 \end{bmatrix}$$

$$D = \frac{C_b}{0.032R}$$

3.2   Identification

From the previous example, it can be seen from that the establishment of the mathematical model that we can use for analysis and design is not an easy task and even if we can get the model from physics laws, the value of the different parameters of the model may be impossible to get and therefore the analytical model is useless.

System identification is a procedure by which a mathematical description of a dynamical system is extracted from test data. The aim of the identification is to construct an algorithm that will allow to build a mathematical model from observed data. Mainly the system we would like to model is seen as black box with some inputs and outputs that are collected at some finite instants of time.

The establishment of an appropriate model for a given linear time invariant system can be done into two steps. Firstly, a structure of a model that may fit with the collected data is chosen and then the parameter of this model are determined.

The identification problem can be stated as follows: given N samples of the pair $(u(k), y(k))$ where $u(k)$ and $y(k)$ denote respectively the input and output collected from experiments on the real system, we wish to determine the system's parameters of the chosen model such that it matches the real system sufficiently well.

3.2.1   Transfer Function Approach

One of the approaches that we may use to build a model with transfer function description is the least-square system identification. To show how this algorithm works, let us assume the structure of the chosen model is given by:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_1 z^{n-1} + b_2 z^{n-2} + \cdots + b_n}{z^n - a_1 z^{n-1} - \cdots - a_n}$$

where $Y(z)$ and $U(z)$ are respectively $\mathscr{Z}$-transform[1] of the output $y(k)$ and $u(k)$, $a_1, \cdots, a_n$ and $b_1, \cdots, b_n$ are the model parameters that we have to determine.

Using $\mathscr{Z}$-transform inverse we get following model:

$$\begin{aligned} y(k) &= a_1 y(k-1) + a_2 y(k-2) + \cdots + a_n y(k-n) \\ &\quad + b_1 u(k-1) + b_2 u(k-2) + \cdots + b_n u(k-n) \end{aligned}$$

The objective of the identification procedure is to determine the model parameters from measurements of the inputs, $u(k), k = 0, 1, \cdots, N$ and the corresponding outputs, $y(k), k = 0, 1, \cdots, N$. For this purpose let:

$$\theta = \begin{bmatrix} a_1 & a_2 & \cdots & a_n & b_1 & b_2 & \cdots & b_n \end{bmatrix} \tag{3.20}$$

Let us now assume that we collected $N + 1$ measurements pairs:

$$(u(0), y(0)), (u(1), y(1)), \cdots, (u(N), y(N))$$

with $N > n$.

---

[1] The definition of the $\mathscr{Z}$-transform will be given later in this book

By defining $f(k)$ as follows:

$$f^\top(k) = \left[y(k-1)\, y(k-2)\, \cdots\, y(k-n)\, u(k-1)\, u(k-2)\, \cdots\, u(k-n) \right]$$

then for the sample periods $n, n+1, \cdots, N$ we have:

$$\begin{cases} y(n) = f^\top(n)\theta + e(n) \\ y(n+1) = f^\top(n+1)\theta + e(n+1) \\ \quad\vdots \\ y(N) = f^\top(N)\theta + e(N) \end{cases} \tag{3.21}$$

where $e(k)$ is the error estimation at period kT.

If we define $\mathbf{y}(N)$, $\mathbf{f}(N)$ and $\mathbf{e}(N)$ as follows:

$$\mathbf{y}(N) = \begin{bmatrix} y(n) \\ y(n+1) \\ \vdots \\ y(N) \end{bmatrix}, \mathbf{f}(N) = \begin{bmatrix} f^\top(n) \\ f^\top(n+1) \\ \vdots \\ f^\top(N) \end{bmatrix}, \mathbf{e}(N) = \begin{bmatrix} e(n) \\ e(n+1) \\ \vdots \\ e(N) \end{bmatrix}$$

then the previous relation becomes:

$$\mathbf{y}(N) = \mathbf{f}(N)\theta + \mathbf{e}(N) \tag{3.22}$$

where $\mathbf{y}(N) \in \mathbb{R}^{N-n+1}$, $\mathbf{e}(N) \in \mathbb{R}^{N-n+1}$, $\mathbf{f}(N) \in \mathbb{R}^{(N-n+1)\times 2n}$ and $\theta \in \mathbb{R}^{2n}$

Using now the least square algorithm with the following cost:

$$J(\theta) = \sum_{k=n}^{N} e^2(k) = \mathbf{e}^\top(N)\mathbf{e}(N) \tag{3.23}$$

This implies the following:

$$\begin{aligned} J(\theta) &= [\mathbf{y}(N) - \mathbf{f}(N)\theta]^\top [\mathbf{y}(N) - \mathbf{f}(N)\theta] \\ &= \mathbf{y}^\top(N)\mathbf{y}(N) - \theta^\top \mathbf{f}^\top(N)\mathbf{y}(N) - \mathbf{y}^\top(N)\mathbf{f}(N)\theta + \theta^\top \mathbf{f}^\top(N)\mathbf{f}(N)\theta \\ &= \mathbf{y}^\top(N)\mathbf{y}(N) - 2\theta^\top \mathbf{f}^\top(N)\mathbf{y}(N) + \theta^\top \mathbf{f}^\top(N)\mathbf{f}(N)\theta \end{aligned} \tag{3.24}$$

To search for the optimal solution $\theta^\star$ that minimizes the cost $J(\theta)$, we can use the optimization conditions (see [3]). By these optimality conditions, we get:

$$\frac{\partial J(\theta)}{\partial \theta} = -2\mathbf{f}^\top(N)\mathbf{y}(N) + 2\mathbf{f}^\top(N)\mathbf{f}(N)\theta^\star = 0$$

that can be rewritten as:

$$\mathbf{f}^\top(N)\mathbf{f}(N)\theta^\star = \mathbf{f}^\top(N)\mathbf{y}(N)$$

from which we obtain the optimal solution as follows:

$$\theta^\star = \left[\mathbf{f}^\top(N)\mathbf{f}(N) \right]^{-1} \mathbf{f}^\top(N)\mathbf{y}(N)$$

provided that the matrix, $[\mathbf{f}^\top(N)\mathbf{f}(N)]$, is not singular.

Remark 3.2.1 *The formula we just developed allows us to compute the parameters off-line after collecting the data. But in some applications we may need to compute these parameters on-line and therefore adapt the controller's parameters as it is the case for adaptive control. This can be done using the recursive form of the least square algorithm.*

To establish the recursive algorithm, we will use some forgetting factors. Consequently, the cost is modified to:

$$
\begin{aligned}
J(\theta) &= \mu(n)e^2(n) + \mu(n+1)e^2(n+1) + \cdots + \mu(N)e^2(N) \\
&= \sum_{k=n}^{N} \mu(k)e^2(k) = \mathbf{e}^\top(N)\mathbf{F}(N)\mathbf{e}(N)
\end{aligned}
\tag{3.25}
$$

where $\mathbf{F}(N)$ is a diagonal matrix, $F(N) = \mathrm{diag}(\mu(1), \cdots, \mu(N))$.

Proceeding similarly as we did previously, we get:

$$
\theta^\star = \left[\mathbf{F}^\top(N)\mathbf{F}(N)\mathbf{f}(N)\right]^{-1}\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N)
\tag{3.26}
$$

Remark 3.2.2 *The forgetting factors are used to give more weight for the recent data.*

Let $\mu(k) = \alpha\beta^{N+1-k}$, with $\beta \le 1$. Based now on the expression of $\theta^\star$, we get:

$$
\begin{aligned}
\mathbf{f}^\top(N+1)\mathbf{F}(N+1)\mathbf{f}(N+1) &= \sum_{k=n}^{N+1} \alpha\beta^{N+1-k} f(k)f^\top(k) \\
&= \sum_{k=n}^{N} \alpha\beta\beta^{N-k} f(k)f^\top(k) + \alpha f(N+1)f^\top(N+1)
\end{aligned}
$$

Let us now define $\Psi(k)$ as follows:

$$
\Psi(k) = \left[\mathbf{f}^\top(k)\mathbf{F}(k)\mathbf{f}(k)\right]^{-1}
\tag{3.27}
$$

Using this we get:

$$
\Psi^{-1}(N+1) = \beta\Psi^{-1}(N+1) + \alpha f(N+1)f^\top(N+1)
\tag{3.28}
$$

Using now the following relation:

$$
[A + BCD]^{-1} = A^{-1} - A^{-1}\left[C^{-1} + DA^{-1}B\right]^{-1} DA^{-1}
$$

the previous relation becomes:

$$
\begin{aligned}
\Psi(N+1) &= \beta^{-1}\Psi(N) - \beta^{-1}\Psi(N)f(N+1) \\
&\quad \times \left[\alpha^{-1} + \beta^{-1}f^\top(N+1)\Psi(N)f(N+1)\right]^{-1} \beta^{-1}f^\top(N+1)\Psi(N)
\end{aligned}
$$

For the second term in the expression of $\theta^\star$ we have:

$$
\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N) = \begin{bmatrix} f(n) & \cdots & f(N+1) \end{bmatrix} \begin{bmatrix} \alpha\beta^{N+1-n} & & \\ & \ddots & \\ & & \alpha \end{bmatrix} \begin{bmatrix} y(n) \\ \vdots \\ y(N) \\ y(N+1) \end{bmatrix}
$$

$$
= \beta\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N) + \alpha f(N+1)y(N+1)
$$

Combining the previous relations and after some algebraic manipulations we get:

$$
\Psi(N) = [\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{f}(N)]^{-1}
$$
$$
Q(N+1) = \beta^{-1}\Psi(N)f(N+1)\left[\alpha^{-1} + \beta^{-1}f^\top(N+1)\Psi(N)f(N+1)\right]^{-1}\beta^{-1}f^\top(N+1)\Psi(N)
$$
$$
\theta(N+1) = \theta(N) + Q(N+1)\left[y(N+1) - f^\top(N+1)\theta(N)\right]
$$
$$
\Psi(N+1) = \beta^{-1}\left[\mathbb{I} - Q(N+1)f^\top(N+1)\right]\Psi(N)
$$

which apply for $N \geq n$.

Example 3.2.1 *To show how to use this technique to identify a given system, let us consider the setup of the dc motor kit. It consists of a dc motor driving a mechanical load. We know that the system a single input single output and its transfer function between the speed of the shaft and the voltage is a first order of the following form:*

$$
G(s) = \frac{K_m}{\tau_m s + 1}
$$

where K_m and τ_m are the two parameters that we have to determine.

For this system we can use two ways to get the model. The first one consists of getting the data $((u(k), y(k))$ using an UART to communicate with a PC and then use the least square method to build the model. The second one consists of using the microcontroller and then take the system as a black box.

For the second method the gain K_m is determined at the steady state regime as the ratio between the output and the input voltage. While for the time constant, τ_m, we take it as the instant at which the output takes 63 % of the steady state value of the output. This procedure can be programmed in our microcontroller and easily the model is established. We use this approach in our mechatronics laboratory.

3.2.2   State Space Description Approach

Consider a dynamical system system described by the following state space description:

$$
\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}
$$

where $x(k) \in \mathbb{R}^n$ is the state and $u(k) \in \mathbb{R}^m$ is the input and $y(k) \in \mathbb{R}^l$ be the output.

It is important to notice that the state description is not unique and any transformation: $\tilde{x}(k) = T^{-1}x(k)$, with T a nonsingular matrix, will give another description:

$$\begin{cases} \tilde{x}(k+1) = \tilde{A}\tilde{x}(k) + \tilde{B}u(k) \\ y(k) = \tilde{C}\tilde{x}(k) + \tilde{D}u(k) \end{cases}$$

where $\tilde{A} = T^{-1}AT$, $\tilde{B} = T^{-1}B$, $\tilde{C} = CT$ and $\tilde{D} = D$.

To determine the model of this system we need to determine the matrices A, B, C and D. If the system is single input single output, we can compute the transfer function and proceed with the previous approach to establish the mathematical model. In the state space description, we try to determine the state space description (A, B, C, D) that matches the set of input-output data. In the literature there exist many approaches to identify system in state space description. The reader is invited to consult the literature for this topic. Here we will present a simple algorithm that can be used to determine the state space description.

Before presenting this algorithm we will establish some relations that the algorithm uses in its computation. If we denote by $u(k), u(k+1), \ldots, y(k), y(k+1), \ldots$ and $x(k), x(k+1), \ldots$ the sequences of inputs, outputs and states, it can be shown that the Hankel matrix, Y_h can be given by:

$$Y_h = \Gamma_i X + H_t U_h$$

where:

$$Y_h = \begin{bmatrix} y(k) & y(k+1) & \cdots & y(k+j-1) \\ y(k+1) & y(k+2) & \cdots & y(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+i-1) & y(k+i) & \cdots & y(k+j+i-2) \end{bmatrix}$$

U_h is a Hankel block with the same block dimensions as Y_h containing the consecutive inputs

$$U_h = \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+j-1) \\ u(k+1) & u(k+2) & \cdots & u(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+i-1) & u(k+i) & \cdots & u(k+j+i-2) \end{bmatrix}$$

X contains the consecutive state vectors:

$$X = \begin{bmatrix} x(k) & x(k+1) & \cdots & x(k+j-1) \end{bmatrix}$$

Γ_i as the extended observability matrix:

$$\Gamma_i \;=\; \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{i-1} \end{bmatrix}$$

and H_t is the block Toeplitz matrix:

$$H_t \;=\; \begin{bmatrix} D & 0 & 0 & \cdots & 0 \\ CB & D & 0 & \cdots & y(k+j) \\ CAB & CB & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \cdots & D \end{bmatrix}$$

Let H be the concatenation of the matrices H_1 and H_2, i.e.:

$$H = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

with

$$H_1 \;=\; \begin{bmatrix} Y_{h1} \\ U_{h1} \end{bmatrix}$$

$$H_2 \;=\; \begin{bmatrix} Y_{h2} \\ U_{h2} \end{bmatrix}$$

where

$$Y_{h1} \;=\; \begin{bmatrix} y(k) & y(k+1) & \cdots & y(k+j-1) \\ y(k+1) & y(k+2) & \cdots & y(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+i-1) & y(k+i) & \cdots & y(k+j+i-2) \end{bmatrix}$$

$$Y_{h2} \;=\; \begin{bmatrix} y(k+i) & y(k+i+1) & \cdots & y(k+i+j-1) \\ y(k+i+1) & y(k+i+2) & \cdots & y(k+i+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+2i-1) & y(k+2i) & \cdots & y(k+j+2i-2) \end{bmatrix}$$

$$U_{h1} \;=\; \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+j-1) \\ u(k+1) & u(k+2) & \cdots & u(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+i-1) & u(k+i) & \cdots & u(k+j+i-2) \end{bmatrix}$$

$$U_{h2} \;=\; \begin{bmatrix} u(k+i) & u(k+i+1) & \cdots & u(k+i+j-1) \\ u(k+i+1) & u(k+i+2) & \cdots & u(k+i+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+2i-1) & u(k+2i) & \cdots & u(k+j+2i-2) \end{bmatrix}$$

that satisfy:

$$Y_{h1} = \Gamma_i X_1 + H_t U_{h1}$$
$$Y_{h2} = \Gamma_i X_2 + H_t U_{h2}$$

with

$$X_1 = \begin{bmatrix} x(k) \ x(k+1) \ \cdots \ x(k+j-1) \end{bmatrix}$$
$$X_2 = \begin{bmatrix} x(k+i) \ x(k+i+1) \ \cdots \ x(k+i+j-1) \end{bmatrix}$$

The following algorithm can be used to compute the matrices (A, B, C, D) in off-line:

1. calculate U and S in the SVD of H:

$$H = USV^\mathsf{T} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & 0 \\ 0 & 0 \end{bmatrix} V^\mathsf{T}$$

2. calculate the SVD of $U_{12}^\mathsf{T} U_{11} S_{11}$

$$U_{12}^\mathsf{T} U_{11} S_{11} = \begin{bmatrix} U_q & U_q^\perp \end{bmatrix} \begin{bmatrix} S_q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_q^\mathsf{T} \\ V_q^{\perp\mathsf{T}} \end{bmatrix}$$

3. solve the following set of linear equations:

$$\begin{bmatrix} U_q^\mathsf{T} U_{12}^\mathsf{T} U(m+l+1 : (i+1)(m+l), :)S \\ U(mi+li+m+1 : (m+l)(i+1), :)S \end{bmatrix} =$$
$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} U_q^\mathsf{T} U_{12}^\mathsf{T} U(1 : mi+li, :)S \\ U(mi+li+1 : mi+li+m, :)S \end{bmatrix}$$

This algorithm can be programmed to get the matrices (A, B, C, D) of the system. Another version for on-line computation can be also obtained and for interested reader we refer to the literature for more details.

3.3   Conclusions

In this chapter, we covered the mathematical modeling of dynamical systems. We presented the technique that uses the physics laws to generate the model. We also developed the identification technique that may be used in some circumstances to establish a valid model that describes adequately the dynamical system under consideration. Both of the techniques require experiment data to establish the desired model.

3.4   Problems

1. In this problem, we ask to build the mathematical model for the dc motor kit without neglecting L. We ask to establish:

   (a) the transfer function

   (b) the state space description

2. Establish the mathematical model of the two wheels robot

3. Establish the mathematical model for the levitation system

4. Consider a dynamical system with a transfer function that you can give. Write a Matlab program that generates a sequence of data $(u(k), y(k))$. Using this data, write a program in Matlab to identify the system and to establish the mathematical model. Compare the two models and conclude.

5. Consider a dynamical system in state space description. Write a Matlab prgram to generate the appropriate data to identify the system using the state space description approach. Using this data write a Matlab program to establish a state space description and compare it with the original one.

Part III

Transfer Function Approaches

Mechatronic systems are in general a combination of hardware and software to assure the desired tasks for which the system was designed for. The analysis and design of such mechatronic systems can be done using different approaches. Among these approaches we quote the ones based on transfer function and the ones using the state space techniques.

This part deals with the analysis and synthesize of mechatronic systems using the transfer function approach. Mainly, we will focus on the analysis of dynamical systems controlled by microcontrollers. We will learn how to determine the performances that the system has. The design of controllers is also tackled. Mainly, we will see how to state the control design problem and how to solve it. The design part focuses on the determination of the controller that gives the desired performances to our dynamical system.

This part is divided into two chapters. The first one treats the analysis tools. Particularly, we will see how we can transform a continuous-time model to a discrete-time version by choosing appropriately the sampling period. Once this period is chosen, a discrete-time version of the model of the system under consideration is obtained that will be used for analysis and design of the dynamical system under study. The first chapter in this part covers the different tools that we may use to get the system's performances. The second chapter presents the design techniques that may be used to design the appropriate controller that will guarantee the desired performances. The design approach is composed of two steps. The first one, determines what will be the structure of the appropriate controller that can guarantee the desired performances. The second one computes the controller's parameters. Some simulations results are needed before implementing the developed algorithm. Matlab and Simulink are used for this purpose.

4

Analysis Based on Transfer Function

After reading this chapter the reader will:

1. master the concept of the transfer function concept

2. be able to perform the analysis of any LTI system and determine the specifications that the system has

3. be able to compute any time response of LTI system for a given input

4. be able to check the stability of any dynamical system

5. plot the root locus of any LTI system and use it for analysis and design purpose

6. plot the Bode diagram of any LTI system and use it for analysis and design purpose

4.1   Introduction

Nowadays the microcontrollers are more powerful and their prices are affordable which makes their use attractive. In mechatronic systems they are used either for On/Off or continuous-time controls. In both cases, the microcontroller is the hearth

of the mechatronic system. In the On/Off case, it is used for security and control purposes. The algorithm in this case is easy and doesn't take time in general to compute the action to be taken. While for the continuous-time case, the microcontroller receives the data at each sampling period and compute the desired action according to a chosen algorithm. The computation in this case may take more time and more care should be taken to prevent surprises. In both cases interrupts are used. The microcontrollers we will use in this book must have a quite high processing speed.

In practice when controlling real processes using microcontrollers two structures can be adopted. In the first one, the error between the output and the reference is done in continuous-time and then sent to the microcontroller via analog-digital converter (A/D) and the control action is computed following the chosen algorithm, while in the second case, the output is converted to a digital value via a A/D. The reference in this case is fed in a digital form. The control action is computed in a similar way as for the first case. These two structures are illustrated respectively by Figs. 4.1 and 4.2.
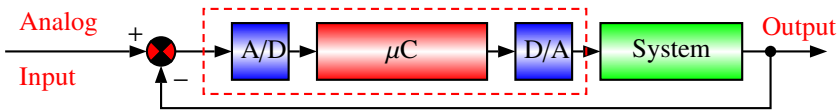
Fig. 4.1 Signal conversion is made in the forward path
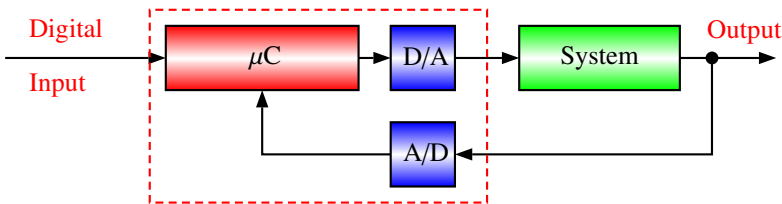
Fig. 4.2 Signal conversion is made in the feedback path

The second structure is more often used in practice and therefore, it is the one that we will use in the rest of the book.

Remark 4.1.1 *In the structure of Fig. 4.2, we have sampled signals that have the following advantages:*

1. *easy modulated*

2. *easy to code*

3. *easy to transmit and to regenerate*

and there are positive points.

In the rest of this chapter we will present the tools that can be used for the analysis of this type of system. Mainly we will show for a given practical system:

1. how to determine the sampling period

2. how to convert the continuous-time model to discrete-time one using the chosen sampling period

3. how to determine the performances of such system such as the stability, the overshoot, the settling time, etc.

4. how to use the root-locus and the Bode-plot techniques for discrete-time case

All these questions are addressed in the rest on this chapter. The rest of the chapter is organized as follows. In Section 2, the sampling process is developed and the relation between the continuous-time and the discrete-time is established. Mainly, the relationship between the poles is established for the two domains (s-domain and z-domain). Section 3 introduces the transfer function concept and the one of poles and zeros. In Section 4, the time response for a given input is developed and the approaches to compute it are presented. Section 5 covers the stability problem and the system error. The techniques of root locus and the Bode plot are developed respectively in Sections 6 and 7. These techniques are used in the analysis and design phases.

4.2   Sampling Process

Real practical processes are more often continuous systems that evolve continuously in time. Their analysis and design should be done carefully. In fact we will always need to convert the dynamics of such systems to a discrete-time corresponding one to analyze and proceed with the design of the controller that will be used to control them using microcontrollers. The choice of the sampling period is a critical problem. In fact a small sampling period will result with huge amount of data that the microcontroller will not be able to handle while a large one will give poor results and may be the system will not be controlled properly. The sampling period must be properly chosen to avoid such problems. It can be chosen smaller than the smallest time constant in the dynamics of the process. The bandwidth is also used for the choice of the sampling period.

The Shannon theorem is used for the choice of the sampling period. This statement of the Shannon theorem is given by the following result:

Theorem 4.2.1 *A signal y(t) which contains no frequency components greater than f_h is uniquely reconstructed by a set of sampled from y(t) spaced by $\frac{1}{2f_h}$. A proper choice for the sampling frequency should satisfy:*

$$f_s \geq 2f_h \tag{4.1}$$

Remark 4.2.1 *In practice the factor two is not enough and generally we choose the number more greater than two. A good choice consists of taking the sampling rate greater than $30w_b$, where w_b is the bandwidth of the closed-loop system.*

Remark 4.2.2 *It is important to notice that we have the following relations between the period, T, the frequency, f and the pulsation, w:*

$$T = \frac{1}{f}$$
$$w = 2\pi f$$

which implies:

$$w = \frac{2\pi}{T}$$

Example 4.2.1 *To show how the sampling period can be chosen for a continuous-time system, let us consider a dynamical system with the following transfer function:*

$$G(s) = \frac{10}{(s+1)(s+2)(s+5)}$$

and determine the sampling period for this system.

First of all notice that there exist an infinite number of sampling of periods that can be chosen for this system. In this example we define the sampling period using two approaches.

From the transfer function of the system we conclude that the highest frequency in the system is $w_b = 5rad/s$. This corresponds to faster dynamics in the system and therefore when sampling we should use this information and sample faster than this.

Notice that we have $w_s T = 2\pi$. Now if we sample thirty times of the highest frequency in the system, we have $w_s = 30 \times 5 = 150 rad/sec$. This gives:

$$T = \frac{2\pi}{150} = 0.021s$$

From the other side, the constant times of the system are respectively $\tau_1 = 1s$, $\tau_2 = 0.5s$ and $\tau_1 = 0.2s$. This implies that fast dynamics in the system has a time constant equal to $0.2s$. A rule to select the sampling period consists of using the following formula:

$$T = \frac{0.2}{a} \tag{4.2}$$

where a is positive real number to be selected between 7 and 14. A proper choice is 10.

Using this rule we get

$$T = 0.02s$$

Once the sampling period is chosen the next step is to convert the continuous-time dynamics to an equivalent discrete-time one. In fact, if the sampling period,

T, is properly chosen the real output can be obtained from the sampled one for a given input and therefore there is no lost of information.

The conversion from continuous-time system to sampled system passes through two devices:

- sampler

- zero-order-hold (ZOH)

The role of the sampler is to convert the continuous-time signal to an equivalent train of pulses while the ZOH blocks the values received from the sampler to make them available to the microcontroller that reads them through the analog/digital converter. The sampling process is illustrated in Fig. 4.3.
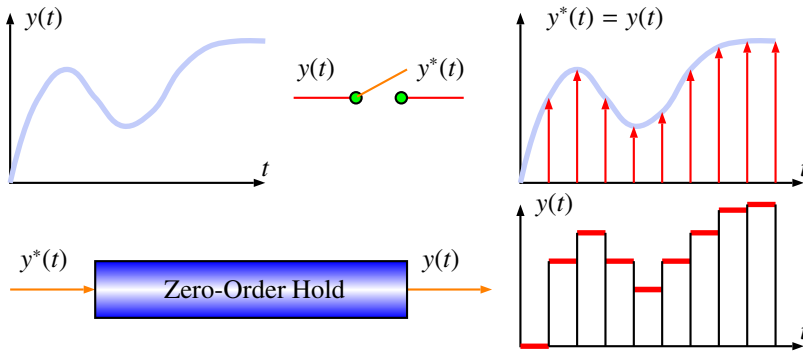
Fig. 4.3 Sampling process

The main objective of the sampling process of a signal $y(t)$ is to keep most of its information in the sampled one. It is also important to notice that the number of bits of the used microcontroller to process this signal has a significant effect on the quantization and therefore on the result. The quantization is the process of approximating the continuous range of values by a set of discrete integer values. It is also referred to as the discretization of the amplitude values of the signal. If a microcontroller with 16 bits is used, we will have $2^{16} = 65536$ possible values per sample.

Let $y(t)$ be an analog signal whose maximum frequency that a sampler should take into account is f_h (bandwidth). Assume $y(t)$ is sampled at frequency f_s. Shannon theorem states that it is possible to reconstruct the signal $y(t)$ from $y^*(t) = y(kT)$ if and only if $f_s \geq 2f_h$. Mathematically, the sampling process of an analog signal can be seen as a mathematical product between the signal $y(t)$ and a train of impulses. This is given by the following expression:

$$y(kT) = \sum_{k=0}^{\infty} y^{\star}(t)\delta(t - kT) \tag{4.3}$$

where $\delta(t)$ is the Dirac impulse and T is the sampling period.
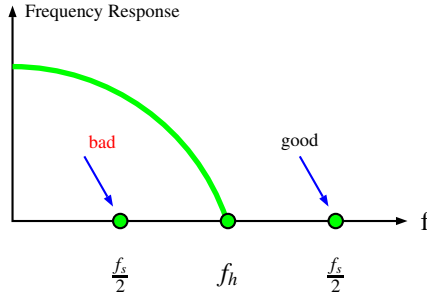
Fig. 4.4 Sampling period choice

For the continuous-time systems, the Laplace transform has been used to transform the set of linear differential equations that describes the dynamics into an algebraic one and the concepts of the transfer function or the transfer matrix function have been defined. Then, the tools for analysis and design that have been developed can be used for this purpose. For sampled systems we will use the same approach since their dynamics are equivalent and the transformation used for the analysis and design called the $\mathscr{Z}$-transform is obtained from the Laplace transform. There exist many similarities between the two transformations.

Let $f^*(t)$ be a sampled signal, such as:

$$\mathscr{L}\left[f^*(t)\right] = F^*(s) = \int_0^{\infty} f^*(t)e^{-st}dt$$

where $f^*(t)$ is equal to zero everywhere except at instants $t = kT$, where $k = 0, 1, 2, 3, \cdots$

As an example of the signal $f(t)$ we give the step signal defined as follows:

$$f(t) = \begin{cases} 1 & \forall t \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Notice that the Laplace transform of $f^*(t)$, yields:

$$\int_0^{\infty} f^*(t)e^{-st}dt = \sum_{k=0}^{\infty} f(kT)e^{-skT} = \sum_{k=0}^{\infty} f(kT)(e^{sT})^{-k} = F^*(s)$$

The $\mathscr{Z}$-transform of $f(t)$ is defined as equal to Laplace transform of $f^*(t)$:

$$\mathscr{Z}\left[f(t)\right] = \mathscr{L}\left[f^*(t)\right]$$

Now, if we define $z = e^{sT}$, then we have:

$$F(z) = \sum_{k=0}^{\infty} f(kT)z^{-k}$$

This expression can be used to compute the $\mathscr{Z}$-transform of any signal.

Example 4.2.2 *Let us now give some examples to show how we use the $\mathscr{Z}$-transform and its properties.*

- *$\mathscr{Z}$ transform of the unit pulse function:*

$$f(k) = \begin{cases} 1 & when\ k = 0 \\ 0 & otherwise \end{cases}$$

  Using the definition of $\mathscr{Z}$-transform, we get:

$$F(z) = \sum_{k=0}^{\infty} f(k)z^{-k} = 1.z^{-0} = 1$$

- *$\mathscr{Z}$-transform of the unit step function:*

$$f(k) = \begin{cases} 1 & when\ k \geq 0 \\ 0 & otherwise \end{cases}$$

  Proceeding similarly we have:

$$F(z) = \sum_{k=0}^{\infty} f(k)z^{-k} = \sum_{k=0}^{\infty} z^{-k} = \sum_{k=0}^{\infty} (z^{-1})^k$$

  This last expression is equivalent to the following series

$$\sum_{k=0}^{\infty} a^k$$

  that will converge if $|a| < 1$ and we get:

$$\sum_{k=0}^{\infty} a^k = \frac{1}{1-a}$$

  To get this relation notice that if we let:

$$S = \sum_{k=0}^{\infty} a^k$$

  Computing $S - aS$ implies the results.
  Using this, we get:

$$F(z) = \frac{1}{1 - z^{-1}}$$

- $\mathcal{Z}$-transform of the exponential function:

$$f(k) = \begin{cases} a^k & when\ k = 0 \\ 0 & otherwise \end{cases}$$

  Using again the definition of $\mathcal{Z}$-transform, we get:

$$F(z) = \sum_{k=0}^{\infty} a^k z^{-k} = \sum_{k=0}^{\infty} (az^{-1})^k = \frac{1}{1 - az^{-1}}$$

  provided that $|az^{-1}| < 1$.

The Table 4.1 gives $\mathcal{Z}$-transform table of some common signals. The computation of the $\mathcal{Z}$-transform of these functions is left as exercises for the reader.

From basic course on control system, the Laplace transform has interesting properties like linearity, homogeneity, etc. and since the $\mathcal{Z}$-transform is obtained from this transform, the properties of the $\mathcal{Z}$-transform are directly obtained:

- Linearity - The $\mathcal{Z}$-transform is a linear operator:

$$\begin{aligned} \mathcal{Z}\left[f_1(t) \pm f_2(t)\right] &= \mathcal{Z}\left[f_1(t)\right] \pm \mathcal{Z}\left[f_2(t)\right] = F_1(z) \pm F_2(z) \\ \mathcal{Z}\left[af(t)\right] &= aF(z) \end{aligned}$$

  where a is real number, $f(t)$, $f_1(t)$ and $f_2(t)$ are given functions that admit Laplace transform, $F(z)$, $F_1(z)$ and $F_2(z)$ are the $\mathcal{Z}$-transform of the functions $f^\star(t)$, $f_1^\star(t)$ and $f_2^\star(t)$ respectively.

- Initial value theorem

$$\lim_{k \to 0} f(kT) = \lim_{z \to \infty} F(z)$$

- Final value theorem

$$\lim_{k \to \infty} f(kT) = \lim_{z \to 1}(1 - z^{-1})F(z)$$

- Shift property:

$$\mathcal{Z}\left[f(t - kT)\right] = z^{-k} F(z)$$

- Back-shift property:

$$\mathcal{Z}\left[f(t + nT)\right] = z^n \left[F(z) - \sum_{k=0}^{n-1} f(kT)z^{-k}\right]$$

  then, for $k = 0, 1, 2, \cdots n$, we have:

$$\begin{aligned} \mathcal{Z}\left[f(t + T)\right] &= \mathcal{Z}\left[f[(k + 1)T]\right] = zF(z) - zf(0) \\ \mathcal{Z}\left[f(t + 2T)\right] &= \mathcal{Z}\left[f[(k + 2)T]\right] = z^2 F(z) - z^2 f(0) - zf(T) \\ \mathcal{Z}\left[f(t + nT)\right] &= \mathcal{Z}\left[f[(k + n)T]\right] = z^n F(z) - z^n f(0) \\ &\quad -z^{n-1} f(T) - \cdots - zf((n - 1)T) \end{aligned}$$

Table 4.1 Z-transform table

$F(s)$	$f(t)$ or $f(k)$	$F(z)$
1	$\delta(t)$	1
e^{-kTs}	$\delta(t-kT)$	z^{-k}
$\frac{1}{s}$	$1(t)$	$\frac{z}{z-1}$
$\frac{1}{s^2}$	t	$\frac{Tz}{(z-1)^2}$
$\frac{2}{s^3}$	t^2	$\frac{T^2z(z+1)}{(z-1)^3}$
$\frac{(k-1)!}{s^k}$	t^{k-1}	$\lim_{a\to 0}(-1)^{k-1}\frac{\partial^{k-1}}{\partial a^{k-1}}\left[\frac{z}{z-e^{-aT}}\right]$
$\frac{(k-1)!}{(s+a)^k}$	$t^k e^{-aT}$	$(-1)^k\frac{\partial^k}{\partial a^k}\left[\frac{z}{z-e^{-aT}}\right]$
$\frac{a^2}{s(s+a)^2}$	$1-e^{-at}(1+at)$	$\frac{z\left[\alpha z+\beta\right]}{(z-1)(z-e^{-aT})^2}$ $\alpha = 1-e^{-at}-aTe^{-at}$ $\beta = e^{-2at}-e^{-at}+aTe^{-at}$
$\frac{1}{s+a}$	e^{-at}	$\frac{z}{z-e^{-aT}}$
$\frac{a}{s(s+a)}$	$1-e^{-at}$	$\frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}$
$\frac{1}{(s+a)^2}$	te^{-at}	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
$\frac{1}{(s+a)(s+b)}$	$\frac{1}{b-a}\left(e^{-at}-e^{-bt}\right)$	$\frac{1}{b-a}\left[\frac{z}{z-e^{-aT}}-\frac{z}{z-e^{-bT}}\right]$
$\frac{(b-a)s}{(s+a)(s+b)}$	$be^{-bt}-ae^{-at}$	$\frac{z\left[z(b-a)-(be^{-aT}-ae^{-bT})\right]}{(z-e^{-aT})(z-e^{-bT})}$
$\frac{a}{s^2(s+a)}$	$t-\frac{1}{a}\left(1-ae^{-at}\right)$	$\frac{Tz}{(z-1)^2}-\frac{z(1-e^{-aT})}{a(z-1)(z-e^{-aT})}$
$\frac{s}{(s+a)^2}$	$(1-at)e^{-at}$	$\frac{z\left[z-e^{-aT}(1+aT)\right]}{(z-e^{-aT})^2}$
$\frac{\omega}{s^2+\omega^2}$	$\sin\omega t$	$\frac{z\sin\omega T}{z^2-2z\cos\omega T+1}$
$\frac{s}{s^2+\omega^2}$	$\cos\omega t$	$\frac{z(z-\cos\omega T)}{z^2-2z\cos\omega T+1}$
$\frac{\omega}{(s+a)^2+\omega^2}$	$e^{-at}\sin\omega t$	$\frac{ze^{-aT}\sin\omega T}{z^2-2ze^{-aT}\cos\omega T+e^{-2aT}}$
$\frac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\cos\omega t$	$\frac{z^2-ze^{-aT}\cos\omega T}{z^2-2ze^{-aT}\cos\omega T+e^{-2aT}}$
	a^k	$\frac{z}{z-a}$
	$a^k\cos k\pi$	$\frac{z}{z+a}$

Example 4.2.3 *Let us compute the $\mathscr{Z}$-transform of the ramp. This function is defined mathematically as follows:*

$$f(t) = \begin{cases} t & when\ t \geq 0 \\ 0 & otherwise \end{cases}$$

At sampling instants the function takes values as follows:

$$f(kT) = kT, k = 0, 1, 2, \cdots$$

where T is the sampling period.

Using now the definition of the $\mathscr{Z}$-transform, we get:

$$
\begin{aligned}
F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\
&= \sum_{k=0}^{\infty} kTz^{-k} \\
&= T\sum_{k=0}^{\infty} kz^{-k} \\
&= T\left(0 + z^{-1} + 2z^{-2} + \cdots + kz^{-k} + \cdots\right) \\
&= T\frac{z^{-1}}{(1 - z^{-1})^2} \\
&= \frac{Tz}{(z - 1)^2}
\end{aligned}
$$

Example 4.2.4 *Let us compute the $\mathscr{Z}$-transform of the exponential function. This function is defined mathematically as follows:*

$$
f(t) = \begin{cases} e^{-\alpha t} & when\ t \geq 0 \\ 0 & otherwise \end{cases}
$$

At sampling instants the function takes values as follows:

$$
f(kT) = e^{-k\alpha T}, k = 0, 1, 2, \cdots
$$

where T is the sampling period.
Using now the definition of the $\mathscr{Z}$-transform, we get:

$$
\begin{aligned}
F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\
&= \sum_{k=0}^{\infty} e^{-k\alpha T}z^{-k} \\
&= 1 + e^{-\alpha T}z^{-1} + e^{-2\alpha T}z^{-2} + \cdots + e^{-k\alpha T}z^{-k} + \cdots \\
&= \frac{1}{1 - e^{-\alpha T}z^{-1}} \\
&= \frac{z}{z - e^{-\alpha T}}
\end{aligned}
$$

Example 4.2.5 *Let us consider the computation of $\mathscr{Z}$-transform of the following function:*

$$
f(t) = \begin{cases} cos(wt) & for\ t \geq 0 \\ 0 & otherwise \end{cases}
$$

At the sampling instants, we have:

$$f(kT) = \begin{cases} \cos(kwT) & for \ k = 0, 1, 2, \cdots, \\ 0 & otherwise \end{cases}$$

Using the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\ &= \sum_{k=0}^{\infty} \cos(kwT)z^{-k} \end{aligned}$$

Notice that

$$\cos(kwT) = \frac{e^{jkwT} + e^{-jkwT}}{2}$$

Using this we have:

$$\begin{aligned} F(z) &= \frac{1}{2} \mathscr{Z} \left[e^{jkwT} + e^{-jkwT} \right] \\ &= \frac{1}{2} \left[\frac{1}{1 - e^{jwT}z^{-1}} + \frac{1}{1 - e^{-jwT}z^{-1}} \right] \end{aligned}$$

Using now the fact that $e^{jwT} = \cos(wT) + j\sin(wT)$, we get:

$$F(z) = \frac{z^2 - z\cos(wT)}{z^2 - 2z\cos(wT) + 1}$$

Example 4.2.6 *Let us compute the $\mathscr{Z}$-transform of a complex function. For this purpose, let us consider the following function:*

$$f(t) = \begin{cases} e^{-\alpha t}\cos(wt) + e^{-\alpha t}\sin(wt) & when \ t \geq 0 \\ 0 & otherwise \end{cases}$$

At sampling instants the function takes values as follows:

$$f(kT) = e^{-\alpha kT}\cos(wkT) + e^{-\alpha kT}\sin(wkT), k = 0, 1, 2, \cdots$$

where T is the sampling period.

Using now the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\ &= \sum_{k=0}^{\infty} \left[e^{-\alpha kT}\cos(wkT) + e^{-\alpha kT}\sin(wkT) \right] z^{-k} \end{aligned}$$

Using now the facts that:

$$\cos(kwT) = \frac{e^{jwkT} + e^{-jwkT}}{2}$$

$$\sin(kwT) = \frac{e^{jwkT} - e^{-jwkT}}{2j}$$

Using now the linearity property of the $\mathscr{Z}$-transform we get:

$$
\begin{aligned}
F(z) &= \sum_{k=0}^{\infty} e^{-\alpha kT} \cos(wkT) z^{-k} + \sum_{k=0}^{\infty} e^{-\alpha kT} \sin(wkT) z^{-k} \\
&= \frac{1}{2} \sum_{k=0}^{\infty} e^{-\alpha kT} \left[e^{jwkT} + e^{-jwkT} \right] z^{-k} + \frac{1}{2j} \sum_{k=0}^{\infty} e^{-\alpha kT} \left[e^{jwkT} - e^{-jwkT} \right] z^{-k} \\
&= \frac{z^2 - e^{-\alpha T} z \cos(wT)}{z^2 - 2e^{-\alpha T} z \cos(wT) + e^{-2\alpha T}} + \frac{e^{-\alpha T} z \sin(wT)}{z^2 - 2e^{-\alpha T} z \cos(wT) + e^{-2\alpha T}} \\
&= \frac{z^2 - e^{-\alpha T} z \cos(wT) + e^{-\alpha T} z \sin(wT)}{z^2 - 2e^{-\alpha T} z \cos(wT) + e^{-2\alpha T}} \\
&= \frac{z^2 + e^{-\alpha T} z \left[\sin(wT) - \cos(wT) \right]}{z^2 - 2e^{-\alpha T} z \cos(wT) + e^{-2\alpha T}}
\end{aligned}
$$

Previously we were able to compute the $\mathscr{Z}$-transform of a signal that may represent the output system that corresponds to a given input. Sometimes we are interested by knowing its expression in time. The inverse $\mathscr{Z}$-transform may be used for this purpose. To perform the inverse $\mathscr{Z}$-transform we can use the following methods:

- expansion into partial fraction

- polynomial division

- residues method

The inverse $\mathscr{Z}$-transform consists of finding the expression of $f(k)$ that corresponds to a given function $F(z)$. A very useful method to find the inverse transform of the function $F(z)$ is the expansion into partial fractions whose inverse transforms can be found in the table. The idea behind this method is firstly write the expression of the function in term of z^{-1}, then perform the expansion into partial fraction as usually done for the continuous-time case. This technique is illustrated by the following example.

Example 4.2.7 *Let us consider the following function $F(z)$*

$$F(z) = \frac{2z^2}{2z^2 - 3z + 1}$$

and determine the expression of $f(k)$.

To answer this question, let us first of all divide the numerator and the denominator simultaneously by $2z^2$. This gives:

$$F(z) = \frac{1}{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}$$

It is not obvious how an inverse transform looks like, but if we factorize the denominator of $F(z)$, then partial expansion gives:

$$F(z) = \frac{1}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = \frac{A}{(1 - z^{-1})} + \frac{B}{(1 - \frac{1}{2}z^{-1})}$$

As for Laplace transform, the residues are :

$$A = \lim_{z \to 1} \frac{(1 - z^{-1})}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = 2$$

$$B = \lim_{z \to \frac{1}{2}} \frac{(1 - \frac{1}{2}z^{-1})}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = -1$$

Finally, we obtain:

$$F(z) = \frac{2}{(1 - z^{-1})} + \frac{-1}{(1 - \frac{1}{2}z^{-1})}$$

and its inverse transform is

$$f(k) = \mathscr{Z}^{-1}[F(z)] = 2 - \left(\frac{1}{2}\right)^k$$

The second method that can be used to compute the inverse $\mathscr{Z}$-transform is the polynomial division method. This technique consists of performing the polynomial division of the numerator by the denominator of the function $F(z)$. To illustrate this method us consider the previous example.

Example 4.2.8 *To show how the polynomial division works, let us continue the same expression for $F(z)$ as for the previous example.*

$$F(z) = \frac{1}{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}$$

Dividing the numerator by the denominator, we obtain :

$$F(z) = 1 + \frac{3}{2}z^{-1} + \frac{7}{4}z^{-2} + \frac{15}{8}z^{-3} + \cdots$$

Since $\mathscr{Z}[\delta(t - kT)] = z^{-k}$, we then obtain:

$$f(kT) = \delta(t) + \frac{3}{2}\delta(t - T) + \frac{7}{4}\delta(t - 2T) + \frac{15}{8}\delta(t - 3T) + \cdots$$

Example 4.2.9 *In this example, we consider the following function:*

$$F(z) = \frac{0.3z}{z^2 - 1.7z + 0.7} = \frac{0.3z^{-1}}{1 - 1.7z^{-1} + 0.7z^{-2}}$$

Polynomial division gives :

$$F(z) = 0.3z^{-1} + 0.51z^{-2} + 0.657z^{-3} + \cdots$$

According to the table of $\mathscr{Z}$-transform, we have :

$$f(kT) = 0.3\delta(t - T) + 0.5\delta(t - 2T) + 0.657\delta(t - 3T) + \cdots$$

As a third method to compute the inverse $\mathscr{Z}$-transform we can use the the method of residues. It consists of using the following expression:

$$f(kT) = \text{sum of residues of } \left[z^{k-1}F(z) \right] = \sum_{n=1}^{n_s} r_n$$

n_s is the number of singularities of $F(z)$

r_n is the residue of $(z - z_n)F(z)z^{k-1}$ corresponding
to the singularity z_n
$r_n = \lim_{z \to z_n}(z - z_n)F(z)z^{k-1}$

Example 4.2.10 *Let us consider the following expression:*

$$F(z) = \frac{0.3z}{(z - 1)(z - 0.7)}$$

and compute the corresponding $f(kT)$.
   The inverse transform $f(kT)$ of the function $F(z)$ is:

$$f(kT) = \text{sum of residues of } \left[z^{k-1} \frac{0.3z}{(z - 1)(z - 0.7)} \right] \text{ at } z=1 \text{ and } z=0.7$$

$$f(kT) = \frac{0.3z}{(z - 1)(z - 0.7)} z^{k-1}(z - 1)|_{z=1} +$$

$$\frac{0.3z}{(z - 1)(z - 0.7)} z^{k-1}(z - 0.7)|_{z=0.7}$$

$$f(kT) = \frac{0.3z^k}{z - 0.7}|_{z=1} + \frac{0.3z^k}{z - 1}|_{z=0.7} = 1 - (0.7)^k$$

Example 4.2.11 *As another example of how to compute the residue for an expression that contains multiple poles, let us consider the following expression:*

$$F(z) = \frac{0.5z^2}{(z - 1)^2(z - 0.5)}$$

and compute the corresponding $f(kT)$.

The inverse transform $f(kT)$ of the function $F(z)$ is:

$$f(kT) = \text{sum of residues of } \left[z^{k-1} \frac{0.5z^2}{(z-1)^2(z-0.5)} \right] \text{ at z=1 and z=0.5}$$

$$= \frac{1}{(2-1)!} \lim_{z \to 1} \frac{d}{dz} \left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)} (z-1)^2 \right]$$

$$+ \left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)} (z-0.5) \right]\Big|_{z=0.5}$$

$$f(kT) = \left[\frac{0.5(k+1)z^k(z-0.5) - 0.5z^{k+1}}{(z-0.5)^2} z^{k-1} \right]\Big|_{z=1} +$$

$$\left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)} (z-0.5) \right]\Big|_{z=0.5}$$

$$f(kT) = \frac{0.5z^{k+1}}{z-0.5}\Big|_{z=1} + \frac{0.5z^{k+1}}{(z-1)^2}\Big|_{z=0.5} = 1 - (0.5)^k$$

In order to understand well the $\mathscr{Z}$-transform, let us see how the complex s-plane is transformed. Based on the definition of the $\mathscr{Z}$-transform, the main relationship between the s-plane and the z-plane is given by $z = e^{sT}$. This expression gives the mapping, called M of the s-plane into the z-plane. Therefore for any s in the s-plane we get the following point in the z-plane:

$$M(s) = e^{sT} = \text{corresponding value in } z$$

Let M^{-1} be the inverse transform, such as $s = \frac{1}{T} \ln z$. Usually, $s = \sigma \pm j\omega$ and $T = \frac{2\pi}{\omega_s}$ is the sampling period. Using these relations we get:

$$M(s) = M(\sigma \pm j\omega) = e^{(\sigma \pm j\omega)T} = e^{(\sigma \pm j\omega)\frac{2\pi}{\omega_s}} = e^{\frac{2\pi\sigma}{\omega_s}} \cdot e^{\pm j\frac{2\pi\omega}{\omega_s}}$$

$$M(s) = \|M\| e^{\pm j\theta}$$

$$M(s) = [\text{magnitude}] \, e^{\pm[\text{angle}]}$$

Example 4.2.12 *Let us assume that $\sigma = 0$, i.e. all the roots are on the imaginary axis in the s-plane, and let us change ω from 0 to $\frac{\omega_s}{2}$. The corresponding roots in the z-plane are given by:*

$$z = e^{j\frac{2\pi\omega}{\omega_s}}$$

Table 4.2 Poles in the z-plane using $z = e^{j\frac{2\pi\omega}{\omega_s}}$

ω	corresponding poles
0	$z_1 = 1$
$\frac{\omega_s}{8}$	$z_2 = e^{j\frac{\pi}{4}} = \cos\frac{\pi}{4} + j\sin\frac{\pi}{4} = 0.707 + j0.707$
$\frac{\omega_s}{4}$	$z_3 = e^{j\frac{\pi}{2}} = \cos\frac{\pi}{2} + j\sin\frac{\pi}{2} = j1$
$\frac{\omega_s}{2}$	$z_4 = e^{j\pi} = \cos\pi + j\sin\pi = -1$

When σ is fixed, $\sigma = 0$, and making varying ω from 0 to $\frac{\omega_s}{2}$, we notice that the corresponding variable follows a half-circle of radius 1. This is shown in Fig. 4.5.
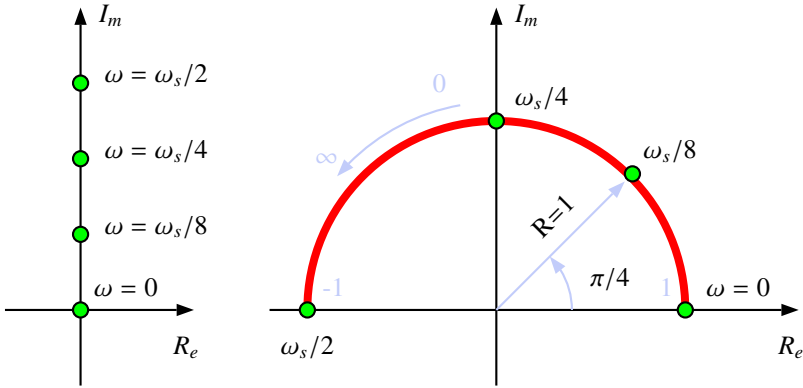
Fig. 4.5 Transformation of the s-plane into z-plane

When σ is fixed but not equal to zero, i.e.: $\sigma \neq 0$, then the radius of the circle is $R = e^{\sigma T}$. All the roots belong to the straight line $\omega = 0$ corresponding to an aperiodic response or oscillatory response. For all the other roots on the circle, the response is oscillatory.
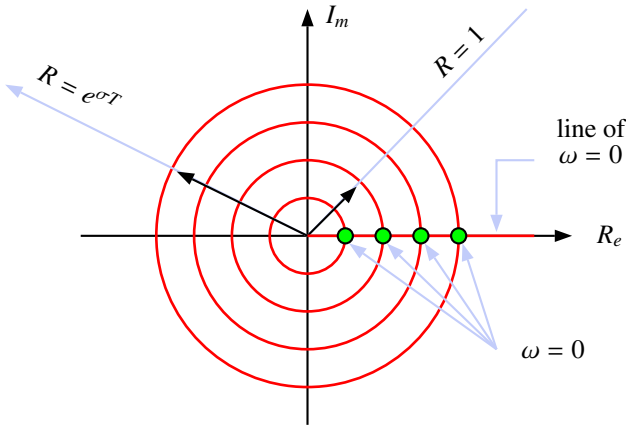
Fig. 4.6 Transformation of the s-plane when the real part is constant

For the resolution purposes, we will be interested to get the solution of a given difference equation for a fixed input. To obtain such solution we proceed as follows:

- we find the $\mathscr{Z}$-transform

- we take the inverse $\mathscr{Z}$-transform to find $y(kT)$

To illustrate this we consider the following example.

Example 4.2.13 *In this example, we consider the Fibonacci equation:*

$$y(k + 2) \ = y(k + 1) + y(k) \text{ with } y(0) = y(1) = 1$$

The $\mathscr{Z}$-transform of the Fibonacci equation is:

$$z^2 Y(z) - z^2 y(0) - zy(1) \ = zY(z) - zy(0) + Y(z)$$

that gives in turn:

$$Y(z) \ = \frac{z^2 - z}{z^2 - z - 1} y(0) + \frac{z}{z^2 - z - 1} y(1)$$

The roots of the characteristic equation are solution of the following equation:

$$z^2 - z - 1 \ = 0$$

which gives:

$$z_1 = 0.5 + \frac{\sqrt{5}}{2} \text{ and } z_2 = 0.5 - \frac{\sqrt{5}}{2}$$

Using for example the method of residues, we find:

$$y(k) \ = \frac{(1 + \sqrt{5})^{k+1} - (1 - \sqrt{5})^{k+1}}{2^{k+1} \sqrt{5}}$$

Each time, we introduce a sampling in analog operations, the transfer function should be transformed in the $\mathscr{Z}$-domain by:

$$z = e^{sT}$$

which corresponds to

$$s = \frac{1}{T} \ln z$$

The transformation $s = \frac{1}{T} \ln z$ is exact, but it is also difficult to implement in practice. That's the reason why we use two approximation methods:

- Numerical Integration

- Poles/zeros transforms

For numerical integration method care should taken when using it since we may get an unstable system after transformation. To illustrate the numerical integration approach, let us consider the following transfer function that represents a first order system:

$$G(s) \ = \frac{Y(s)}{U(s)} = \frac{a}{s + a}, a > 0$$

which gives in time domain:

$$\frac{dy(t)}{dt} + ay(t) = au(t)$$

that gives in turn:

$$\int \frac{dy(t)}{dt} = \int (-ay(t) + au(t))$$

Integrating between 2 consecutive samples, i.e. from $(k-1)T$ to kT, we obtain:

$$y(kT) - y((k-1)T) = \int_{(k-1)T}^{kT} f(t)dt$$

where $f(t) = -ay(t) + au(t)$.

In this last equation, the major problem is how to integrate the right-hand term?

- First numerical integration method: The approximation of the integral is taken equal to the one of the area shown in the Fig. 4.7.
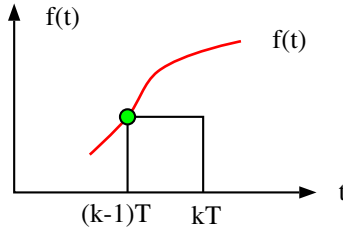
Fig. 4.7 Forward integration

Based on Fig. 4.7, we get:

$$y(kT) = y((k-1)T) + T\left[-ay((k-1)T) + au((k-1)T)\right]$$

that gives in turn:

$$y(kT) = y((k-1)T)\left[1 - aT\right] + aTu((k-1)T)$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = z^{-1}Y(z)\left[1 - aT\right] + aTz^{-1}U(z)$$

Finally we get:

$$\frac{Y(z)}{U(z)} = \frac{aTz^{-1}}{1 - z^{-1}(1 - aT)} = \frac{a}{\frac{z-1}{T} + a}$$

Now if we compare the two transfer functions (in the s-domain and in the z-domain), we conclude that the expression in z-domain is obtained from the one in the s-domain by using the following transformation:

$$s = \frac{z-1}{T}$$

- Second numerical integration method: The approximation of the integral is taken equal to the one of the area shown in the Fig. 4.8.
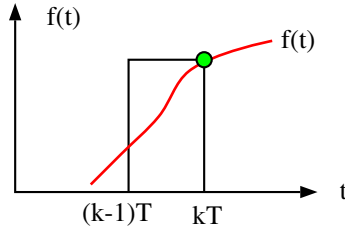
Fig. 4.8 Backward integration

Following the same steps as before and using now Fig. 4.8, we obtain:

$$y(kT) = y((k-1)T) + T\left[-ay(kT) + au(kT)\right]$$

that gives in turn in the z-domain:

$$Y(z) = z^{-1}Y(z) - aTY(z) + aTU(z)$$

From which we have:

$$\frac{Y(z)}{U(z)} = \frac{aT}{1 + aT - z^{-1}} = \frac{a}{\frac{z-1}{zT} + a}$$

Comparing again the two transfer functions as we did previously, we obtain the following transformation:

$$s = \frac{z-1}{zT}$$

- Third numerical integration method: In the two previous schemas, we have either underestimate or overestimate the area of the curve. Another alternate consists of computing the average of these two approaches. Referring now to the Fig. 4.9, we obtain the following for the approximation of the integral is that of the area shown in the figure.

$$y(kT) = y((k-1)T) + \frac{T}{2}\left[f(kT) + f((k-1)T)\right]$$
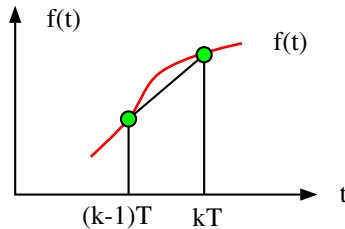
Fig. 4.9 Trapezoidal integration

From this expression we get:

$$Y(z) = z^{-1}Y(z) + \frac{T}{2}F(z) + \frac{T}{2}z^{-1}F(z)$$

Using now the expression of $F(z)$, we obtain:

$$Y(z) = z^{-1}Y(z) + \frac{T}{2}[-aY(z) + aU(z)] + \frac{T}{2}z^{-1}[-aY(z) + aU(z)]$$

that gives finally

$$\frac{Y(z)}{U(z)} = \frac{a}{\frac{2}{T}\left(\frac{z-1}{z+1}\right) + a}$$

Proceeding as before we get the following transformation:

$$s = \frac{2}{T}\left(\frac{z-1}{z+1}\right)$$

Example 4.2.14 *Consider the following transfer function:*

$$\frac{Y(s)}{U(s)} = \frac{1}{s^2 + 0.4s + 0.4} = \frac{1}{(s + 0.2 + j0.6)(s + 0.2 - j0.6)}$$

Our objective is to see the effect of the transformation we will use of the poles of the system. First of all, let us determine the sampling period. Since we have a second order, we have:

$$w_n = \sqrt{0.4} = 0.6325 rad/s$$

which gives $w_b = w_n$ and a proper choice for the sampling period is given by:

$$T = \frac{2\pi}{30w_b} = 0.33s$$

For this purpose let us compute the poles using the previous transformation for this system:

- *Using $s = \frac{z-1}{T}$, the corresponding transfer function is:*

$$G(z) = \frac{T^2}{z^2 + (-2 + 0.4T)z + 1 - 0.4T + 0.4T^2}$$

  The poles of the system in the z-plane are: $z_{1,2} = 0.9338 \pm 0.1987j$

- *Using $s = \frac{z-1}{Tz}$, the corresponding transfer function is:*

$$G(z) = \frac{T^2}{(1 + 0.4T + 0.4T^2)z^2 + (-2 - 0.4T)z + 1}$$

  The poles of the system in the z-plane are: $z_{1,2} = 0.9064 \pm 0.1689j$

- *Using $s = \frac{2}{T}\frac{z-1}{z+1}$, the corresponding transfer function is:*

$$G(z) = \frac{0.25T^2(z + 1)^2}{(1 + 0.2T + 0.1T^2)z^2 + (-2 + 0.2T^2)z + 1 - 0.2T + 0.1T^2}$$

The poles of the system in the z-plane are: $z_{1,2} = 0.9182 \pm 0.1845\,j$

- *Using the transformation* $s = \frac{1}{T} \ln z$ ($z = e^{Ts}$), *the poles are* $0.9175 \pm 0.1847\,j$.

As it can be seen from this example that the trapezoidal approximation is the more close to the exact transformation since it gives almost the same poles. The other approximations give different results. Therefore the stability and precision should be tested before choosing a particular method.

As another approach that can be used to approximate the transfer function in $\mathscr{Z}$- domain is what it is always referred in the literature to as the poles/zeros transformation. It consists of doing the following steps:

- make all the poles of $G(s)$ correspond to $z = e^{-sT}$. That is, if $s = -a$, is a pole in the s-domain, then $G(z)$ will have a pole in the z-domain at $z = e^{-aT}$

- do the same thing for the zeros of $G(s)$

- place all the poles of $G(s)$ corresponding to $s = \infty$ at $z = -1$. This means adding $(z + 1)$, $(z + 1)^2$, $\cdots$ to the numerator of $G(z)$ such that the degree of the numerator will be equal to the one of the denominator.

- make the gain of $G(s)$ correspond to the one of $G(z)$. This means that we must do the following for that:

$$[G(s)]_{s=0} = [G(z)]_{z=1}$$

Example 4.2.15 *To show how this procedure works, let us consider the following transfer function:*

$$G(s) = \frac{10}{(s + 1)(s + 2)}.$$

The poles of this transfer function are $s_1 = -1$ *and* $s_2 = -2$. *Their corresponding poles are respectively* $z_1 = e^{-T}$ *and* $z_2 = e^{-2T}$. *If we fix the sampling period to* $T = 0.02s$, *then these poles becomes* $z_1 = 0.9802$ *and* $z_2 = 0.9608$.

Since the denominator is of degree 2, *then the numerator also should be of degree* 2. *To do that, we add to the numerator the term* $(z + 1)^2$.

The gain is then calculated by:

$$\left[\frac{10}{(s + 1)(s + 2)}\right]_{s=0} = \left[K\frac{(z + 1)^2}{(z - 0.9802)(z - 0.9608)}\right]_{z=1}$$

$$1 = K\frac{4}{(0.285)(0.487)}$$

which gives:

$$K = 0.0019$$

Finally the transfer function in the $\mathscr{Z}$-*domain is given by:*

$$G(z) = \frac{0.0019(z + 1)^2}{(z - 0.9802)(z - 0.9608)}$$

As another approach, it is possible to derive $G(z)$ from $G(s) = \frac{N(s)}{D(s)}$ when $D(s)$ has distinct roots. This can be computed using the following formula:

$$G(z) = \sum_{n=1}^{p} \frac{N(x_n)}{D'(x_n)} \frac{z}{z - e^{x_n T}}$$

with $D'(x_n) = \frac{\partial D}{\partial s}|_{s=x_n}$ for $n = 1, 2, 3, \cdots, p$

Example 4.2.16 *To show the idea how to get a $G(z)$ from a $G(s)$ with a denominator that has distinct roots, let us consider the following transfer function:*

$$G(s) = \frac{1}{(s + a)(s + b)} = \frac{1}{s^2 + (a + b)s + ab}$$

The denominator and the numerator of this transfer function are given by:

$$D(s) = (s + a)(s + b)$$
$$N(s) = 1$$

The denominator derivative with respect to s is given by:

$$D'(s) = 2s + (a + b)$$

The values of the derivatives at the two roots are:

$$D'(x_1 = -a) = b - a$$
$$D'(x_2 = -b) = -(b - a)$$

Using this and the previous formula, we get:

$$G(z) = \frac{1}{b - a} \frac{z}{z - e^{-aT}} - \frac{1}{b - a} \frac{z}{z - e^{-bT}} = \left(\frac{1}{b - a}\right)\left[\frac{z}{z - e^{-aT}} - \frac{z}{z - e^{-bT}}\right]$$

4.3   Transfer Function Concept

The concept of transfer function for sampled systems can be defined similarly as it has been done for continuous-time one. To clarify this, let us refer to the Fig. 4.10 where the upstream sampler is a real one while the downstream one is a fictitious that we assume to be ideals and synchronized at the same sampling period. The second sampler is introduced for the purpose to define $Y(z)$ and therefore define properly the pulse transfer function. Based on the Fig. 4.10, we get:

$$Y(s) = G(s)U^{\star}(s)$$

Since the output is sampled by the fictitious sampler, we can then have:

$$Y^{\star}(s) = \left[G(s)U^{\star}(s)\right]^{\star}$$
$$= G^{\star}(s)U^{\star}(s)$$
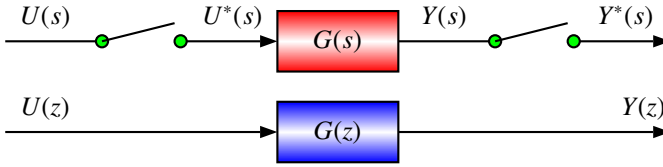
Fig. 4.10 Pulse transfer function definition

and if we apply the $\mathscr{Z}$-transform, we obtain:

$$Y(z) \ = \ G(z)U(z)$$

where $Y(z) = \mathscr{Z}\left[Y^{\star}(s)\right]$ and $U(z) = \mathscr{Z}\left[U^{\star}(s)\right]$.

This relation can be proved in an elegant way starting from the time domain. In fact, we have:

$$y(t) \ = \ \mathscr{L}^{-1}\left[G(s)U^{\star}(s)\right]$$

Using now the convolution theorem, we get:

$$y(t) \ = \ \int_{0}^{t} g(t-\sigma)u^{\star}(\sigma)d\sigma$$

From the other side we know that $u^{\star}(\sigma)$ can be written as follows:

$$u^{\star}(\sigma) \ = \ \sum_{k=0}^{\infty} u(kT)\delta(t-kT)$$

Using this, the expression of $y(t)$ becomes:

$$y(t) \ = \ \int_{0}^{t} g(t-\sigma)\sum_{k=0}^{\infty} u(kT)\delta(t-kT)d\sigma$$

$$= \ \sum_{k=0}^{\infty}\int_{0}^{t} g(t-\sigma)u(kT)\delta(t-kT)d\sigma$$

$$= \ \sum_{k=0}^{\infty} g(t-kT)u(kT)$$

Using now the definition of the $\mathscr{Z}$-transform of the sampled signal $y^{\star}(t)$ we have:

$$Y(z) \ = \ \sum_{k=0}^{\infty} y(kT)z^{-k}$$

$$= \ \sum_{k=0}^{\infty}\left[\sum_{l=0}^{\infty} g(kT-lT)u(lT)\right]z^{-k}$$

Performing the change of variable, $m = k - l$, we get:

$$Y(z) = \sum_{m=0}^{\infty} \sum_{l=0}^{\infty} [g(mT)u(lT)] z^{-m-l}$$

that can be rewritten as follows:

$$Y(z) = \left[\sum_{m=0}^{\infty} g(mT)z^{-m} \right]\left[\sum_{l=0}^{\infty} u(lT)z^{-l} \right]$$

$$= G(z)U(z)$$

Finally, the transfer function is given by:

$$G(z) = \frac{Y(z)}{U(z)}$$

which is the ratio between the $\mathscr{Z}$-transform of the output and $\mathscr{Z}$-transform of the input .

When manipulating the block diagrams of sampled systems, care should be taken. The following relations will help for this purpose.

- If $Y(s) = G(s)U(s)$, then

$$Y(z) = \mathscr{Z}[Y^*(s)] = \mathscr{Z}[[G(s)U(s)]^*] \neq \mathscr{Z}[G^*(s)U^*(s)] = G(z)U(z).$$

- If $Y(s) = G(s)U^*(s)$, then

$$Y(z) = \mathscr{Z}[Y^*(s)] = \mathscr{Z}[[G(s)U^*(s)]^*] = \mathscr{Z}[G^*(s)U^*(s)] = G(z)U(z).$$

Example 4.3.1 *In this example we consider the system of the Fig. 4.11 that represents two systems in serial with an ideal sampler between. The expression of the two transfer functions are:*

$$G_1(s) = \frac{1}{s+a}$$

$$G_2(s) = \frac{a}{s(s+a)}$$

Our goal is to compute the equivalent transfer function for this system.
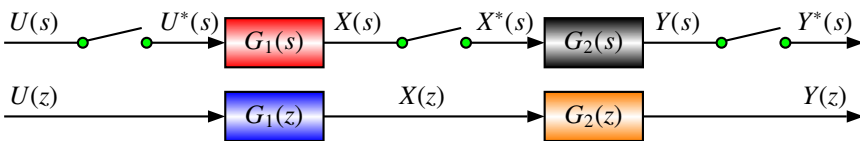
Fig. 4.11 Cascade transfer functions with sampler between

Based on this figure, we get:

$$Y^*(s) = G_2^*(s)X^*(s)$$
$$X^*(s) = G_1^*(s)U^*(s)$$

which gives:

$$Y^*(s) = G_2^*(s)G_1^*(s)U^*(s)$$

that implies in turn:

$$\frac{Y(z)}{U(z)} = G_1(z)G_2(z)$$

Using the table of $\mathscr{Z}$-transform, we have:

$$\frac{Y(z)}{U(z)} = G_1(z)G_2(z) = \frac{z}{(z - e^{-aT})} \frac{z(1 - e^{-aT})}{(z - 1)(z - e^{-aT})}$$
$$= \frac{z^2(1 - e^{-aT})}{(z - 1)(z - e^{-aT})^2}$$

Example 4.3.2 *In this example we consider the situation where the sample is removed between the two transfer function in serial. This situation is illustrated by the Fig. 4.12. The transfer function $G_1(s)$ and $G_2(s)$ are given by the following expression:*

$$G_1(s) = \frac{a}{s + a}$$
$$G_2(s) = \frac{a}{s(s + a)}$$

where a is a positive scalar.

Our goal is to compute the equivalent transfer function and compare it with the one obtained in the previous example.
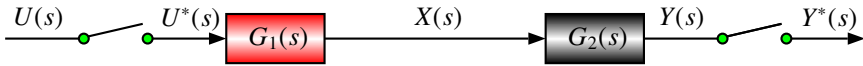
Fig. 4.12 Cascade transfer functions without sampler between

In this case we have:

$$\frac{Y^*(s)}{U^*(s)} = [G_1(s)G_2(s)]^*$$

that gives in turn

$$\frac{Y^*(s)}{U^*(s)} = \mathscr{Z}[G_1(s)G_2(s)] = G_1G_2(z)$$

It is important to notice that the equivalent transfer function we obtain for this case is different from the one we obtained for the system of the previous example.

Using the expression of $G_1(s)$ and $G_2(s)$, we get:

$$G_1(s)G_2(s) = \frac{a^2}{s(s+a)^2}$$

Based on the table of $\mathscr{Z}$-transform, we have:

$$\mathscr{Z}[G_1(s)G_2(s)] = G_1G_2(z) = \frac{z}{z-1} - \frac{z}{z-e^{-aT}} \frac{zaTe^{-aT}}{(z-e^{-aT})^2}$$

Example 4.3.3 *In this example we consider the case where we have transfer functions in feedback and we search to compute the equivalent one as we did in the previous examples. The system is illustrated by the Fig. 4.13. The transfer functions are given by the following expression:*

$$G(s) = \frac{a}{s(s+a)}$$

$$H(s) = \frac{1}{s}$$
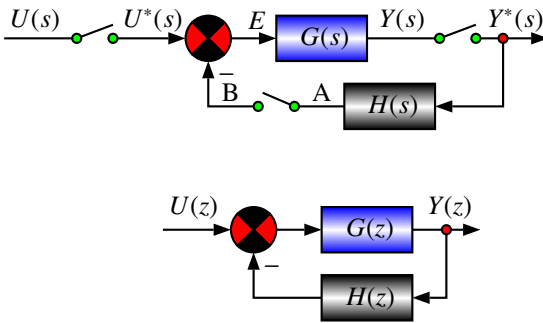
Fig. 4.13 Transfer functions in feedback

Based on this figure we have:

$$A = H(s)Y^*(s)$$
$$B = [A]^* = [H(s)Y^*(s)]^* = H^*(s)Y^*(s)$$
$$E = U^*(s) - B = U^*(s) - H^*(s)Y^*(s)$$
$$Y(s) = G(s)[U^*(s) - H^*(s)Y^*(s)]$$

that gives in turn:

$$Y^*(s) = [Y(s)]^* = [G(s)[U^*(s) - H^*(s)Y^*(s)]]^* = G^*(s)[U^*(s) - H^*(s)Y^*(s)]$$

From this we get:

$$\frac{Y^*(s)}{U^*(s)} = \frac{G^*(s)}{1 + G^*(s)H^*(s)}$$

That gives the following pulse transfer function:

$$\frac{Y(z)}{U(z)} = \frac{G(z)}{1 + G(z)H(z)}$$

From the table of $\mathscr{Z}$-transform we get:

$$G(z) = \frac{z(1 - e^{-at})}{(z - 1)(z - e^{-aT})}$$

$$H(z) = \frac{z}{z - 1}$$

Using this we obtain:

$$\frac{Y(z)}{U(z)} = \frac{G(z)}{1 + G(z)H(z)} = \frac{\frac{z(1-e^{-at})}{(z-1)(z-e^{-aT})}}{1 + \frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}\frac{z}{(z-1)}}$$

$$= \frac{(1 - e^{-aT})z(z - 1)(z - e^{-aT})}{(z - 1)^2(z - e^{-aT}) + z^2(1 - e^{-aT})}$$

Example 4.3.4 *As a second example of the previous case let us consider the system of the Fig. 4.14. The question is how to compute the pulse transfer function $F(z) = \frac{Y(z)}{G(z)}$ of this system.*
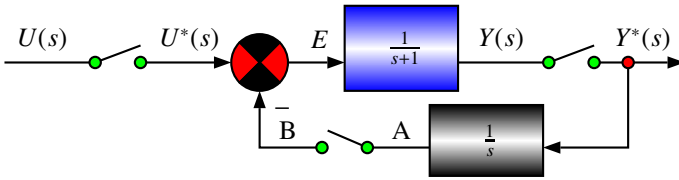
Fig. 4.14 Transfer functions in feedback

Since (see the table for $\mathscr{Z}$-transform)

$$G(z) = \mathscr{Z}\left[\frac{1}{s + 1}\right] = \frac{z}{z - e^{-T}} \quad \text{and } H(z) = \mathscr{Z}\left[\frac{1}{s}\right] = \frac{z}{z - 1}$$

we get the following expression for the closed-loop pulse transfer function:

$$\frac{Y(z)}{U(z)} = \frac{G(z)}{1 + G(z)H(z)} = \frac{z(z - 1)}{(z - e^{-T})(z - 1) + z^2}$$

Example 4.3.5 *In this example the system represented by the Fig. 4.15 where a zero order hold (ZOH) is used.*

1. *Find the open loop and closed loop pulse transfer functions $\frac{Y(z)}{U(z)}$*

2. *Find the unit-step response if $K = 1$ for $T = 0.1$*
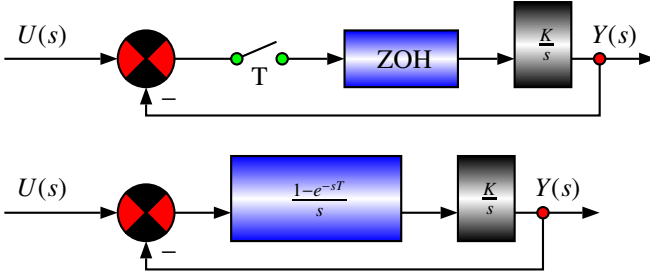
Fig. 4.15 Transfer functions in feedback

The solution of this example can be obtained easily. In fact we have:

- *Open loop:*

$$\frac{Y(s)}{U(s)} = \frac{K}{s^2}(1 - e^{-sT})$$

From which we have:

$$\frac{Y(z)}{U(z)} = \frac{KTz}{(z-1)^2}\frac{z-1}{z} = \frac{KT}{z-1}$$

Finally we obtain:

$$Y(z) = \frac{KT}{z-1}U(z)$$

- *Closed loop:*

$$Y(z) = \frac{KT/(z-1)}{1 + \frac{KT}{z-1}}U(z) = \frac{KT}{z-(1-KT)}\frac{z}{z-1}$$

Using the method of residues for $z_1 = 1$ and $z_2 = 1 - KT$, and the fact that $K = 1$, we find:

$$y(kT) = 1 - (1 - T)^k \quad pour \ k = 0, 1, 2, 3, \cdots$$

If we use $T = 0.1s$, we get:

$$y(k) = 1 - 0.9^k$$

Example 4.3.6 *Let us consider the system of the Fig. 4.16 and compute the transfer function.*
  Using this figure, we have:

$$E(s) = R(s) - H(s)Y(s)$$
$$Y(s) = G(s)E^\star(s)$$

which gives in turn:

$$E^\star(s) = R^\star(s) - [H(s)Y(s)]^\star$$
$$Y^\star(s) = G^\star(s)E^\star(s)$$
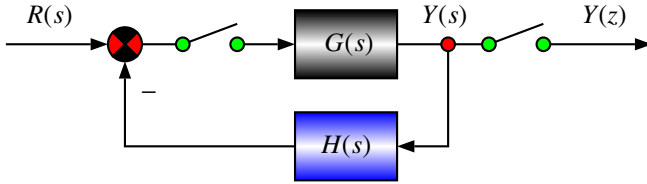
Fig. 4.16 Transfer functions in feedback

Using the $\mathscr{Z}$-transform, we obtain:

$$Y(z) \;=\; \frac{G(z)R(z)}{1 + GH(z)}$$

Example 4.3.7 *Let us consider the system of the Fig. 4.17 and compute the transfer function.*
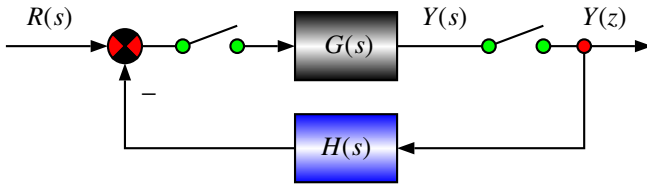
Fig. 4.17 Transfer functions in feedback

Using this figure, we have:

$$E(s) \;=\; R(s) - H(s)Y^{\star}(s)$$
$$Y(s) \;=\; G(s)E^{\star}(s)$$

which gives in turn:

$$E^{\star}(s) \;=\; R^{\star}(s) - H^{\star}(s)Y^{\star}(s)$$
$$Y^{\star}(s) \;=\; G^{\star}(s)E^{\star}(s)$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) \;=\; \frac{G(z)R(z)}{1 + G(z)H(z)}$$

Example 4.3.8 *Let us consider the dynamical system of the block diagram illustrated by Fig. 4.18*

$$Y(z) \;=\; \frac{RG(z)}{1 + HG(z)}$$
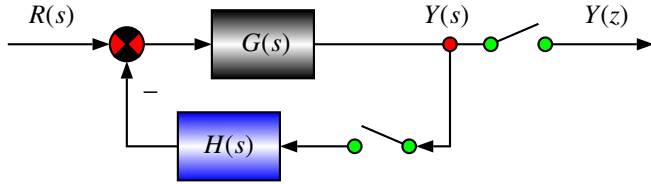
Fig. 4.18 Transfer functions in feedback
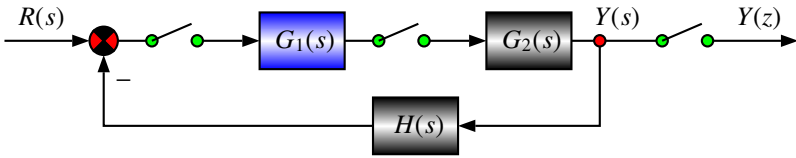
Fig. 4.19 Transfer functions in feedback

Example 4.3.9 *Let us consider the system of the block diagram of the figure 4.19 and compute the transfer function.*

$$Y(z) = \frac{G_2(z)RG_1(z)}{1 + G_1G_2H(z)}$$

Example 4.3.10 *Let us consider the system of the block diagram of the figure 4.20 and compute the transfer function.*
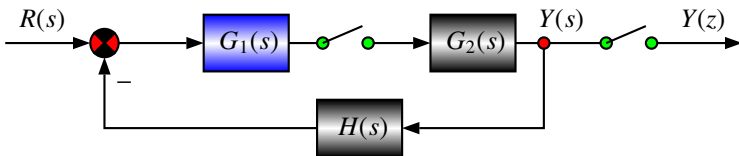
Fig. 4.20 Transfer functions in feedback

Using this figure, we have:

$$E(s) = R(s) - H(s)Y(s)$$
$$Y(s) = [G(s)E(s)]^{\star}$$

which gives in turn:

$$Y^\star(s) = \left[\left[(R(s) - H(s)Y(s))\,G_1(s)\right]^\star\right]^\star G_2(s)$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = \frac{G_1(z)G_2(z)R(z)}{1 + G_1(z)G_2(z)H(z)}$$

Based on these examples, we are always able to compute the transfer function of the system and its expression is given by:

$$G(z) = \frac{Y(z)}{U(z)}$$

where $Y(z)$ and $U(z)$ are respectively the $\mathscr{Z}$-transform of the output $Y(s)$ and the input $U(s)$.

This transfer function is always in the following form:

$$G(z) = \frac{N(z)}{D(z)}$$
$$= \frac{b_n z^n + b_{n-1}z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0}$$

where a_i and b_i are real scalars and n is an integer representing the degree of the system.

The roots of the polynomials $N(z)$ and $D(z)$, i.e.: the solutions of the following equations:

$$N(z) = 0$$
$$D(z) = 0$$

are called respectively zeros and poles of the system.

The poles play an important role in the system response. Their location is very important and it related to the system performances like the stability, the transient regime, etc. as it will be shown later on.

Example 4.3.11 *Let us consider a dynamical system with the following transfer function:*

$$G(z) = \frac{N(z)}{D(z)}$$
$$= \frac{z^2 - z + 0.02}{z^3 - 2.4z^2 + z - 0.4}$$

Compute the poles and zeros of the system and plot them in the z-domain.
From the expression of the transfer function we have:

$$N(z) = z^2 - z + 0.02$$
$$D(z) = z^3 - 2.4z^2 + z - 0.4 = (z - 2)\left(z^2 - 0.4z + 0.2\right)$$

The roots of this polynomials are $0.1 \pm 0.1j$ *for the zeros and* 2 *and* $0.2 \pm 0.4j$ *for the poles. The zeros are all inside the unit circle. The complex poles are also inside the unit circle while the real one is outside this circle.*

We have introduced the concept of transfer function and we have learnt how to manipulate the block diagrams. It is now time to compute the time response of the system for given signal inputs. This is the subject of the next section.

4.4   Time Response and Its Computation

More often, the control system has to guarantee certain performances such as:

- the settling time at a given percentage

- the overshoot

- the damping ratio

- etc.

For time definitions we ask the reader to look to the Fig. 4.21. To have an idea on the concept of the settling time, the overshoot, etc., let us consider a linear time invariant system with an input $r(t)$ and an output $y(t)$. If we apply a step function at the input, the output of this system will be as shown in Fig. 4.21. From this figure, it can be seen that the settling time is defined as the time for the system response, $y(t)$ to reach the error band (that is defined with a certain percentage, 2 %, 5 %, etc.) and stay for the rest of the time. The lower the percentage is, the longer the settling time will be.

The overshoot is another characteristic of the time response of a given system. If we refer to the previous figure, the overshoot is defined as the maximum exceed of the steady state value of the system output. More often, we use the percentage overshoot, which is defined as the maximum value of the output minus the step value divided by the step value.

The error is also another characteristic of the output behavior. It is defined as the difference between the steady value taken by the output and the desired value. For a closed-loop system with a unity feedback, the error, $E(z)$, is defined mathematically as:

$$E(z) = R(z) - Y(z)$$

where $R(z)$ is the reference input and $Y(z)$ is the output.

Previously we developed tools that can be used to compute the expression in time of a given signal. Here we will use this to compute the time response of a given system to a chosen input that may be one or a combination of the following signals:

- Dirac impulse

- step

- ramp
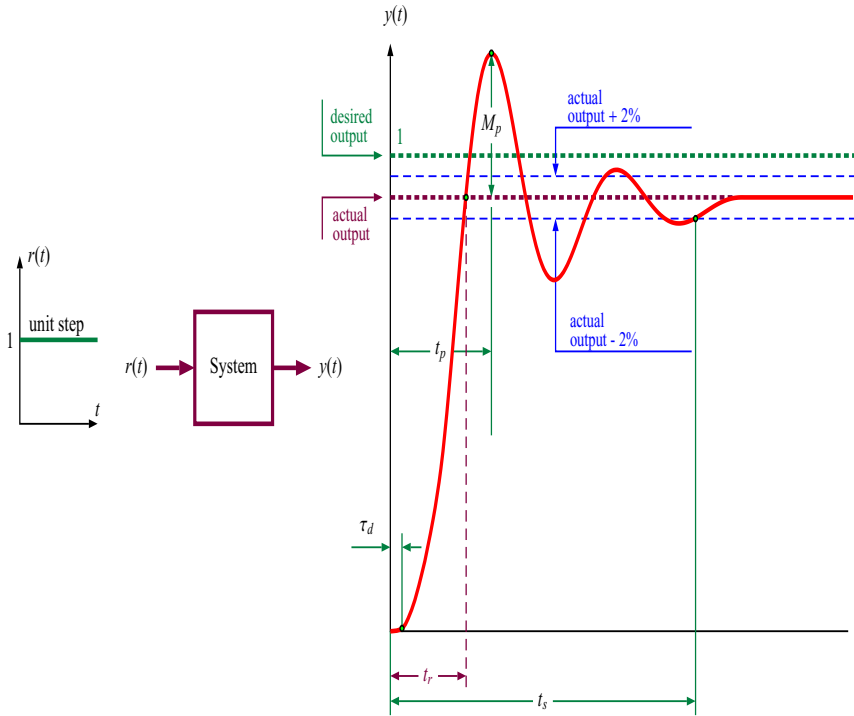
Fig. 4.21 Behavior of the time response for a step input

To compute the time response let us consider a system which has a pulse transfer function $G(z)$ with a given input signal, $U(z)$, and consider the computation of the expression of $y(kT)$. The system is represented in Fig. 4.22. This figure may represent either an open loop pulse transfer function or its equivalent closed-loop pulse transfer function that we get after simplifying the system block diagram.

From this figure, we get:

$$Y(z) = G(z)U(z)$$

The computation of time response, $y(kT)$, is brought to the computation of the inverse $\mathscr{Z}$-transform that can be determined using one of the following methods:
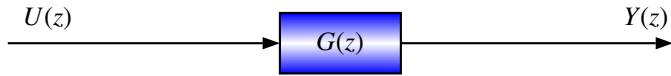
Fig. 4.22 Block diagram (BD)

- expansion into partial fraction
- polynomial division
- residues method

To illustrate how the time response, let us consider the following examples.

Example 4.4.1 *In this example we consider the speed control of a dc motor driving via a gear a given mechanical load. We assume that the system is controlled using a microcontroller. The transfer function of the system is given by:*

$$G(s) = \frac{K}{\tau s + 1}$$

with $K = 2$ and $\tau = 2$.

The system is considered in open-loop. In this case since we have the presence of a ZOH, we obtain:

$$G(z) = \left(1 - z^{-1}\right) \mathscr{Z} \left[\frac{2}{s(2s + 1)}\right]$$

Using the $\mathscr{Z}$-transform table, we get:

$$\begin{aligned} G(z) &= \left(1 - z^{-1}\right) \left[\frac{z\left(1 - e^{-\frac{T}{2}}\right)}{(z - 1)\left(z - e^{-\frac{T}{2}}\right)}\right] \\ &= \frac{\left(1 - e^{-\frac{T}{2}}\right)}{\left(z - e^{-\frac{T}{2}}\right)} \end{aligned}$$

where T is the sampling period.

For our system, since the time constant is equal to 2sec, a proper choice for the sampling period is $T = 0.2sec$. Using this, we get:

$$G(z) = \frac{0.0952}{z - 0.9048}$$

If now we consider that the signal input is unit step, we get

$$Y(z) = \frac{0.0952z}{(z - 1)(z - 0.9048)}$$

To compute the time response either we can use the table or proceed with the expansion into partial fraction.

Using the $\mathscr{L}$-transform table, we have:

$$y(kT) = 1 - e^{-0.1k}$$

With the expansion into partial fraction we obtain:

$$\frac{Y(z)}{z} = \frac{0.0952}{(z-1)(z-0.9048)}$$

$$= \frac{K_1}{z-1} + \frac{K_2}{z-0.9048}$$

$$= \frac{1}{z-1} + \frac{-1}{z-0.9048}$$

From this we get:

$$Y(z) = \frac{z}{z-1} + \frac{-z}{z-0.9048}$$

Using now the $\mathscr{L}$-transform table, we get:

$$y(kT) = 1 - e^{-0.1k}$$

since $e^{-0.1} = 0.9048$.

Example 4.4.2 *In this example we consider the position control of a dc motor driving via a gear a given mechanical load. We assume that the system is controlled using a microcontroller. The transfer function of the system is given by:*

$$G(s) = \frac{K}{s(\tau s + 1)}$$

with $K = 2$ and $\tau = 2$.

The system is considered in open-loop. In this case since we have the presence of a ZOH, we obtain:

$$G(z) = \left(1 - z^{-1}\right) \mathscr{L}\left[\frac{2}{s^2(2s+1)}\right]$$

Using the $\mathscr{L}$-transform table with $T = 0.2$ second, we get:

$$G(z) = \left(1 - z^{-1}\right)\left[\frac{Tz}{(z-1)^2} - \frac{z\left(1 - e^{-\frac{T}{2}}\right)}{0.5(z-1)\left(z - e^{-\frac{T}{2}}\right)}\right]$$

$$= \frac{(0.4048z - 0.5476)}{0.5(z-1)(z-0.9048)}$$

If now we consider that the signal input is unit step, we get

$$Y(z) = 0.8096\frac{z(z-1.3528)}{(z-1)^2(z-0.9048)}$$

To compute the time response either we can use the $\mathscr{L}$-transform table or proceed with the method of expansion into partial fraction or with the method of residues.

Using the $\mathscr{Z}$-transform table, we get:

$$y(kT) = kT - \frac{1}{a}\left[1 - e^{akT}\right]$$

with $a = 0.5$ and $T = 0.2$

With the method expansion into partial fraction we have:

$$Y(z) = \frac{K_1}{(z-1)^2} + \frac{K_2}{(z-1)} + \frac{K_3}{(z-0.9048)}$$

With the method of residues, we obtain:

$$y(kT) = \sum residues\ of\ 0.8096\frac{z\,(z-1.3528)\,z^{k-1}}{(z-1)^2\,(z-0.9048)}$$

at the poles $z = 1$ and $z = 0.9048$.

These residues are computed as follows:

- *residue at pole $z = 1$*

$$\frac{1}{(2-1)!}\frac{d}{dz}\left[(z-1)^2 0.8096\frac{z\,(z-1.3528)\,z^{k-1}}{(z-1)^2\,(z-0.9048)}\right]_{|z=1}$$

$$= \frac{d}{dz}\left[0.8096\frac{(z-1.3528)\,z^k}{(z-0.9048)}\right]_{|z=1}$$

$$= 119.0464 - 2.9568k$$

- *residue at pole $z = 0.9048$*

$$\left[0.8096\frac{(z-1.3528)\,z^k}{(z-1)^2}\right]_{|z_r=0.9048}$$

$$= -40.0198\,(0.9048)^k$$

Using now the table we get:

$$y(kT) = 1 - e^{-0.1k}$$

since $e^{-0.1} = 0.9048$.

From the time response we computed in the previous section, it can be seen that for a given system the output can take either finite or infinite value for a given signal signal input. The question is why this happen. The answer of this question is given by the stability analysis and this will be covered in the next section.

4.5   Stability and Steady-State Error

For systems in the continuous-time domain, the stability implies that all the poles must have negative real parts. With the transform $z = e^{Ts}$, with T is the sampling period, we saw that the left half plane of the s-domain corresponds to the inside unit

circle and therefore, in the z-domain, the system will be stable if all the poles are inside this unit circle.

To analyze the stability of discrete-time systems, let us consider the system of the Fig. 4.23. The closed loop transfer function of this system is given by:

$$F(z) = \frac{Y(z)}{R(z)} = \frac{C(z)G(z)}{1 + C(z)G(z)}$$

where $R(z)$ and $Y(z)$ are respectively the input and the output.

The poles of the system are the solution of the following characteristic equation:

$$1 + C(z)G(z) = 0$$

The study of stability requires the computation of these roots. For small order system we can always solve the characteristic equation by hand and then obtain the poles and the conclusion on stability will be done based on the fact where the poles are located. For high order this approach is not recommended and an alternate is needed. Some criterions have been developed to study the stability. Among these criterions we quote the one of Jury and the one of Raible.
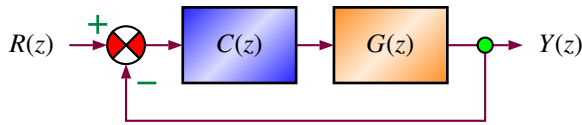
Fig. 4.23 Block diagram of the closed-loop

Let $z = e^{sT}$ with $s = \sigma \pm j\omega$. Therefore,

if $\sigma < 0$ then $|z| < 1$ and the system is stable
if $\sigma > 0$ then $|z| > 1$ and the system is unstable
if $\sigma = 0$ then $|z| = 1$ and the system is at the limit of stability

Example 4.5.1 *Let us consider a dynamical system with the following characteristic equation:*

$$1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2} = 0$$

The roots of the characteristic equation are: $z = \frac{1}{2}$ and $z = \frac{1}{4}$. These roots are located inside the unit circle and therefore the system is stable.

Example 4.5.2 *Let us consider a dynamical system with the following characteristic equation:*

$$1 - 2z^{-1} + \frac{5}{4}z^{-2} = 0$$

or equivalently:

$$z^2 - 2z + \frac{5}{4} = 0$$

The roots of the system are $z_{1,2} = 1 \pm j\frac{1}{2}$ and are both outside the unit circle which implies that the system is unstable.

A direct approach to study the stability of discrete-time system is to convert it to an equivalent continuous-time one, and then use the Routh-Hurwitz's Criterion. The idea is to find an adequate application that maps the inside of the unit circle onto the left-hand half plane. Then, we can apply the Routh-Hurwitz criterion. The transformation we're looking for is:

$$z = \frac{1 + w}{1 - w} \text{ with } w \neq 1$$

Replacing z by this expression in the characteristic equation will give a new one in w and we can apply the Routh-Hurwitz's Criterion.

Example 4.5.3 *To show how we use the Routh-Hurwitz's Criterion, let us consider the dynamical system with the following characteristic equation:*

$$z^3 - 2.4z^2 + z - 0.4 = 0$$

It can be shown that the poles are 2 and $0.2 \pm 0.4j$. Therefore the system is unstable.

Let us now replace z by $\frac{1+w}{1-w}$ in the characteristic equation. This gives:

$$\left[\frac{1 + w}{1 - w}\right]^3 - 2.4\left[\frac{1 + w}{1 - w}\right]^2 + \left[\frac{1 + w}{1 - w}\right] - 0.4 = 0$$

which can be put in the following form:

$$4.8w^3 + 3.2w^2 + 0.8w - 0.8 = 0$$

The Routh-Hurwitz's Criterion consists then of filling the following table:

w^3	4.8	0.8	0
w^2	3.2	−0.8	0
w^1	2	0	
w^0	−0.8		

Based on the first column, we can see that there one change in the sign and therefore the system is unstable. This confirm the results we has already remarked earlier.

It is also important to notice that the roots of the characteristic equation in w are given by:

$$w_1 = 0.3333$$
$$w_{2,3} = -0.5000 \pm 0.5000j$$

These roots can also be obtained from the ones in z-domain using $w = \frac{z-1}{z+1}$.

Example 4.5.4 *Consider the characteristic equation:*

$$z^2 + z(6.32K - 1.368) + 0.368 = 0$$

Applying the bilinear transform yields:

$$\left(\frac{1+w}{1-w}\right)^2 + \left(\frac{1+w}{1-w}\right)(6.32K - 1.368) + 0.368 = 0$$

that gives in turn:

$$w^2[2.736 - 6.32K] + 1.264w + (6.32K - 1) = 0$$

Applying Routh-Hurwitz gives:

w^2	$2.736 - 6.32K$	$6.32K - 1$
w^1	1.264	0
w^0	$6.32K - 1$	

To guarantee the stability we should determine the range of the parameter K such that we don't have sign change in the first column. For the row w^0, we should have $6.32K - 1 > 0$, i.e. $K > \frac{1}{6.32} = 0.158$. For the row w^2, we should also have $2.736 - 6.32K > 0$, i.e. $K < \frac{2.736}{6.32} = 0.4329$. If we look to these two conditions, we conclude that the system is stable for $0.158 < K < 0.4349$.

To check this, let us consider $K = 0.2$, which is inside the interval. Using this value, we obtain the following characteristic equation:

$$z^2 - 0.104z + 0.368 = 0$$

that has as roots $z_1 = 0.052 + j0.6044$ and $z_2 = 0.052 - j0.6044$. The roots are located inside the unit circle and therefore, the system is then stable. For $K = 1$, we obtain:

$$z^2 + 4.952z + 0.368 = 0$$

The roots are $z_1 = -0.076$ and $z_2 = -4.876$. The system is then unstable because $|z_2| > 1$.

For discrete-time Jury has developed a criterion that gives an idea on stability of any system without solving the characteristic equation. To show how this approach works, let us consider the following characteristic polynomial with real coefficients:

$$P(z) = a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0 = 0$$

where $a_n > 0$ and a_i is a real scalar.

Jury's stability criterion consists of building the following array of coefficients:

row 1	a_0	a_1	a_2	$\cdots$ a_{n-k}	$\cdots$	a_{n-1}	a_n
row 2	a_n	a_{n-1}	a_{n-2}	$\cdots$ a_k	$\cdots$	a_1	a_0
row 3	b_0	b_1	b_2	$\cdots$ b_{n-k}	$\cdots$	b_{n-1}	
row 4	b_{n-1}	b_{n-2}	b_{n-3}	$\cdots$ b_k	$\cdots$	b_0	
row 5	c_0	c_1	c_2	$\cdots$	c_{n-2}		
row 6	c_{n-2}	c_{n-3}	c_{n-4}	$\cdots$	c_0		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$			
row 2n-5	p_0	p_1	p_2	p_3			
row 2n-4	p_3	p_2	p_1	p_0			
row 2n-3	q_0	q_1	q_2				

The Jury's array coefficients are computed as follows:

$$b_k = \begin{vmatrix} a_0 & a_{n-k} \\ a_n & a_k \end{vmatrix}, \qquad c_k = \begin{vmatrix} b_0 & b_{n-1-k} \\ b_{n-1} & b_k \end{vmatrix},$$

$$d_k = \begin{vmatrix} c_0 & c_{n-2-k} \\ c_{n-2} & c_k \end{vmatrix}, \qquad \cdots$$

$$q_0 = \begin{vmatrix} p_0 & p_3 \\ p_3 & p_0 \end{vmatrix}, \qquad q_2 = \begin{vmatrix} p_0 & p_1 \\ p_3 & p_2 \end{vmatrix}$$

The necessary and sufficient conditions that the system described by P(z) is stable are:

$$P(1) > 0$$

$$P(-1) \quad \begin{cases} > 0 & \text{if n is even} \\ < 0 & \text{if n is odd} \end{cases}$$

with (n - 1) constraints
$$\begin{bmatrix} |a_0| < a_n & |b_0| > |b_{n-1}| \\ |c_0| > |c_{n-2}| & |d_0| > |d_{n-3}| \\ \cdots & \cdots \\ |q_0| > |q_2| & \end{bmatrix}$$

Example 4.5.5 *Examine the stability of the system described by the following polynomial:*

$$P(z) = z^3 + 3.3z^2 + 3z + 0.8 = 0$$

We form the Jury's array of coefficients:

row1.	0.8	3	3.3	1
row2.	1	3.3	3	0.8
row3.	−0.36	−0.9	−0.36	0

$$b_0 = \begin{vmatrix} a_0 & a_3 \\ a_3 & a_0 \end{vmatrix}, \ b_1 = \begin{vmatrix} a_0 & a_2 \\ a_3 & a_1 \end{vmatrix}, \ b_2 = \begin{vmatrix} a_0 & a_1 \\ a_3 & a_2 \end{vmatrix}, \ b_3 = \begin{vmatrix} a_0 & a_0 \\ a_3 & a_3 \end{vmatrix}$$

Since n = 3, then the following conditions should apply:

- *$P(1)$ must be positive: $1 + 3.3 + 3 + 0.8 = 8.1 > 0$ which is true*

- *$P(-1)$ must be negative because $n = 3 =$ is odd: $-1 + 3.3 - 3 + 0.8 = 0.1 > 0$ and this is false*

- *$|a_0| < a_n$, i.e: $|0.8| < 1$ which is true*

- *$|b_0| < |b_{n-1}|$, i.e: $|-0.36| = |-0.36|$ which is false*

One false condition is enough to conclude that the system is unstable.

Example 4.5.6 *Let us consider a dynamical system with the following characteristic equation:*

$$1 + K \frac{z}{(z-1)(z-0.4)} = 0$$

where K is a parameter to determine such that the system is stable.
This characteristic equation can be rewritten as follows:

$$z^2 + (K - 1.4)z + 0.4 = 0$$

Applying Jury criterion gives:

- *$P(1) > 0$, which gives $K > 0$*

- *$P(-1) > 0$ which gives $K < 2.8$*

- *$|a_0| < a_n$, i.e: $0.4 < 1$ which is true*

Therefore, our system will be stable if $K \in]0, 2.8[$. For instance, if we fix K to 2, which gives the following characteristic equation:

$$z^2 + 0.6z + 0.4 = 0$$

the roots are $z_{1,2} = -0.3000 \pm 0.5568j$ which are inside the unit circle since $|z_{1,2}| < 1$.

Another criterion to study the stability bas been developed by Raible. This stability Criterion consists also as for the Jury criterion to fill an array and then conclude on stability. To show how this criterion works, let us consider the following characteristic equation:

$$P(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n$$

where a_i is a real scalar.

row 1	a_0	a_1	$\cdots$	a_{n-1}	a_n	multiplier
row 2	a_n	a_{n-1}	$\cdots$	a_1	a_0	$\alpha_n = \frac{a_n}{a_0}$
row 3	$a_0^{(n-1)}$	$a_1^{(n-1)}$	$\cdots$	$a_{n-1}^{(n-1)}$	0	multiplier
row 4	$a_{n-1}^{(n-1)}$	$a_{n-2}^{(n-1)}$	$\cdots$	$a_0^{(n-1)}$	0	$\alpha_{n-1} = \frac{a_{n-1}^{(n-1)}}{a_0^{(n-1)}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$			
row 2n-1	$a_0^{(1)}$	$a_1^{(1)}$				multiplier
row 2n	$a_1^{(1)}$	$a_0^{(1)}$				$\alpha_1 = \frac{a_1^{(1)}}{a_0^{(1)}}$
row 2n+1	$a_0^{(0)}$					

- The 1$^{\text{st}}$ row is formed by the polynomial coefficients

- The 2$^{\text{nd}}$ row is formed by the same coefficients but in the opposite order

- The 3$^{\text{rd}}$ row is obtained by multiplying the 2$^{\text{nd}}$ row by $\alpha_n = \frac{a_n}{a_0}$, then by subtracting the result of the 1$^{\text{st}}$ row

- The 4$^{\text{th}}$ row is formed by coefficients of the 3$^{\text{rd}}$ row placed in the opposite order.

These procedures are repeated until the array gets $2n + 1$ rows. The last row contains only one number.

Raible's Stability Criterion

When $a_0 > 0$, the roots of the polynomial are all inside the unit circle if and only if $a_0^{(i)} > 0, i = 0, 1, \cdots, n - 1$

The coefficients $a_0^{(i)} > 0, i = 0, 1, \cdots, n - 1$ appear in the Raible's array .

Remark 4.5.1 *The assumption $a_0 > 0$ is not restrictive. In fact, when $a_0 < 0$, it is enough to change the signs of all coefficients of the polynomial $P(z)$ to obtain $-P(z)$, which in turn is used for Raible's criterion.*

This procedure is correct since the roots of $P(z)$ and of $-P(z)$ are identical.

Example 4.5.7 *To show how the Raible's criterion works, let us consider the following characteristic equation:*

$$P(z) = -z^3 - 0.7z^2 - 0.5z + 0.3$$

The coefficient a_0 must be positive, then we form the coefficient array of the polynomial $-P(z) = z^3 + 0.7z^2 + 0.5z - 0.3$

1	0.7	0.5	-0.3	
-0.3	0.5	0.7	1	$\alpha_3 = \frac{0.3}{-1} = -0.3$
0.91	0.85	0.71		
0.71	0.85	0.91		$\alpha_2 = \frac{0.71}{0.91} = 0.78$
0.36	0.19			
0.19	0.36			$\alpha_1 = \frac{0.19}{0.36} = 0.53$
0.26				

The system is stable because $a_0^{(i)} > 0$, $i = 0, 1, \cdots, n - 1$

We have presented some techniques to study the stability of discrete-time systems. It is also important to notice that we can also apply the criterions in the frequency domain.

4.6   Root Locus Technique

The root locus technique is a powerful approach that is usually used for continuous-time or discrete-time systems either for analysis or design. The technique gives an idea on how the poles of the closed-loop dynamics behave when a gain or more (a parameter or more) are changed. The direct conclusion is that we know immediately how the stability and the other performances of the system are affected by the parameters changes.

Nowadays there exist many tools to plot the root loci of any dynamical system some of them are available free for use. In the rest of this section, we will use Matlab for our plotting but we will develop rules of how obtain a sketch of the root locus in case we don't have a computer at hand.

As for the continuous case, the root locus for the discrete system is described by the characteristic equation that we write in the following form:

$$1 + KG(z) = 0$$

where K is the parameter that varies and

$$G(z) = \frac{(z - n_1)(z - n_2)\cdots(z - n_m)}{(z - z_1)(z - z_1)\cdots(z - z_n)}$$

with $z_1, z_2, \cdots, z_n$ are the poles and $n_1, n_2, \cdots, n_m$ are the zeros of the open loop transfer function.

When the parameter K varies from 0 to infinity (∞). The same rules as we use for the plotting of the root locus of the continuous-time systems in the s-plane apply to the plotting of the one of discrete-time systems in the z-plane, except that the interpretation of the results is different mainly in regard of stability.

From the characteristic equation, we get the following conditions:

$$\frac{1}{K} = \frac{\Pi_{i=1}^{m}|z - n_i|}{\Pi_{i=1}^{n}|z - z_i|} \tag{4.4}$$

$$\sum_{i=1}^{m} \arg(z - n_i) - \sum_{i=1}^{n} \arg(z - z_i) = (2k + 1)\pi, k = 0, 1, 2, \cdots, \tag{4.5}$$

The first condition is referred to as the magnitude condition while the second is referred to as angle condition. Any point in the z-plane that satisfies these two conditions belongs to the root locus of the system. To this point corresponds a gain K_{z_0}. If this point is z_0, then we have:

$$\frac{1}{K_{z_0}} = \frac{\Pi_{i=1}^{m}|z_0 - n_i|}{\Pi_{i=1}^{n}|z_0 - z_i|}$$

$$\sum_{i=1}^{m} \arg(z_0 - n_i) - \sum_{i=1}^{n} \arg(z_0 - z_i) = \theta_0$$

where θ_0 is the corresponding angle of this point.

A point of the z-plane will belong to the root locus, if it satisfies these two conditions. In general plotting the exact root locus for a given system is a hard task unless we have the appropriate tools for that. More often a sketch of this root locus can be easily obtained using some simple rules. Some of these rules are:

1. the number of branches is equal to the order of the system, i.e.: n;

2. the root locus is symmetric with respect to the real axis. This is due to the fact that the roots of the characteristic equation are either real or complex. And if there is a complex root, we have automatically its conjugate.

3. The loci originate from the poles of the open loop transfer function and terminate on the zeros of the this transfer function. To explain why the loci originate from the poles, we can make K equal to zero, while why the loci terminate on the zeros can be explained by letting K goes to ∞ in Eq. (4.4).

4. the number of asymptotes is equal to the difference between the number of poles, n, and the number of zeros, m, of the open loop transfer function. These asymptotes are characterized by:

$$\delta = \frac{\sum \text{poles} - \sum \text{zeros}}{n - m}$$

$$\beta_k = (2k + 1)\frac{\pi}{n - m}, k = 0, 1, 2, \cdots,$$

The parameter, δ, gives the intersections of the asymptotes with the real axis, while β_k gives the angle that make each asymptote with the real axis.

5. for the breakpoints of the root locus, firstly we determine the expression of the varying parameters K, i.e.:

$$K = \frac{\prod_{i=1}^{n}|z - z_i|}{\prod_{i=1}^{m}|z - n_i|}$$

The breakpoints are solution of the following equation:

$$\frac{dK}{dz} = 0$$

It is important to select from the roots of this equation those are feasible solution for the breakpoints.

6. the intersection of the imaginary axis in the z-plane can be determined by replacing z by jv in the characteristic equation and write it as follows:

$$\Re(K, v) + j\Im(K, v) = 0$$

that gives in turn two equations:

$$\Re(K, v) = 0$$
$$\Im(K, v) = 0$$

The solution gives the frequency at which the intersection occurs and the corresponding gain.

7. the angle of departure from a complex pole or the angle of arrival to a complex zero is computed using the angle condition. If the point at which we want to calculate the angle is z_0, the condition angle becomes:

$$\sum_{i=1}^{m} \arg(z_0 - n_i) - \sum_{i=1}^{n} \arg(z_0 - z_i) - \theta_0 = 180$$

where θ_0 is the corresponding angle of this point.

Example 4.6.1 *To show how the technique of root locus works, let us consider the system of the Fig. 4.24 where the plant is the double integrator and the controller is a proportional action with a gain K, that we will assume to change between zero and infinity for some physical reasons like heating, aging, etc.*

Using the $\mathscr{Z}$-transform table and the expression of the closed-loop transfer function we get the following characteristic equation of this system:

$$1 + K\frac{(z + 1)}{(z - 1)^2} = 0, \text{with } K = \frac{k}{2}$$

- *Number of branches: $n = 2$*

- *Finite number of branches: $m = 1$*

- *Infinite number of branches: $n - m = 2 - 1 = 1$*

- *Angle of asymptotes: $\beta = \frac{\pi(2k+1)}{n-m} = \frac{\pi(2k+1)}{2-1} = \pi, k = 0$*
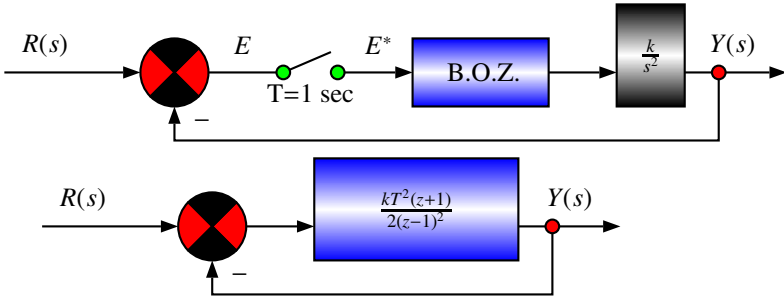
Fig. 4.24 BD of the system with characteristic eqn: $1 + K\frac{(z+1)}{(z-1)^2} = 0$

- *Intersection of the asymptote with the real axis:* $\delta = \frac{(1)+(1)-(-1)}{2-1} = 3$

- *Intersection of the locus with the real axis:* $\frac{dK}{dz} = 2z^2 + 4z - 6 = 0$, *which gives* $z_1 = -1$ *et* $z_2 = -3$.

    The root locus is illustrated in Fig. 4.25. All the roots are outside the unit circle in blue. The system is unstable. THis means that a proportional controller is not able to stabilize a double integrator.
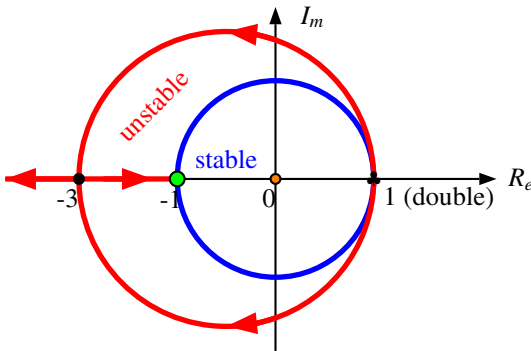
Fig. 4.25 RL of the system with characteristic eqn: $1 + K\frac{(z+1)}{(z-1)^2} = 0$

Example 4.6.2 *As a second example for the root locus technique let us consider the system of the Fig. 4.26.*
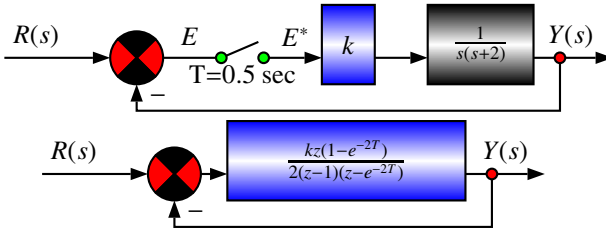
Fig. 4.26 BD of the system with characteristic eqn: $1 + K \frac{z}{(z-1)(z-0.368)} = 0$

The characteristic equation of this system is given by:

$$1 + k \frac{z(1 - e^{-2T})}{2(z-1)(z-e^{-2T})} = 1 + K \frac{z}{(z-1)(z-0.368)} = 0$$
$$with \ K = 0.316k$$

- *Number of branches: n = 2.*

- *Finite Number of branches: m = 1.*

- *Infinite Number of branches n − m = 2 − 1 = 1.*

- *Angle of asymptotes: $\beta = \frac{\pi(2k+1)}{n-m} = \frac{\pi(2k+1)}{2-1} = \pi$.*

- *Intersection of the locus with the real axis: $\frac{dK}{dz} = -z^2 + 0.368 = 0$. The resolution of this equations gives: $z_1 = -0.606$ et $z_2 = +0.606$.*

If we replace z by −1 in the characteristic equation, we find:

$$1 + K \frac{z}{(z-1)(z-0.368)} = 1 + K \frac{(-1)}{(-1-1)(-1-0.368)} = 0$$

which implies in turn:

$$K \ = 2.738$$

$$K \ = 0.316k$$

which gives:

$$k = \frac{K}{0.316} = \frac{2.738}{0.316} = 8.65$$

The root locus is drawn in Fig. 4.27. All the the roots are inside the unit circle in blue. Therefore, the system is stable for all gains k < 8.65.

4.7 Bode Plot Technique

The frequency response plays an important role in the analysis and design of continuous-time and discrete-time systems. As for the time response, the frequency
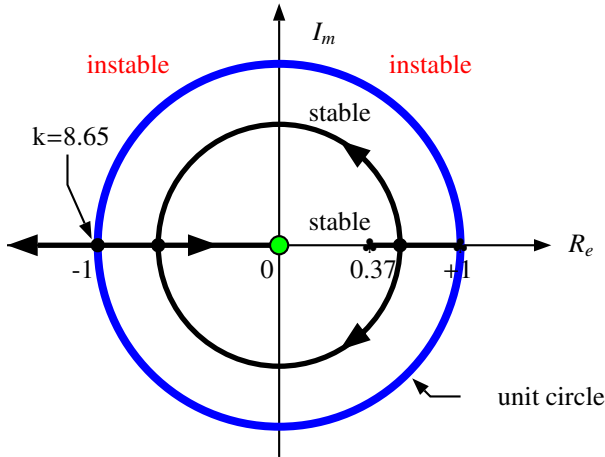
Fig. 4.27 RL of the system with characteristic eqn: $1 + K\frac{z}{(z-1)(z-0.368)} = 0$

response consists of exciting the system by a sinusoidal input. In the continuous-time system, it was proven that for a sinusoidal input, the output of the a stable linear system is sinusoidal with same frequency of the input, and the magnitude and the phase of the output are function of this frequency. For discrete-time system, the output is also sinusoidal with the same frequency as the input signal and the phase and the magnitude are still function of this frequency. To show this, let us consider a stable linear system with the following transfer function:

$$G(z) = \frac{Y(z)}{R(z)} = \frac{b_m z^m + b_{m-1} z^{m-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}$$
$$= K\frac{\Pi_{i=1}^{m}(z - n_i)}{\Pi_{i=1}^{n}(z - z_i)}$$

Let the input $r(t)$ has the following expression:

$$r(t) = \sin(wt)$$

where w is the frequency of the input. The magnitude is taken here equal to one.

The $\mathscr{Z}$-transform of this signal is given by (see $\mathscr{Z}$-transform table):

$$R(z) = \frac{z\sin(wt)}{z^2 - 2z\cos t(wt) + 1} = \frac{z\sin(wT)}{(z - e^{-jwT})(z - e^{jwT})}$$

Now if we consider that the system is excited by $R(z)$ the corresponding output, $Y(z)$ is given by:

$$Y(z) = G(z)R(z)$$
$$= K\frac{\Pi_{i=1}^{m}(z - n_i)}{\Pi_{i=1}^{n}(z - z_i)} \frac{z\sin(wT)}{(z - e^{-jwT})(z - e^{jwT})}$$

To get the expression of the output, let us proceed with a partial fraction of $Y(z)$. This gives:

$$Y(z) = \frac{cz}{z - e^{-jwT}} + \frac{\bar{c}z}{z - e^{jwT}} + \text{terms due to } G(z)$$

Let s now multiply both sides this equality by $\frac{(z-e^{-jwT})}{z}$ to get the following:

$$G(z)\frac{sinwT}{(z - e^{jwT})} = c + \frac{\bar{c}\left(z - e^{-jwT}\right)}{z - e^{jwT}} + \left[\frac{\left(z - e^{-jwT}\right)}{z}\right] \text{terms due to } G(z)$$

where

$$c = \left[G(z)\frac{sin(wT)}{(z - e^{jwT})}\right]_{\big|_{z=e^{-jwT}}}$$

$$\bar{c} = \text{conjugate of } c$$

Notice that $e^{-jwT} = cos - wT + jsin - wT = coswT - jsinwT$, which implies that

$$\left(z - e^{jwT}\right)_{\big|_{z=e^{-jwT}}} = -2jsinwT$$

Using this we get:

$$c = \frac{G(e^{-jwT})}{-2j}$$

$$\bar{c} = \frac{G(e^{jwT})}{2j}$$

Using now the fact that for any complex number we have:

$$G(e^{jwT}) = M(w)e^{j\theta(w)}$$

where M and θ represent respectively the magnitude and the phase at the frequency w.

The steady state, the terms due to $G(z)$ vanish and we have:

$$Y(z) = \frac{G(e^{-jwT})}{-2j}\frac{z}{z - e^{-jwT}} + \frac{G(e^{jwT})}{2j}\frac{z}{z - e^{jwT}}$$

$$= \frac{M(w)}{2j}\left[-\frac{e^{-\theta(w)}z}{z - e^{-jwT}} + \frac{e^{\theta(w)}z}{z - e^{jwT}}\right]$$

The $\mathscr{Z}$-transform inverse of $Y(z)$ at the steady state is given by:

$$y(kT) = \frac{M(w)}{2j}\left[e^{j\theta(w)}e^{jwT} - e^{-j\theta(w)}e^{-jwT}\right]$$

$$= \frac{M(w)}{2j}\left[e^{j(\theta(w)+wT)} - e^{-j(\theta(w)+wT)}\right]$$

$$= M(w)sin\left(wT + \theta(w)\right)$$

Remark 4.7.1 *It is important to mention that the magnitude and the phase of the output for a sinusoid input are both functions of its frequency. Therefore, their values will change when the frequency changes.*

A certain parallel can be made with frequency response of continuous time. In fact, for these system, the frequency response can be obtained from the transfer function, $G(s)$ that describes the system by doing the following:

- *the magnitude $M(w)$ is given by:*

$$M(w) = |G(jw)|$$

- *the phase $\theta(w)$ is given by:*

$$\theta(w) = \arg\left(G(jw)\right)$$

This means that the magnitude and the phase of the output at frequency w are obtained from the transfer function of the system by replacing firstly s by jw and then compute the magnitude and the phase using the previous formulas.

For the discrete time, the same reasoning applies except that we have to replace z by e^{jwT} and use the following formulas:

- *the magnitude $M(w)$ is given by:*

$$M(w) = \left|G(e^{jwT})\right|$$

- *the phase $\theta(w)$ is given by:*

$$\theta(w) = \arg\left(G(e^{jwT})\right)$$

Some precautions have to be taken for the frequency response of discrete time system. In fact, the $\mathscr{Z}$-transform is obtained by replacing z by e^{sT}. Therefore, the primary and the complementary strips of the left hand side of the s-domain are mapped to the interior of the unit circle in the z-domain. If we replace in turn z by e^{jwT} to get the frequency response of the discrete time system, the result we will get has no sense since it deals with the entire z-plane. To avoid this the following transformation is usually used:

$$z = \frac{1 + \frac{T}{2}\omega}{1 - \frac{T}{2}\omega}$$

which implies:

$$\omega = \frac{2}{T}\frac{z-1}{z+1}$$

Using the $\mathscr{Z}$-transform and and the w-transform respectively, the primary trip of the left half of the s-plane is then transformed into the unit circle which in turn transformed to the entire left half of the w-plane. More specifically, the range of frequencies in the s-plane $-\frac{\omega_s}{2} \leq w \leq \frac{\omega_s}{2}$ is firstly transformed into the unit circle in the z-plane, which in turn transformed into the entire left half of the w-plane.

Finally, it is important to notice the relationship between the frequencies ω and ν. In fact, ω is defined by:

$$\omega|_{\omega=i\nu} = j\nu = \left[\frac{2}{T}\frac{z-1}{z+1}\right]_{|z=e^{j wT}}$$

$$= \frac{2}{T}\frac{e^{jwT}-1}{e^{jwT}+1}$$

Multiplying the numerator and the denominator by e^{-jwT}, we get:

$$w|_{w=i\nu} = j\nu$$

$$= j\frac{2}{T}\tan\left(\frac{wT}{2}\right)$$

which gives the following relationship between w and ν:

$$w = \frac{2}{T}\tan\left(\frac{wT}{2}\right)$$

At low frequencies, we have equality between these frequencies. In fact, when w is low, we have $\tan\left(\frac{wT}{2}\right) = \frac{wT}{2}$, which gives $w = \nu$.

Based on this remark, the frequency response of the discrete time consists then of replacing w by $j\nu$, with ν is a fictitious frequency, in the new expression of the transfer function obtained after replacing z by $z = \frac{1+\frac{T}{2}w}{1-\frac{T}{2}w}$. To have an idea on how the frequency response can be plotted, let us consider the following example.

Example 4.7.1 *As a first example of the frequency response, let us consider the system of the Fig. 4.28. It represents the speed control of a load driven by a dc motor. The controller is a proportional. The transfer function of the system and the controller is given by:*

$$\bar{G}(s) = \frac{K_p k}{\tau s + 1} = \frac{K}{\tau s + 1}$$
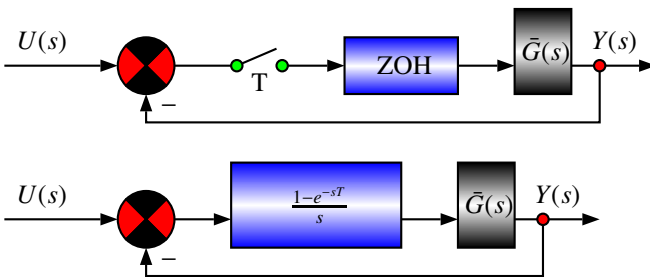
Fig. 4.28 Speed control of mechanical part driven by a dc motor

Firstly, let us compute the open loop transfer function of the system in Fig. 4.26. Since we have a ZOH we get:

$$G(s) = \left(1 - e^{-sT}\right) \frac{K}{s(\tau s + 1)}$$

where $K = K_p k = 2$, $\tau = 1s$ and T is the sampling period used for our system and it is equal to 0.1s.

Using the $\mathscr{Z}$-transform table we get:

$$
\begin{aligned}
G(z) &= K \frac{(z-1)}{z} \frac{z\left(1 - e^{-T}\right)}{(z-1)(z - e^T)} \\
&= K \frac{\left(1 - e^{-T}\right)}{(z - e^T)} \\
&= \frac{0.1903}{z - 0.9048}
\end{aligned}
$$

Replacing now z by $\frac{1 + \frac{T}{2}w}{1 - \frac{T}{2}w} = \frac{1 + 0.05w}{1 - 0.05w}$, we get:

$$
\begin{aligned}
G(z) &= \frac{0.1903}{\frac{1+0.05w}{1-0.05w} - 0.9048} \\
&= \frac{0.1903\,(1 - 0.05w)}{0.0952 + 0.0952w} \\
&= \frac{1.9989\,(1 - 0.05w)}{1 + w}
\end{aligned}
$$

Using Matlab, we can get the bode diagram of this transfer function as illustrated by Fig. 4.29.

4.8   Conclusions

This chapter covers the analysis tools based on the transfer function concept. Mainly, we developed the techniques of how to compute the time response and determine the system performances. We also presented the root locus and bode plot techniques.

4.9   Problems

1. Compute the $\mathscr{Z}$-transform of the following signals:
   (a) the unit step
   (b) the unit ramp
   (c) the unit exponential
   (d) $r(t) = t + sinwt$
   (e) $1 - coswt$
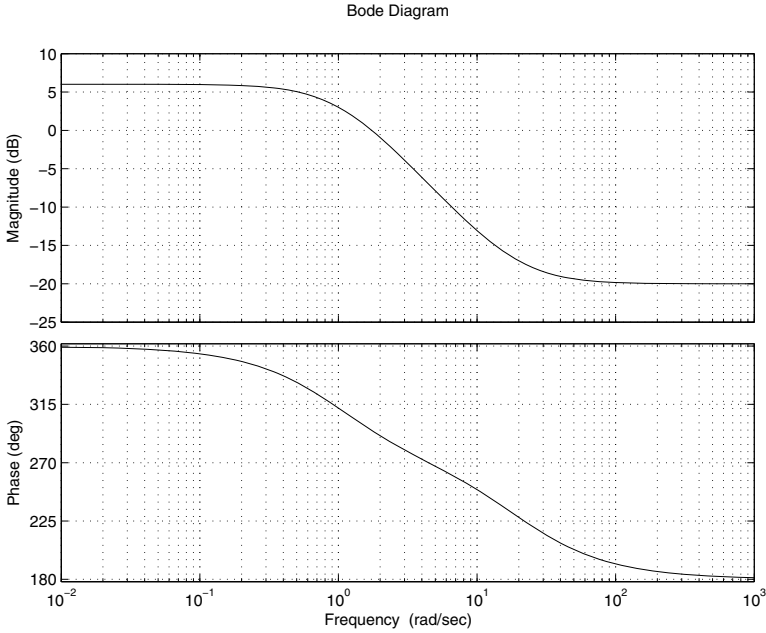
Bode Diagram

Fig. 4.29 Bode diagram of $\frac{1.9989(1-0.05w)}{1+w}$

2. Compute the expression of the signal in time of the following ones in z:

   (a) $Y(z) = \frac{Tze^{aT}}{(z-e^{-aT})^2}$, $a > 0$

   (b) $Y(z) = \frac{z(1-e^{aT})}{(z-1)(z-e^{-aT})}$, $a > 0$

   (c) $Y(z) = \frac{1}{b-a}\left[\frac{z}{z-e^{aT}} - \frac{z}{z-e^{bT}}\right]$, $a > 0$, $b > 0$ and $a \neq b$

3. For the dynamical systems with the input $u(t)$ and the output $y(t)$ with the following dynamics:

   - $\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} = u(t)$
   - $\frac{d^2y(t)}{dt^2} + 4\frac{dy(t)}{dt} + 4y(t) = 4u(t)$
   - $\frac{d^2y(t)}{dt^2} + 6\frac{dy(t)}{dt} + 8y(t) = 8u(t)$
   - $\frac{d^3y(t)}{dt^3} + 3\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} = u(t)$

   (a) determine the sampling period T
   (b) using the approximation methods determine the relationship between the input $U(z)$ and the output $Y(z)$
   (c) determine the pulse transfer function for each dynamics
   (d) using Matlab compute the step response of each dynamics

(e) using now the zero-order-hold, determine the corresponding transfer function and compute the step response. Compare this response to the one of the previous question

4. In this problem we consider the system of the Fig. 4.30 where the transfer function of the system is given by:

$$G(s) = \frac{10}{(s+1)(s+10)}$$
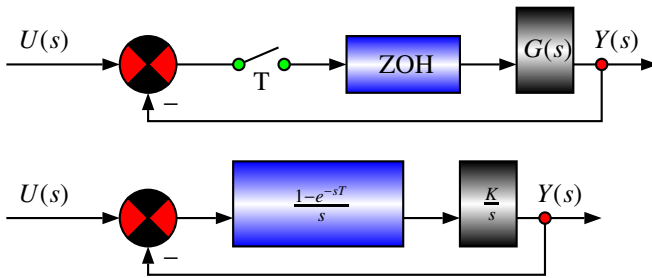
Fig. 4.30 Transfer functions in feedback

(a) determine the sampling period that we can use for this system
(b) using this sampling period determine the open loop transfer function and the closed-loop one
(c) determine the step response of the system
(d) plot the behavior of the output with respect to time

5. Study the stability of the dynamical systems with the following characteristic equation:

(a) $z^3 + 0.8z^2 + 0.17z + 0.01$
(b) $z^4 + 1.4z^3 + 0.65z^2 + 0.112z + 0.006$
(c) $z^5 + 2.39z^4 + 2.036z^3 + 0.7555z^2 + 0.1169z + 0.0059$
(d) $z^5 + 11.4z^4 + 14.65z^3 + 6.6120z^2 + 1.126z + 0.06$

6. In this problem we consider the dynamical system show in the block diagram illustrated by the Fig. 4.31. The transfer functions are given by:

$$G(z) = \frac{z\left(1 - e^{aT}\right)}{(z-1)\left(z - e^{-aT}\right)}$$

$$C(z) = K$$

with $a = 0.1$ and $T = 0.01$
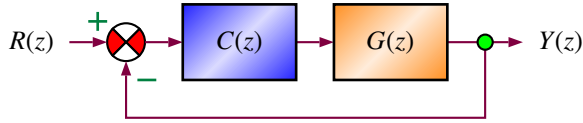
Fig. 4.31 Block diagram of the closed-loop

   (a) study the stability in function of the gain K

   (b) plot the root locus of the system and conclude on the stability

7. Consider the system of the Fig. 4.30 with the following expression for $G(s)$:

$$G(s) \; = \; \frac{K}{s(\tau s + 1)}$$

with K is the gain and $\tau = 1s$ is the time constant of the system.

- determine the sampling period
- compute the transfer function $G(z)$
- plot the root locus the system when the gain K varies between 0 and ∞

8. Consider the system of the Fig. 4.30 with the following expression for $G(s)$:

$$G(s) \; = \; \frac{K}{s(\tau s + 1)}$$

with $K = 10$ is the gain and $\tau = 0.1s$ is the time constant of the system.

- determine the sampling period
- compute the transfer function $G(z)$
- plot the Bode diagram of the system

5

Design Based on Transfer Function

After reading this chapter the reader will:

1. master the concept of the design of classical controllers based on the transfer function of the system

2. be able to choose the structure of the classical controller that responds to the desired performances and determine its parameters

3. be familiar with the design of the proportional, proportional and integral, proportional integral and derivative controllers and their approximations

4. be able to determine the recurrent equation for the control law that we must implement in the microcontroller

5.1   Introduction

Tackling a control design problem is always a challenge even for more experienced control engineers. The system for which the controller must be designed, may be an existing one with some poor performances and that we would like to improve, or a new system that we are building. In both cases, the design procedure starts, after

getting the mathematical model for the system, by defining the desired performances that will allow us to determine the structure of the controller and its parameters.

More often the control systems are designed to guarantee certain performances to the closed-loop dynamics of the system under consideration. Such performances can be summarized to the stability and the behaviors of the transient and the steady state regimes. By respecting the limitations of the given system, it is always the case that we search to improve the transient regime by searching for a compromise between the overshoot that the system may have and its rapidity. For the steady state, we search to guarantee that the error is less than a certain chosen tolerance. The controllers we will consider in this chapter to respond to the design requirements are classical ones like the proportional, integral and derivative actions and their approximations.

The rest of the chapter is organized as follows. In Section 2, the control design problem is formulated. Section 3 presents the empirical methods to design classical controllers. In Section 4, the design of classical controllers using the root locus method is developed. Section 5 presents the Bode method. Section 6 presents a case study which consists of designing different controllers for the dc motor kit.

5.2   Formulation of the Control Design Problem

In this chapter we will consider an existing system with poor performances that we would like to improve. Our desire is to act simultaneously on the transient and steady state regimes by introducing a controller in the closed-loop to force the overall system to behave as it is desired.

The performances may be given either in time or frequency domains. In both domains, the stability is the first requirement in the design procedure. Beside the stability, we would like the transient and the steady state regimes to behave in a desirable ways.

In the time domain for the transient regime, we should control the overshoot, the rising time and the settling time for a chosen percentage that will depend on the precision we would like to guarantee to our system. For the steady state regime, we would like to assure that the system's error is less than a certain specified value.

In the frequency domain, the situation is similar except that the performances are given in function of the stability of the closed-loop dynamics, the gain phase and the margin phase, the bandwith, etc. In general, it is hard to establish a link between the performances in the time domain and the ones in the frequency domain.

More specifically, the system under study is described by a transfer function that can be obtained using the identification approach for instance. Let us denote by $G(s)$ this transfer function. This model must be determined in the first stage of the control design. Then, from the performances and the expertise of the control engineer design we can choose the structure of the controller that may respond properly to the design goal. Then using the appropriate approach we can determine the controller gains.

Therefore, the control design problem consists of determining:

- the structure of the controller

- and its parameters

using the desired performances and some heuristics approaches to force the closed-loop dynamics with the chosen controller to behave as it is desirable. This approach may require some refinement in practice due to different phenomena like neglected dynamics.

The controllers we will consider in this chapter are the classical controllers referred in the literature to as the combination of the proportional (P), integral (I) and derivative (D) actions and their approximations referred also to as phase lag, phase lead and phase lead-lag. The transfer function of the controller will be denoted by $C(z)$. Once the controller is determined, the corresponding difference equation is obtained and implemented in real time using an appropriate microcontroller. For more detail on this topic, we refer the reader to the implementation part where this is detailed.
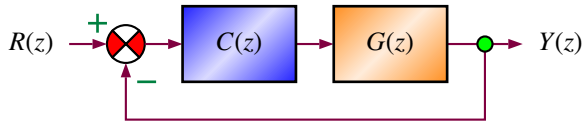
Fig. 5.1 Block diagram of the closed-loop

More often, the system's performances are given in continuous-time since it is more natural to do so. The design procedure can be done either in the continuous-time or the discrete-time. Generally speaking, the design approach uses the following steps:

- the performances are converted to poles

- the structure of the desired controller is chosen

- the controller parameters are determined using the desired poles

- some tunings of the controller's parameters are done to compensate for the discrepancy between the desired and the real behaviors that may result from system's zeros that are not considered in the design procedure.

It is important to notice that determination of the controller parameters can be done either in the continuous-time or the discrete-time. In the continuous-time case, the controller parameters are determined and after that the controller transfer function is converted to discrete-time domain to get the difference equation that we should implement in real time. For the discrete-time, the difference equation is directly obtained and implemented.

The design approach can be one of the following methods:

- Design based on empirical methods

- Design based on root locus method

- Design based on Bode plot method

In the rest of this chapter we will cover these methods and present some examples to show how these techniques apply for real systems. Simulations results will be used to show their validity. The design of the controller is done in continuous-time and then the corresponding discrete-time version of the controller is obtained. The methods developed in Boukas (see [1]) are used in this chapter.

5.3   Design Based on Empirical Methods

The empirical methods are based on the work of Ziegler-Nichols. These methods have the advantage over the other methods since they permit the design of the desired controller even in the absence of the mathematical model of the system. The Ziegler-Nichols methods are mainly based on the response of the dynamical system. Ziegler-Nichols proposed methods that use the time response and others using the frequency response. In the rest of this section we will cover these methods.

Let us first of all concentrate on the time response methods. In these methods, we can handle systems that are stable and unstable in open loop. The first method considers the case of stable system with no poles at the origin neither dominant complex pair of poles. In this case, the step response is given by the one in Fig. 5.1, from which the parameters T, τ and k are determined directly and the Tab. 5.1 is used to fix the controller parameters directly. The corresponding expression of $G(s)$ is given by the following:

$$G(s) \ = k \frac{e^{-\tau s}}{T s + 1}$$

where k is the gain of the system, τ is the rise time and T is the delay time.

The general expression for the controllers used by the Tab. 5.1 is given by:

$$C(s) \ = K_P \left[1 + \frac{1}{T_I s} + T_D s \right]$$

where K_P, T_I and T_D are the controller parameters to be fixed using Tab. 5.1.

Remark 5.3.1 *It is important to notice that the Ziegler-Nichols method is applicable only when the following holds:*

$$0.15 \le \frac{\tau}{T} \le 0.6$$

The following procedure can be used to fix the controller parameter:

1. obtain the step response of the open loop system

2. determine the values of the parameters τ and T from this time response

3. compute the controller parameters using Tab. 5.1

4. compute the closed-loop transfer function and check if the performances are obtained

5. adjust the parameters of the controller if necessary to obtain the desired performances

Remark 5.3.2 *Mostly the time response we will obtain using the controllers fixed by Tab. 5.1 has an overshoot between 10 % and 60 % and an adjustment of the controller parameters is always necessary.*
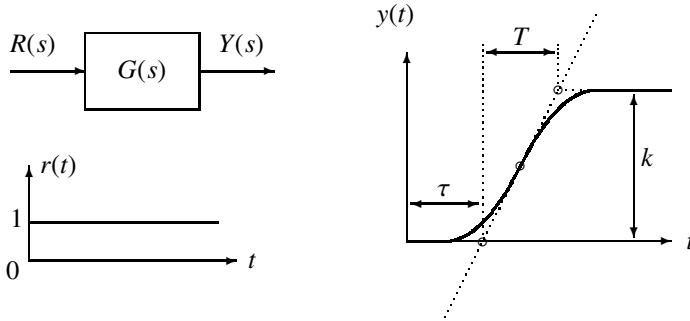
Fig. 5.2 Ziegler-Nichols methods: stable case

Table 5.1 Ziegler-Nichols methods: controller parameters

Controllers	Parameters
P	$K_P = \frac{T}{\tau}$
PI	$K_P = \frac{0.9T}{\tau}$
	$T_I = 3.3\tau$
PID	$K_P = \frac{1.2T}{\tau}$
	$T_I = 2\tau$
	$T_D = 0.5\tau$

Remark 5.3.3 *The values of the gains, K_P is computed using $k = 1$. If it is not the case, the controller gain, K_P has be to corrected by dividing the value of Tab. 5.1 by k. As an example, the gain in case of PID is $K_P = \frac{1.2T}{k\tau}$ instead of $K_P = \frac{1.2T}{\tau}$.*

Example 5.3.1 *To show how this method works, let us consider a dynamical stable system with step response as illustrated in Fig. 5.3. From this figure we get the following parameters:*

$$k = 2$$
$$\tau = 0.2$$
$$T = 1$$

From these data, we conclude that the condition of the Ziegler-Nichols is satisfied and therefore, we can use the Tab. 5.1 to fix the desired controller.
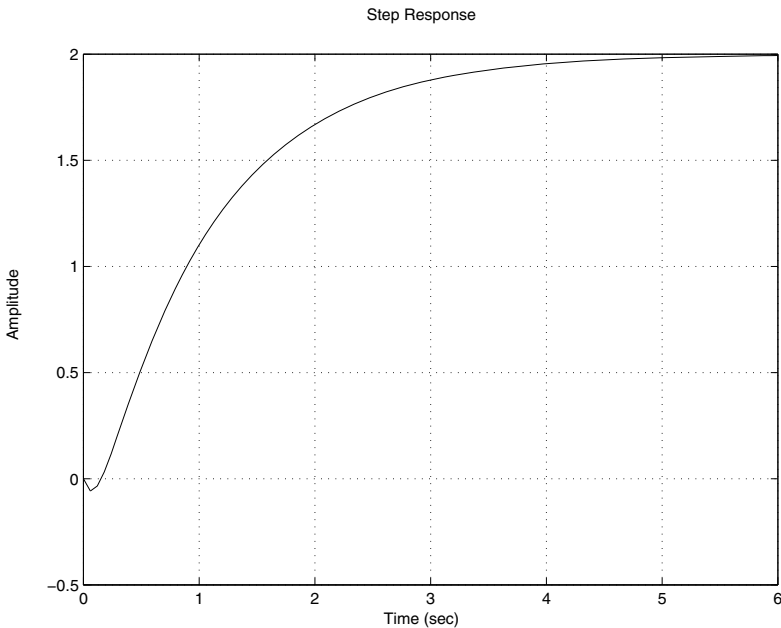
Fig. 5.3 Step response of a stable dynamical system

If we opt for a PID, the parameters of this controller are given by:

$$K_P = \frac{1.2T}{\tau} = 1.2$$
$$T_I = 2\tau = 0.4$$
$$T_D = 0.5\tau = 0.1$$

The closed-loop dynamics with this controller is given by:

$$F(s) = \frac{2K_P(T_IT_Ds^2 + T_Is + 1)e^{-\tau s}}{s(T_ITs + T_I) + 2K_P(T_IT_Ds^2 + T_Is + 1)e^{-\tau s}}$$

Using the Padé approximation, i.e.:

$$e^{-\tau s} = \frac{1 - \frac{\tau}{2}s}{1 + \frac{\tau}{2}s}$$

we get:

$$F(s) = \frac{2K_P(T_I T_D s^2 + T_I s + 1)(1 - \frac{\tau}{2}s)}{a_3 s^3 + a_2 s^2 + a_1 s + a_0}$$

with $a_3 = \frac{\tau}{2}T_I[T - 2K_P T_D]$, $a_2 = T_I\left[T + \frac{\tau}{2} + 2K_P(T_D - \frac{\tau}{2})\right]$, $a_1 = \left[T_I + 2K_P(T_I - \frac{\tau}{2})\right]$ *and* $a_0 = 2K_P$.

The step response of the closed-loop dynamics with this controller is illustrated by Fig. 5.4. From this figure we can see that the overshoot is approximatively 20 % and the other performances are acceptable.
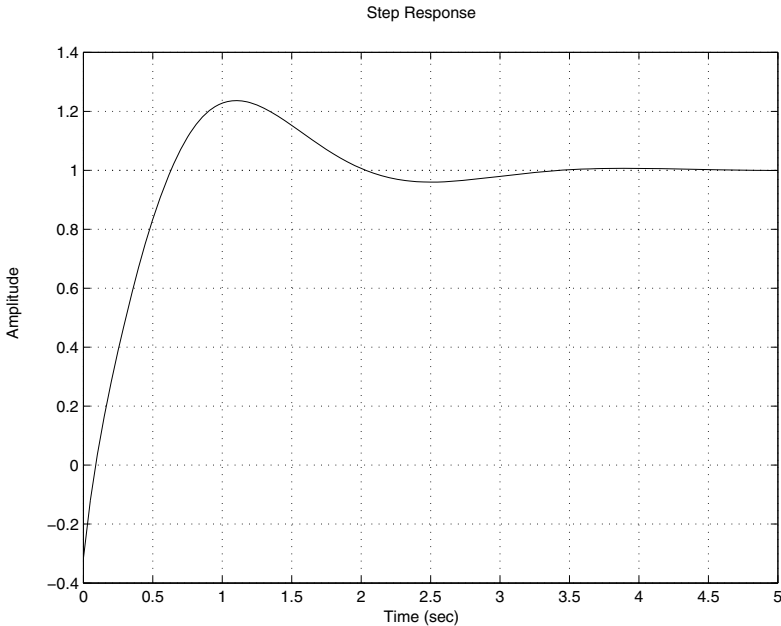
Fig. 5.4 Step response of the closed-loop dynamics with a PID controller

.

Let us now consider the case of unstable systems in open loop. For this class of systems, the approach consists of mounting the system with a PID controller with $T_I = \infty$ and $T_D = 0$ and by varying the gain K_P to bring the closed loop dynamics to the limit of stability (periodic oscillations). Let denote by $\bar{K}_P$ and T_c the corresponding gain and the corresponding period. Fig. 5.5 gives an idea of such set-up. Once these two parameters are determined the ones for the controllers can be obtained using Tab. 5.2.
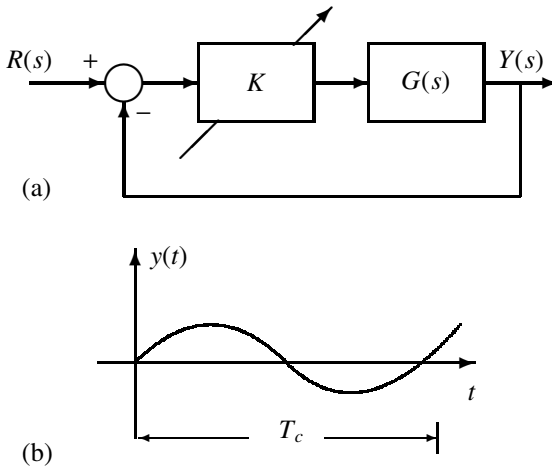
Fig. 5.5 Ziegler-Nichols: unstable case (a) and determination of T_c (b)

Table 5.2 Ziegler-Nichols method: case of unstable systems

Controllers	Parameters
P	$K_P = 0.5\bar{K}_P$
PI	$K_P = 0.45\bar{K}_P$
	$T_I = 0.83T_c$
PID	$K_P = 0.6\bar{K}_P$
	$T_I = 0.5T_c$
	$T_D = 0.125T_c$

The following procedure can be used to fix the controller parameter:

1. mount the system in closed loop with $T_I = \infty$ and $T_D = 0$ and vary the proportional gain of the controller, K_P till the time response gives oscillations as in Fig. 5.5

2. determine the values of the parameters $\bar{K}_P$ and T_c from this time response

3. compute the controller parameters using Tab. 5.2

4. compute the closed-loop transfer function and check if the performances are obtained

5. adjust the parameters of the controller if necessary to obtain the desired performances

Example 5.3.2 *To show how the Ziegler-Nichols method in case of unstable system works, let us consider the following dynamical system:*

$$G(s) = \frac{1}{s\,(0.1s + 1)\,(0.2s + 1)}$$

It is important to notice that this transfer function has a pole at the origin and therefore, the first method will not work.

Now if we mount this system with a proportional controller, we get the following characteristic equation:

$$1 + K_P \frac{1}{s\,(0.1s + 1)\,(0.2s + 1)} = 0$$

The corresponding Routh Hurwitz table is given by:

$$
\begin{array}{c|cc}
s^3 & 1 & 50 \\
s^2 & 15 & 50K_P \\
\hline
s^1 & \frac{15\times 50 - 50K_P}{15} & 0 \\
s^0 & K_P & 0
\end{array}
$$

The critical gain, $\bar{K}_P$ is given by $\bar{K}_P = 15$. The corresponding complex poles are solution of the following equation:

$$15s^2 + 50\bar{K}_P = 0$$

which gives:

$$s = \pm j\sqrt{50} = \pm 7.0711 j$$

The period T_c is equal to $\sqrt{50}$.
If we choose a PID controller its parameters are given by:

$$K_P = 0.6\bar{K}_P = 9$$
$$T_I = 0.5T_c = 0.4443$$
$$T_D = 0.125T_c = 0.1111$$

The closed-loop dynamics with this controller is given by:

$$F(s) = \frac{K_P\left(T_I T_D s^2 + T_I s + 1\right)}{0.02T_I s^4 + 0.3T_I s^3 + (T_I + K_P T_I T_D)\,s^2 + K_P T_I s + K_P}$$

The step response of the closed-loop dynamics is illustrated by Fig. 5.6.

To close this section let us see how we can design PID controllers (P, PI, PID) using the Ziegler-Nichols frequency methods (these methods are mainly based on the idea to assure for the closed-loop dynamics a margin phase between 45° and 50° and gain margin greater than 8 *db*). For this purpose, let us assume that the dynamics of the system in open loop is described by:

$$G(s) = k\frac{1}{\Pi_{i=1}^{n}(\tau_i s + 1)}$$
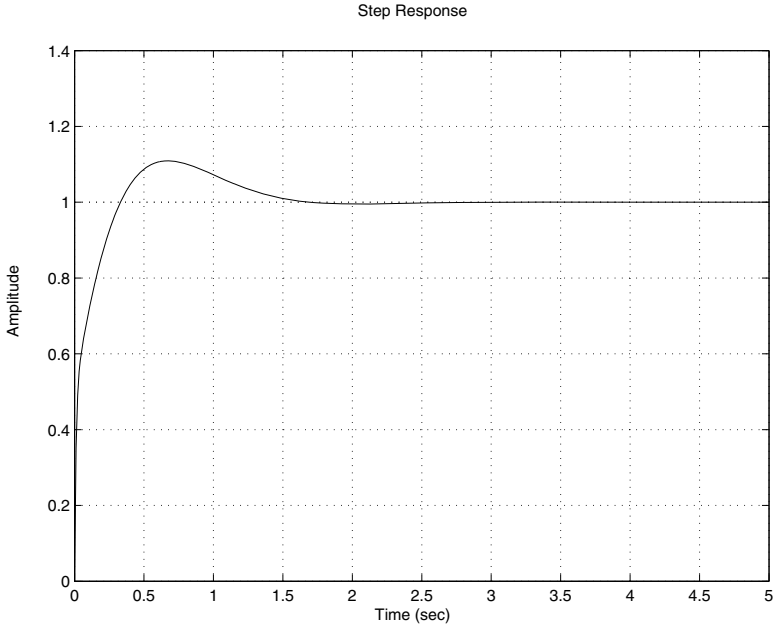
Step Response

Fig. 5.6 Step response of the closed-loop dynamics with a PID controller

where k is the gain of the system and $\tau_i, i = 1, \cdots, n$ are the different constant time of the system.

By defining $\bar{K}_P$ as the gain in open loop that assures the gain margin and the phase margin, and τ^1 and τ^2 as follows:

$$\tau^1 = \max\{\tau_1, \cdots, \tau_n\}$$
$$\tau^2 = \max\{\{\tau_1, \cdots, \tau_n\} - \{\tau^1\}\}$$

the controller parameters are fixed by Tab. 5.3 . The expression of the PID controller is given by:

$$C(s) = K_P \frac{(\tau^1 s + 1)(\tau^2 s + 1)}{(\tau^1 + \tau^2)s}$$

It is important to notice that the open transfer function is given by:

$$T(s) = C(s)G(s) \tag{5.1}$$

$$= \begin{cases} \frac{kK_P}{\prod_{i=1}^n (\tau_i s+1)} = \frac{K}{\prod_{i=1}^n (\tau_i s+1)} & \text{for P controller, with } K = kK_P \\ \frac{kK(T_I s+1)}{T_I s \prod_{i=1}^n (\tau_i s+1)} = \frac{K(T_I s+1)}{s \prod_{i=1}^n (\tau_i s+1)} & \text{for PI controller, with } K = \frac{kK_P}{T_I} \\ \frac{kK_P(T_I T_D s^2 + T_I s+1)}{T_I s \prod_{i=1}^n (\tau_i s+1)} = \frac{K(T_I T_D s^2 + T_I s+1)}{s \prod_{i=1}^n (\tau_i s+1)} & \text{for PID controller, with } K = \frac{kK_P}{T_I} \end{cases}$$

The following procedure can be used to design the appropriate controller using the following steps:

Table 5.3 Ziegler Nichols method in frequency domain

Controllers	Parameters
P	$K_P = \frac{\bar{K}_P}{k}$
PI	$K_P = \frac{\bar{K}_P}{k}$
	$T_I = \tau^1$
PID	$K_P = \frac{\bar{K}_P}{k}$
	$T_I = \tau^1 + \tau^2$
	$T_D = \frac{\tau^1 \tau^2}{T_I}$

1. determine the open loop transfer function with the compensator as in 5.1

2. plot the bode diagram for $K = 1$ and determine the gain $\bar{K}_P$ that gives the desired phase margin and a gain margin greater than 8 *db*

3. determine the gain, K_P of the controller using:

$$K_P = \begin{cases} \frac{\bar{K}_P}{k} & \text{fpr P controller} \\ \frac{\bar{K}_P T_I}{k} & \text{fpr PI controller, with } T_I = \tau^1 \\ \frac{\bar{K}_P T_I}{k} & \text{fpr PID controller, with } T_I = \tau^1 + \tau^2, T_D = \frac{\tau^1 \tau^2}{T_I} \end{cases}$$

4. check if the performances of the system are satisfied. In case of negative answer, adjust the controller parameters to get such performances.

Example 5.3.3 *To show how this method work let us consider the following dynamical system:*

$$G(s) = \frac{4}{(0.1s + 1)(0.2s + 1)(0.5s + 1)}$$

Our goal is to design a PID controller that provides the following performances:

1. *stable system*

2. *margin phase between* 45° *and* 50°

3. *margin gain greater than* 8 *db*

First of all following the step of the previous, we have:

$$\tau^1 = 0.5$$
$$\tau^2 = 0.2$$

which gives:

$$T_I = \tau^1 + \tau^2 = 0.5 + 0.2 = 0.7$$
$$T_D = \frac{\tau^1 \tau^2}{T_I} = \frac{0.2 \times 0.5}{0.7} = 0.1429$$

The gain, $\bar{K}_P$ that gives the desired phase margin and gain margin greater than 8 *db* is given by:

$$\bar{K}_P = 3.8019$$

which gives the following gain for the PID controller:

$$K_P = \frac{\bar{K}_P T_I}{k} = \frac{3.8019 \times 0.7}{4} = 0.6653$$

The transfer function of the closed-loop dynamics with this controller is given by:

$$F(s) = \frac{kK_P T_D s^2 + kK_P s + \frac{kK_P}{T_I}}{0.01 s^4 + 0.17 s^3 + (0.8 + KK_P T_D)\, s^2 + (1 + kK_P)\, s + \frac{kK_P}{T_I}}$$

The step response of the closed-loop dynamics is illustrated by Fig. 5.7.
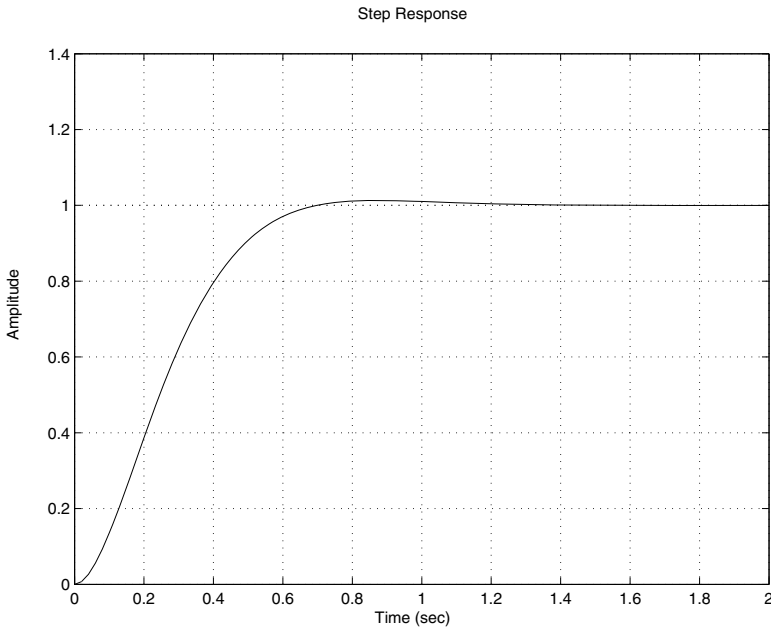
Fig. 5.7 Step response of the closed-loop dynamics with a PID controller

Remark 5.3.4 *For the expression of the controller in the discrete-time and its analytical expression of the recurrent equation for real-time implementation, we will cover this in the next section.*

5.4   Design Based on Root Locus

The root locus technique is a powerful tool for analysis and design of control systems. In this section, we will use it to design a controller that will guarantee the desired performances. The model of the system is supposed to be given in term of a transfer function.

The root locus technique can be used to design the classical controllers. The technique behind this method consists of choosing the controller gains that make the loci passes through given poles that come from the performances. In the rest of this section we will assume that the transfer function $G(s)$ is given by the following expression:

$$G(s) = k\frac{\Pi_{i=1}^{n}(s + z_i)}{\Pi_{i=1}^{n}(s + p_i)}$$

where k, $-z_i$ and $-p_i$ are respectively the gain, the zero and the pole of the system.

Let us firstly concentrate on the design of the proportional controller. Let its transfer function be given by:

$$C(s) = K_P$$

where K_P is the gain of the controller to be determined.

As it is well known from basic control course, the proportional controller acts simultaneously on the transient and the steady state regimes but its capacity is limited. It can reduce the error but never makes it equal to zero.

To compute the gain of the controller we will use the following procedure (see Boukas [1]):

1. compute the characteristic equation of the closed-loop dynamics, i.e.: $1+K_pG(s)$ and let $K = kK_p$

2. draw the root locus for K varying from 0 to infinity

3. determine the intersection between the loci and the line corresponding to the desired damping ratio ξ, $(\cos\theta = \xi)$ and get the dominant pair of poles. Let s_d be the one with the positive imaginary part.

4. compute the gain K that gives the pole s_d, then obtain the desired gain for the proportional controller by:

$$K_P = \frac{\Pi_{i=1}^{n}|(s_d + p_i)|}{K\Pi_{i=1}^{m}|(s_d + z_i)|}$$

The lines that we should include in the control loop during the implementation part are:

```
compute the system's error, e
compute the control law using   u = Kp*e
 send the control and wait for the next interrupt
```

Example 5.4.1 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in position.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{s(\tau s + 1)}$$

with $k = 5$, and $\tau = 1s$.

From basic control theory, we can see that the system is unstable. Our desire is to make it stable in the closed-loop with an overshoot less or equal to 5 % and a steady state error equal to zero.

From basic control theory, a proportional controller is enough to reach our goal. To obtain the controller gain, let us follow the steps of the precedent procedure. The characteristic equation is:

$$1 + K_p \frac{5}{s(s+1)} = 0$$
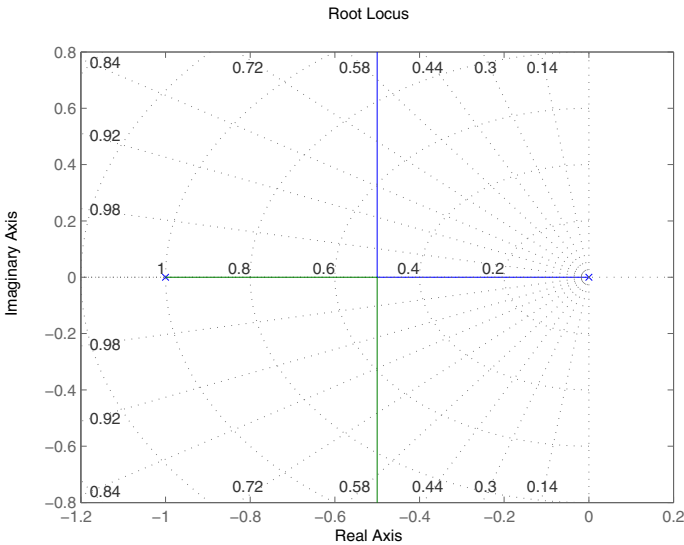
Fig. 5.8 Root locus of $\frac{1}{s(s+1)}$

The root locus of the closed-loop dynamics is given by Fig. 5.8 and from which we get:

$$s_d = -0.5 + j0.5$$

and the corresponding gain is $K = 0.5$. *This gives the following gain for the controller:*

$$K_P = \frac{K}{5} = 0.1$$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.9. The simulation results show the efficiency of the designed controller. The closed-loop dynamics is stable and the overshoot is less than 5 % as it is expected.
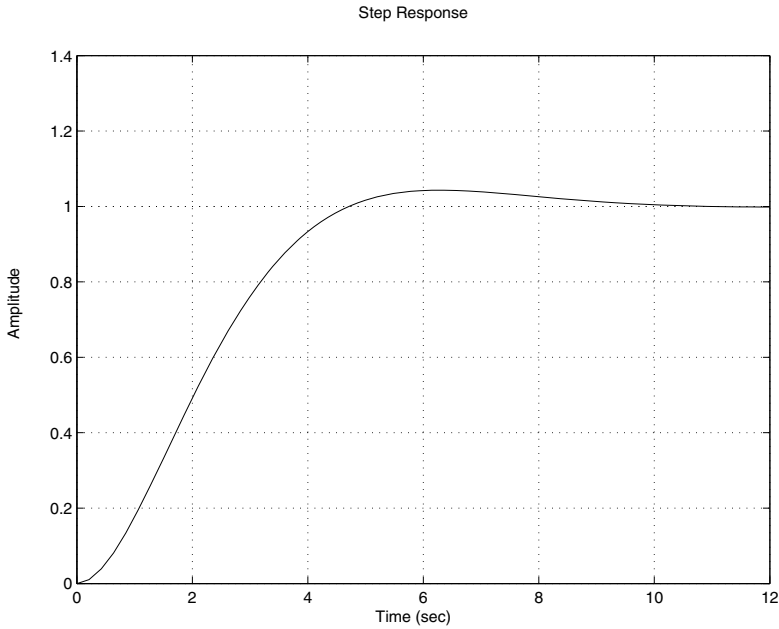
Fig. 5.9 Step response of $\frac{0.5}{s(s+1)+0.5}$

Remark 5.4.1 *It is important to notice that the best setting time at* 2 % *that we can get with this type of controller*

$$t_s = \frac{4}{\zeta w_n} = \frac{4}{0.5} = 8 \ sec$$

where ζ *and* w_n *are respectively the damping ratio and the natural frequency of the closed-loop dynamics. This can be checked from the Fig. 5.9.*

For the design of a proportional and integral controller the same technique can be used. If we let the transfer function of this controller be as follows:

$$C(s) = K_P + \frac{K_I}{s} = K_P \frac{s+z}{s}, z = \frac{K_I}{K_P}$$

where the gains K_P and K_I have to be determined.

This controller may be used to act simultaneously on the transient and the steady state regimes and therefore overcomes what the proportional controller alone can not perform. Most often the proportional and integral controller is used to make the error equal to zero for a step input and fix the overshoot and the settling time. The following procedure can be used (see Boukas [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d

2. using this pole and the angle condition, we can determine the angle of the controller's zero, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n+1} \angle(s_d + p_i)$$

The value of the zero is then given by:

$$z = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta \omega_n}$ if the settling is fixed at 5 %

3. plot the loci of

$$K \frac{s+z}{s} \frac{\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = \frac{\Pi_{i=1}^{n+1}|(s_d + p_i)|}{\Pi_{i=1}^{m+1}|(s_d + z_i)|}$$

4. the controller gains are given by:

$$K_P = \frac{K}{k}$$
$$K_I = z K_P$$

To obtain the corresponding discrete-time transfer function we can use one of the approaches presented earlier. The third approach (trapezoidal method) is used here and it consists of replacing s by $\frac{2}{T}\frac{z-1}{z+1}$, where T is the chosen sampling period. Using this we get:

$$C(z) = \frac{\left(\frac{K_I T}{2} + K_P\right)z + \frac{K_I T}{2} - K_P}{z - 1}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = u_{k-1} + a e_k + b e_{k-1} \tag{5.2}$$

where $a = \frac{K_I T}{2} + K_P$ and $b = \frac{K_I T}{2} - K_P$

The lines that we should include during the implementation in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.2 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in speed.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{\tau s + 1}$$

with k = 5, and τ = 1s.

From basic control theory, we can see that the settling time of the open-loop system with 5 % is t_s = 3τ = 3s. The system doesn't have an overshoot and the response is a little bit slow.

Our desire is to make the system faster with an overshoot less or equal to 5 %, a settling time t_s at 5% less or equal to 1s, and a steady state error for a step input equal to zero.

To solve this design problem, let us proceed in continuous-time domain. For this purpose, let us first of all mention that the type of the system is equal to zero and therefore to guarantee that the error is equal to zero at the steady state for a step input, we need at least a proportional and integral controller.

Following the procedure of the proportional integral controller we have:

1. *the dominant pole with the positive imaginary value is given by:*

$$\begin{aligned} s_d &= -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2} \\ &= -3 + 3j \end{aligned}$$

This comes from the fact that we have:

$$\zeta = -\frac{\log\left(\frac{d}{100}\right)}{\sqrt{\pi^2 + \left(\log\left(\frac{d}{100}\right)\right)^2}} = 0.6901$$

$$\omega_n = \frac{3}{\zeta t_s} = 4.3472$$

2. *using this pole, we get:*

$$\begin{aligned} \alpha &= \pi - 0 + \angle(-3 + 3j) + \angle(-2 + 3j) \\ &= 180 + 135 + 123.6901 \\ &= 78.6901 \end{aligned}$$

which gives the following value for the zero

$$z = -3 - \frac{3}{\tan(78.6901)}$$
$$= -3.6$$

3. *the loci of the controlled system is given by Fig. 5.10, from which we conclude that K = 4.73.*

4. *the controller gains are:*

$$K_P = 0.9460$$
$$K_I = 3.4056$$

The root locus of the system is illustrated in Fig. 5.10.
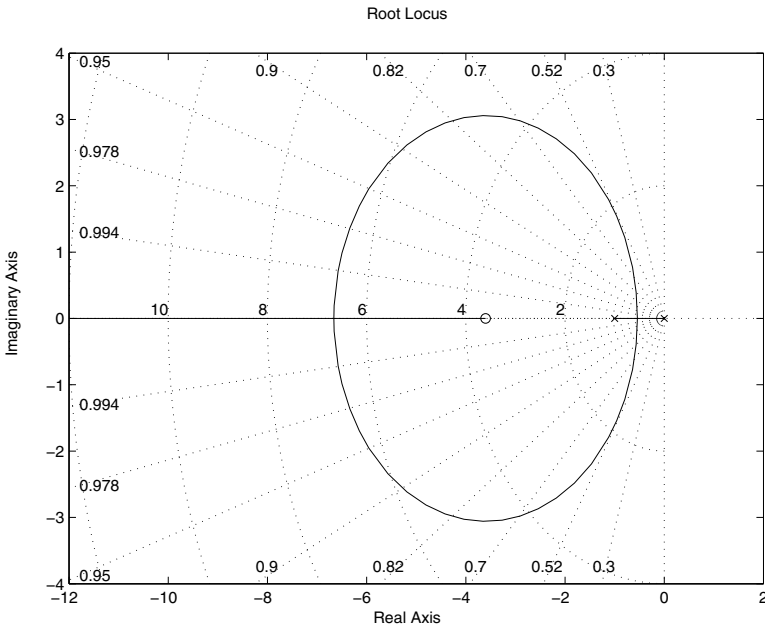
Fig. 5.10 Root locus of $\frac{s+z}{s(s+1)}$, $z = -3.6$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.11. The simulation results show that the overshoot is over what we desire while the settling time is acceptable. To reduce the overshoot, we can redo the design by pushing a little bit the zero to the left and get new set of gains for the controller.

In some circumstances, we may have a system that has acceptable steady regime but the transient one needs some improvements. In this case the proportional and
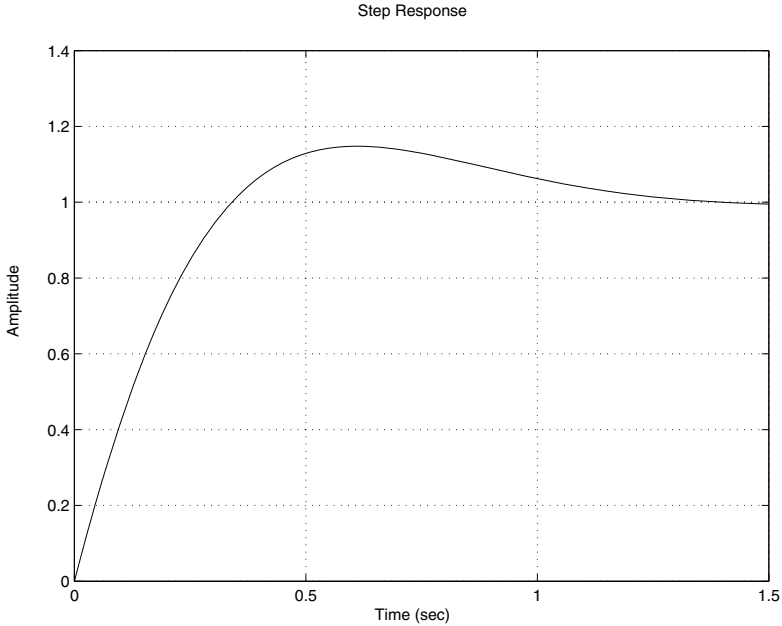
Fig. 5.11 Step response of $\frac{5K_P s + 5K_I}{s^2 + (1+5K_P)s + 5K_I}$

derivative controller can be used. The transfer function of this controller is given by:

$$C(s) = K_P + K_D s$$

where K_P and K_D are the controller gains that we should compute in order to satisfy the desired performances.

To design the proportional and derivative controller the following procedure can be used (see [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d

2. using this pole and the angle condition, we can determine the angle of the controller zero as it was done for the proportional and integral controller previously, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n} \angle(s_d + p_i)$$

The value of the zero is then given by:

$$z = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta \omega_n}$ if the settling fixed at 5 %

3. plot the loci of

$$K\frac{(s+z)\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = \frac{\Pi_{i=1}^{n}|(s_d+p_i)|}{\Pi_{i=1}^{m+1}|(s_d+z_i)|}$$

4. the controller gains are given by:

$$K_D = \frac{K}{k}$$
$$K_P = zK_D$$

The corresponding discrete-time version of this controller, using the same transformation as for the proportional and integral controller, is given by:

$$C(z) = \frac{(K_P + \frac{2K_D}{T})z + (K_P - \frac{2K_D}{T})}{z+1}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = -u_{k-1} + ae_k + be_{k-1} \qquad (5.3)$$

where $a = K_P + \frac{2K_D}{T}$ and $b = K_P - \frac{2K_D}{T}$.

Remark 5.4.2 *For this controller the backward approach is more appropriate for the derivative action and the trapezoidal for the integral action. In tis case we get:*

$$u(k) = ae(k) + be(k-1)$$

with $a = K_P + \frac{K_D}{T_s}$ and $b = -\frac{K_D}{T_s}$.

The lines that we should include in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.3 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in position.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{s(\tau s + 1)}$$

with $k = 5$, and $\tau = 1s$.

The settling time of the system with 5 % is $t_s = 3\tau = 3s$. The system is unstable. Our desire is to make the system stable and faster with an overshoot less or equal to 5 % and a settling time t_s at 5% less or equal to 0.5s.

To solve this design problem, let us proceed in continuous-time domain. For this purpose, let us first of all mention that the type of the system is equal to one and therefore the error is equal to zero at the steady state for a step input. For the transient to be improve we need at least a proportional controller but here we use a proportional and derivative controller to get better settling time.

Following the previous procedure we have:

1. *the dominant pole with the positive imaginary value is given by:*

$$s_d = -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2}$$
$$= -6 + 6j$$

2. *using this pole, we get:*

$$\alpha = \pi + \angle(-6+6j) + \angle(-5+6j)$$
$$= 180 + 135 + 129.8056$$
$$= 84.8056$$

   which give the following value for the zero

$$z = -6 - \frac{3}{\tan(84.8056)}$$
$$= -6.7273$$

3. *the loci of the controlled system is given by Fig. 5.12, from which we conclude that $K = 10.8$.*

4. *the controller gains are:*

$$K_D = 2.1600$$
$$K_P = 14.5310$$

The simulation results illustrated in Fig. 5.13 show that we have an overshoot greater than the one we need but the settling time is acceptable. To reduce the overshoot we can move the zero a little bit to left and get new set of gains for the controller.

For some systems, we need to improve simultaneously the transient and the steady regimes. In this case, the most appropriate choice is the proportional, integral and derivative controller. Its transfer function is given by:

$$C(s) = K_P + \frac{K_I}{s} + K_D s$$
$$= K_D \frac{(s+a_1)(s+a_2)}{s}$$

with $K_P = K_D(a_1 + a_2)$ and $K_I = K_D a_1 a_2$

To design the proportional, integral and derivative controller, the following procedure can be used (see [1]):
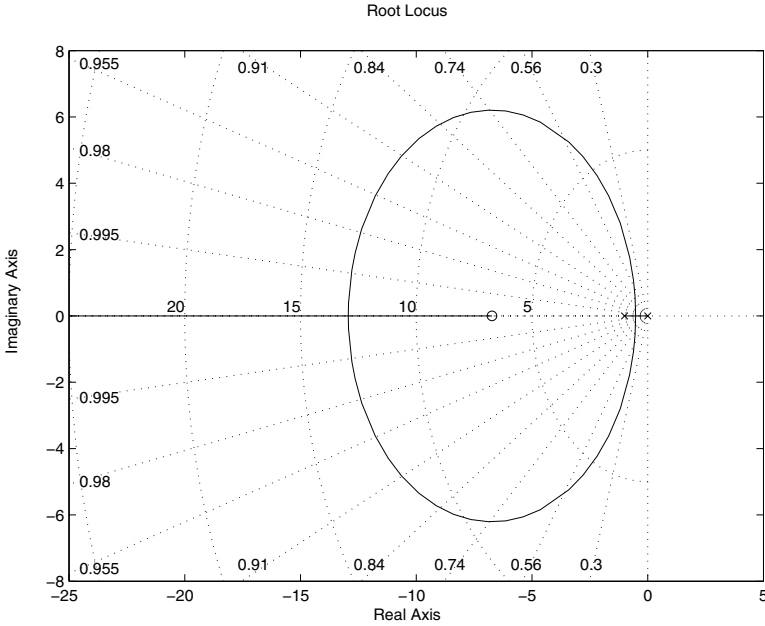
Fig. 5.12 Root locus of $\frac{s+z}{s(s+1)}$, $z = 6.7273$

1. to determine the parameter a_1 (that is related to the first zero), we choose the slow pole and proceed with a zero-pole cancelation. If we denote by $-p_s$ the slow pole, the parameter a_1 is then given by:

$$a_1 = p_s$$

2. with the damping ratio value and the settling time, we can determine the dominant pole with the positive imaginary part, s_d

3. using this pole and the angle condition, we can determine the angle of the controller zero that corresponds to $s + a_2$, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n} \angle(s_d + p_i)$$

Remark 5.4.3 *It is important to notice that a pole has been cancelled by the zero at position $-a_1$ and a new pole at 0 has been added to the equation.*

The value of the second zero is then given by:

$$a_2 = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta\omega_n}$ if the settling fixed at 5 %
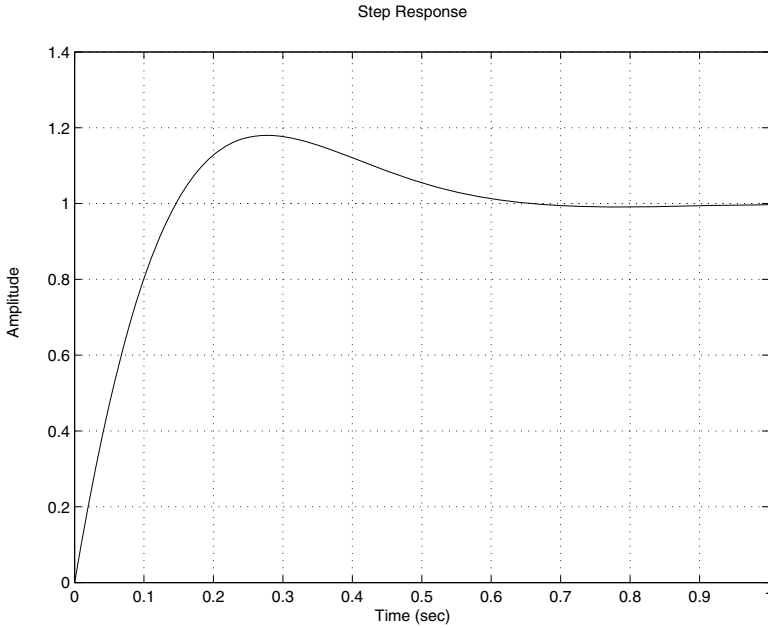
Fig. 5.13 Step response of $\frac{s+z}{s(s+1)}$, $z = 6.7273$

4. plot the loci of

$$K \frac{(s + a_2)\Pi_{i=1}^{m}(s + z_i)}{s\Pi_{i=1}^{n-1}(s + p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = |s_d| \frac{\Pi_{i=1}^{n-1}|s_d + p_i|}{|s_d + a_2|\Pi_{i=1}^{m}|s_d + z_i|}$$

5. the controller gains are given by:

$$K_D = \frac{K}{k}$$
$$K_P = K_D(a_1 + a_2)$$
$$K_I = K_D a_1 a_2$$

The corresponding discrete-time version of this controller, using the same transformation as for the proportional and integral controller, is given by:

$$C(z) = \frac{K_P z^2 + \left(\frac{K_I T}{2} + \frac{2K_D}{T}\right)z + \left(\left(\frac{K_I T}{2} + \frac{2K_D}{T}\right) - K_P\right)}{(z - 1)(z + 1)}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = u_{k-2} + ae_k + be_{k-1} + ce_{k-2} \qquad (5.4)$$

where $a = K_P + \frac{K_I T}{2} + \frac{2K_D}{T}$, $b = K_I T - \frac{4K_D}{T}$ and $c = \frac{K_I T}{2} + \frac{2K_D}{T} - K_P$.

Remark 5.4.4 *As we said regarding the schema for the discretization of the controller we can use here also the trapezoidal schema for the integral action and the backward schema for the derivative one. In this case we get:*

$$u(k) = u(k - 1) + ae(k) + be(k - 1) + ce(k - 2)$$

with $a = K_P + \frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $b = -K_P - 2\frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $c = \frac{K_D}{T_s}$;

The lines that we should include in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.4 *To show how this procedure can be used to design a proportional, integral and derivative controller, let us consider the following system with:*

$$G(s) = \frac{3}{(s + 1)(s + 3)}$$

For this system we would like to guarantee that the steady state error for a step input is equal to zero with an overshoot less or equal to 5 % and a settling time about $1s$. Following the steps of the previous procedure we get:

1. *the slow pole in this case is equal to -1 and therefore, the parameter is then $a_1 = 1$.*

2. *the dominant pole with the positive imaginary value is given by:*

$$\begin{aligned} s_d &= -\zeta\omega_n + j\omega_n\sqrt{1 - \zeta^2} \\ &= -3 + 3j \end{aligned}$$

3. *using this pole, we get since the pole -1 has been cancelled with the zero at $-a_1$:*

$$\begin{aligned} \alpha &= \pi + \angle(-3 + 3j) + \angle(3j) \\ &= 180 + 135 + 90 \\ &= 45 \end{aligned}$$

   which give the following value for the zero

$$\begin{aligned} a_2 &= -3 + \frac{3}{\tan(45)} \\ &= -6 \end{aligned}$$

4. *the loci of the controlled system is given by Fig. 5.14, from which we conclude that $K = 2.99$ is the appropriate one that give the closest dominant poles and the damping ratio ($s_d = -2.99 \pm 2.99j$, $\zeta = 0.707$, $d = 4.51 \% w_n = 4.23$).*
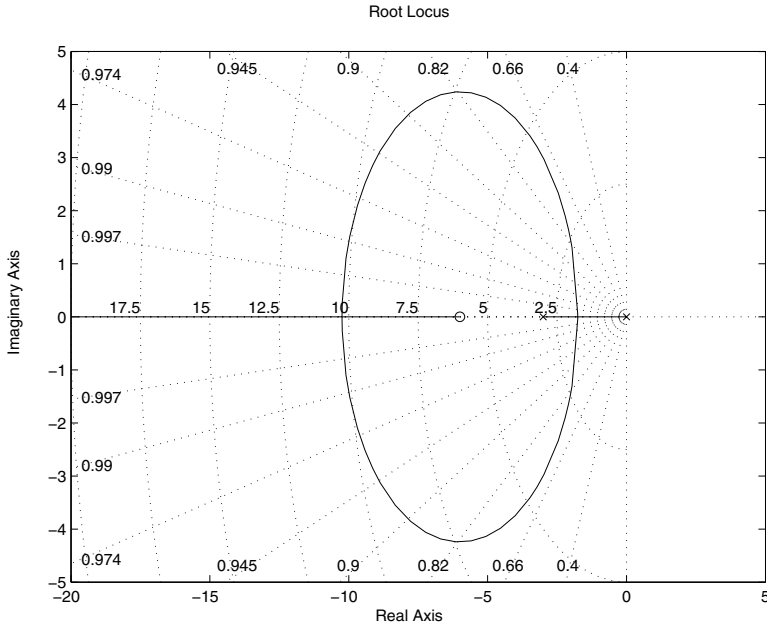
Fig. 5.14 Root locus of $\frac{s+a_2}{s(s+3)}$, $a_2 = 6$

5. *the controller gains are:*

$$K_D = 0.9967$$
$$K_P = 6.9769$$
$$K_I = 5.9802$$

The closed-loop transfer function is given by:

$$F(s) = \frac{kK_D s^2 + kK_P s + kK_I}{s^3 + (4 + kK_D)\, s^2 + (3 + kK_P)\, s + kK_I}$$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.15
The simulation results show the efficiency of the designed controller.

The phase lead controller can be used to approximate the proportional and derivative one. The transfer function of this controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}$$

where K_P, a and T are parameters to be computed with $a > 1$.

This controller offers the advantage to improve the transient regime. This can be obtained if the placement of the pair pole/zero is well positioned since we can pull the asymptotic branches to get a smaller settling time.
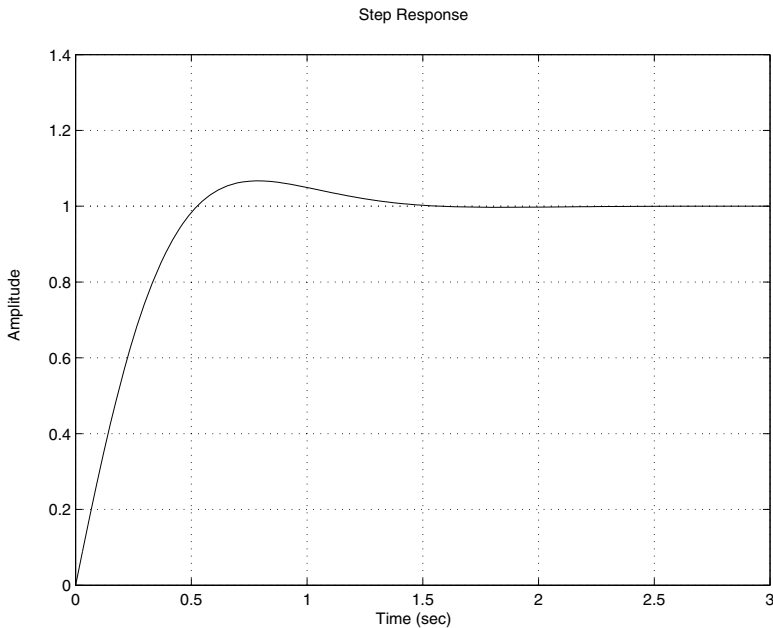
Step Response

Fig. 5.15 Step response of $\frac{s+a_2}{s(s+3)}$, $a_2 = 6$

The following procedure can be used to design such controller (see [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d

2. by varying the system gain, try to get the desired dominant poles, if it is not possible, determine the contribution in angle of the pair pole/zero that the controller has to add

3. place the pole and the zero of the phase lead controller in order to compensate for this desired angle. Among the possibilities that can be used in this case, we can place the zero at a value equal to the real part of the dominant poles and then using the angle condition we can determine the pole position.

4. determine the value for the controller gain in order to satisfy the error

5. check if the desired specifications are obtained. In case of negative answer replace the pair pole/zero of the controller and repeat the design procedure

Example 5.4.5 *To show how the procedure of the phase lead controller can be applied, let us consider the position control for a dc motor driving a mechanical load as it was considered before. Let the dynamics be given by:*

$$G(s) = \frac{2}{s\,(s+2)}.$$

Our goal in this example is to guarantee that the closed-loop system is stable, with a settling time at 5% equal to 0.5 s, an overshoot less or equal to 5% and having a zero error for a step input.

First of all notice that the time constant of the system is equal to 0.5 s which may give the best settling time at 5 % with a proportional controller equal to 6 s. Our requirement in regard to the settling time is far from this value and therefore a proportional controller is not sufficient for our case.

To respond to these specifications a phase lag controller can be used and its design can be done using the previous procedure.

1. *based on the settling time and the overshoot requirements we get the following dominant pole with positive imaginary value:*

$$s_d = -6 + 6j$$

This desired poles can not be obtained by varying the gain of a proportional controller and therefore a design of a phase lead controller is needed. From this value for the dominant pole, we have:

$$\angle G(s_d) = \angle(2) - \angle(-6 + 6j) - \angle(-5 + 6j)$$
$$= 0 - 135 - 123.6901 = -258.6901$$

The controller can be designed to bring an angle $258.6901 - 180 = 78.6901$. This is obtained if $\angle(aTs + 1) - \angle Ts + 1 = 78.6901$

2. *following the method we used in the procedure, we get $aT = \frac{1}{6}$ and therefore $\angle(Ts + 1) = 90 - 78.6901 = 11.3099$. This gives the location of the controller pole. Using now the following trigonometric relation we get:*

$$\tan(11.3099) = \frac{\Im(s_d)}{\frac{1}{T} - |\Re(s_d)|}$$

which gives $T = 0.0278$. This in turn implies that $a = \frac{1}{6T} = 5.9952$.

3. *The open-loop transfer function of the compensated system is then given by:*

$$G_c(s) = K\frac{(s + 6)}{s(s + 2)(s + 35.9712)}$$

which gives the following gain, K that corresponds to the desired pole s_d:

$$K = \frac{|s_d||s_d + 2||s_d + 35.9712||}{|s_d + 6|} = 311.7120$$

The corresponding controller gain is $K_P = \frac{K}{ak} = 25.9968$. The root locus of the compensated system is illustrated by Fig. 5.16.

The closed-loop transfer function with this controller is given by:

$$F(s) = \frac{2aK_P\left(s + \frac{1}{aT}\right)}{s^3 + \left(2 + \frac{1}{T}\right)s^2 + \left(\frac{2}{T} + 2aK_P\right)s + \frac{2K_P}{T}}$$

Fig. 5.16 Root locus of $\dfrac{s+\frac{1}{aT}}{s(s+2)(s+\frac{1}{T})}$

*The behavior of the closed-loop dynamics is illustrated in Fig. 5.17
The simulation results show the efficiency of the designed controller. It is clear
that the performances are a little bit far from the desired ones. This is due to the
place of the zero of the controller that we can from see from Fig. 5.16. We can
play with this position by pushing it to the left and we will get what we want.*

Remark 5.4.5 *It is important to notice that phase lead controller or the phase lag
or the phase lead-lag controllers are not able to to make the error equal to zero
since they can't improve the type the system. But they can improve it if it is constant.*

The phase lag controller can be used to approximate the proportional and integral
one. Its task is to improve the steady state regime if it is well designed. The pair
pole/zero of the controller is put close to the origin. The transfer function of this
controller is given by:

$$C(s) = K_p \frac{aTs + 1}{Ts + 1}$$

where K_P, a and T are parameters to be computed with $a < 1$.
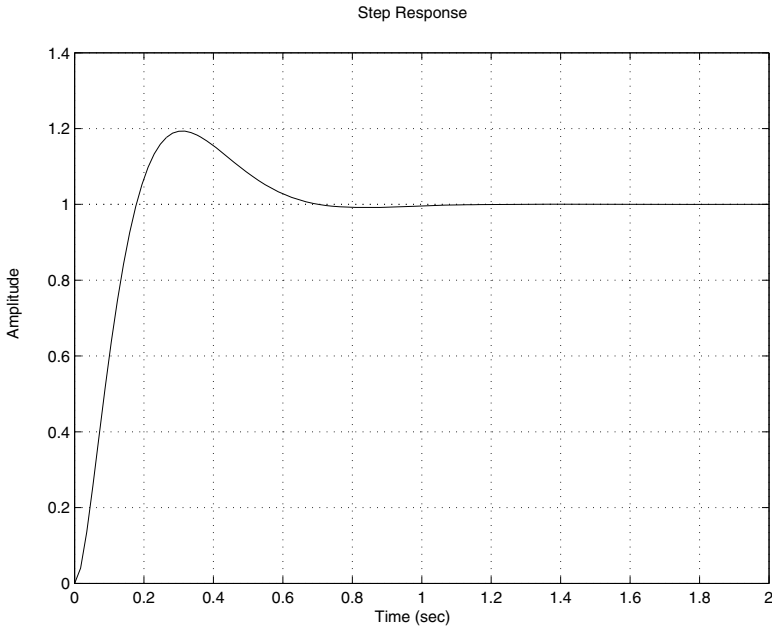
Step Response

Fig. 5.17 Step response of $F(s) = \dfrac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3+\left(2+\frac{1}{T}\right)s^2+\left(\frac{2}{T}+2aK_P\right)s+\frac{2K_P}{T}}$

To have an idea of the design approach let us assume that the system to be controller is described by:

$$G(s) = k\frac{\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

where k is the gain, $-z_i, i = 1, \cdots, m$ and $-p_i, i = 1, \cdots, n$ are respectively the zeros and the poles of the system.

In fact, if we write the transfer function of the controller as:

$$C(s) = K_P\frac{s+z}{s+p}$$

with $K_P = aK_p$, $z = \frac{1}{aT}$ and $p = \frac{1}{T}$.

With the gain of the controller only, the constant error is given by:

$$K_1 = kK_P\frac{\Pi_{i=1}^{m}z_i}{\Pi_{i=1}^{n}p_i}$$

In order to improve the steady state error, we would get a constant error, K_2, greater than K_1. By introducing the zero and the pole of the controller this constant error is given by:

$$K_2 = kK_P\frac{z}{p}\frac{\Pi_{i=1}^{m}z_i}{\Pi_{i=1}^{n}p_i}$$

Our desire is that the new pair of pole/zero of the controller doesn't change the transient regime which is acceptable for the designer and the main goal is to change the steady state regime only by reducing the error.

Using the expressions of K_1 and K_2, we get:

$$\frac{K_1}{\frac{\Pi_{i=1}^m z_i}{\Pi_{i=1}^n p_i}} = kK_P = \frac{K_2}{\frac{z}{p}\frac{\Pi_{i=1}^m z_i}{\Pi_{i=1}^n p_i}}$$

This implies that:

$$a = \frac{p}{z} = \frac{K_1}{K_2} < 1$$

Therefore, if we choose T in a way that the pole and the zero are close each other (to be cancelled in the open transfer function of the system), the open loop transfer function of the controlled system becomes:

$$C(s)G(s) = kK_P \frac{\Pi_{i=1}^m (s + z_i)}{\Pi_{i=1}^n (s + p_i)}$$

The idea we will use here is mainly based on the improvement of the steady state error. The following procedure can be used to design such controller (see [1]):

1. with the damping ratio and the settling time values, we can determine the pole dominant with the positive imaginary part, s_d and determine the gain that gives such poles. Compute the corresponding constant error.

2. determine the constant error, K_1, with a proportional controller. Determine the constant error, K_2 when the pole and the zero of the controller are considered. The parameter a of the controller is given by:

$$a = \frac{K_1}{K_2}$$

   This parameter, a is also given by:

$$a = \frac{p}{z}$$

3. the value for T is chosen in a way to make the pole and the zero of the controller are close each other and at the same time close to the origin to improve the steady error. This choice will imply that the angle contribution of the controller is very small.

4. determine the gain, $\bar{K}_P$, using the following relation:

$$\bar{K}_P = \left[\frac{|s_d + p|}{|s_d + z|}\right]\left[\frac{\Pi_{i=1}^n |s_d + p_i|}{\Pi_{i=1}^n |s_d + z_i|}\right]$$

   then detrmine the controller gain, K_P by:

$$K_P = \frac{\bar{K}_P}{ak}$$

5. check if the specifications are similar to the desired ones. In the case of negative answer adjust the placement of the pole and the zero of the controller and repeat the procedure

Example 5.4.6 *To show how the procedure of the phase lag controller design can be applied, let us consider the position control for a dc motor driving a mechanical load as it was considered before. Let the dynamics be given by:*

$$G(s) = \frac{2}{s\,(s+2)}.$$

Our goal in this example is to guarantee that the closed-loop system is stable, with a settling time at 5% equal to 3 s, an overshoot less or equal to 5% and having an error for a ramp input less or equal to 0.01.

To respond to these specifications a phase lag controller can be used and its design can be done using the previous procedure.

1. *based on the settling time and the overshoot requirements we get the following dominant pole with positive imaginary value:*

$$s_d = -1 + 1j$$

The root locus the system with a proportional controller is given by Fig. 5.18.
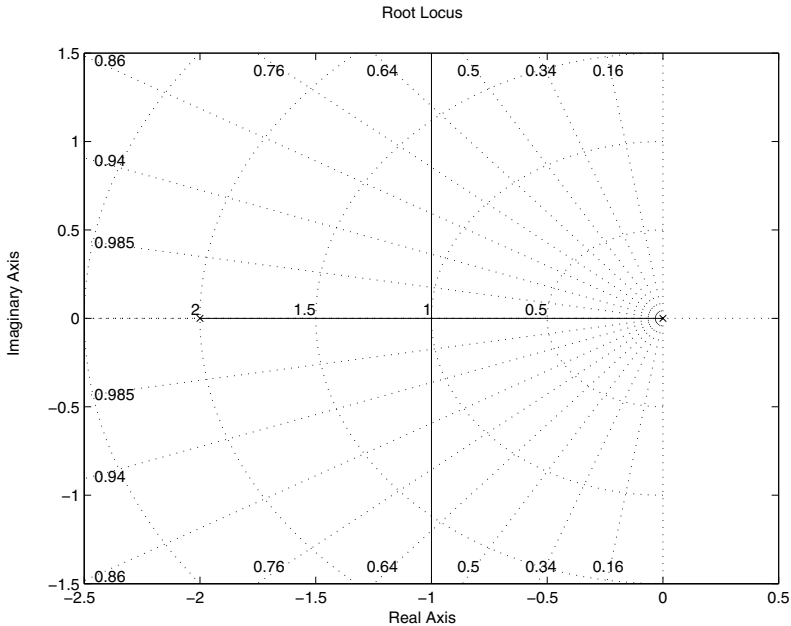
Fig. 5.18 Root locus of $\frac{1}{s(s+2)}$

The gain that gives this pair of poles is given by:

$$K_1 = \frac{|s_d||s_d + 2|}{1} = 2.0$$

This correspond to an error equal:

$$e(\infty) = \frac{1}{2} = 0.5$$

which far from the desired one.

2. *To get our error we need a constant K_2 equal to 100. This implies that the factor a of the controller is given by:*

$$a = \frac{K_1}{K_2} = \frac{2}{100} = 0.02 = \frac{p}{z}$$

3. *since the pole and the zero of the controller have to be placed closed to each other and close to the origin. If we place the zero at -0.3 which a pole for the controller at -0.006 using the fact that $a = \frac{p}{z}$. The value of T can be computed using either the expression of the zero of the one of the pole. This gives $T = 166.6667$.*

4. *The open-loop transfer function of the compensated system is then given by:*

$$G_c(s) = K \frac{(s + 0.3)}{s(s + 2)(s + 0.006)}$$

which gives the following gain, K that corresponds to the desired pole s_d:

$$K = \frac{|s_d|s_d + 2||s_d + 0.01||}{|s_d + 0.5|} = 2.3102$$

The corresponding controller gain is $K_P = \frac{K}{ak} = 57.7549$. The root locus the system with a proportional controller is given by Fig. 5.19.
The closed-loop transfer function with this controller is given by:

$$F(s) = \frac{2aK_P \left(s + \frac{1}{aT}\right)}{s^3 + \left(2 + \frac{1}{T}\right)s^2 + \left(\frac{2}{T} + 2aK_P\right)s + \frac{2K_P}{T}}$$
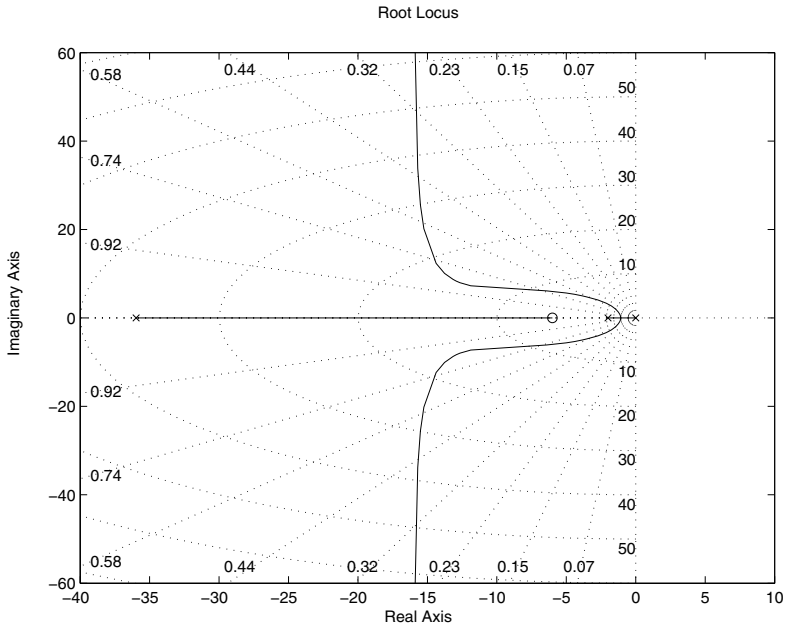
The behavior of the closed-loop dynamics is illustrated in Fig. 5.20
The root locus of the compensated system unfortunately doesn't pass through the desired poles. The closed ones are $s_d = -0.8082 \pm 1.14j$ that corresponds to a gain $K = 2.56$, that gives a gain $K_P = 64$. With this gain we get an overshoot approximately equal to 11 %.
The behavior of the closed-loop dynamics with this new setting is illustrated in Fig. 5.21

Remark 5.4.6 *It is important to notice that the overshoot is a little bit far from the desired one and it is the same for the settling time. This discrepancy is due to the presence of the zero that he introduce high overshot once it is close to the origin.*
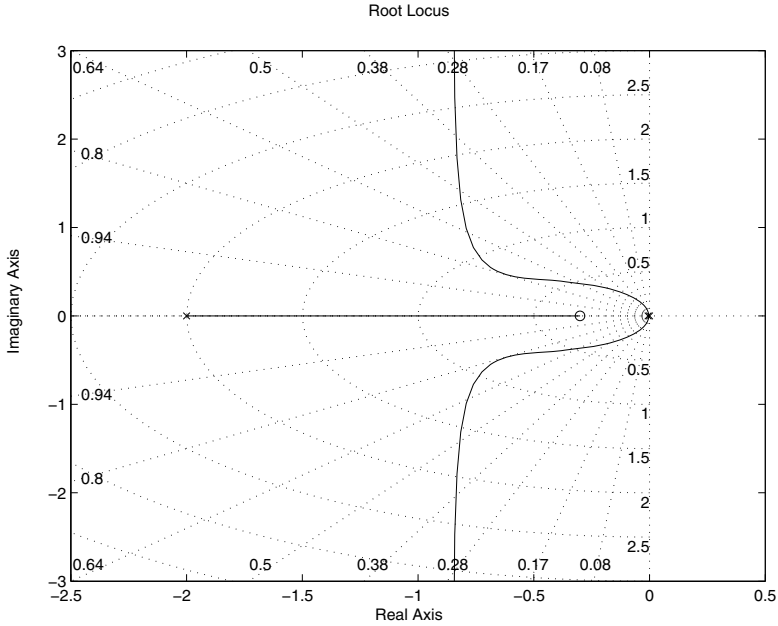
Fig. 5.19 Root locus of $\frac{s+0.3}{s(s+2)(s+0.06)}$

And also due to the fact the cancellation pole zero of the controller is not correct since the pole is a little bit far from the zero.

The phase lead-lag controller is designed to approximate the PID controller. It has the advantage as the PID has to act on both the transient and the steady regimes. Previously we have seen how to design the phase lead controller and the phase lag controller. The first one is used to act on the transient while the second acts of the steady state regime.

The transfer function of this controller is given by:

$$C(s) = K_P \frac{s + \frac{1}{a_1 T_1}}{s + \frac{1}{T_1}} \frac{s + \frac{1}{a_2 T_2}}{s + \frac{1}{T_2}}$$

where K_P is the controller gain, a_1 with $a_1 > 1$ and T_1 are the parameter of the lead part, while a_2 with $a_2 < 1$ and T_2 are the parameter of the phase lag part.

To design such controller, we use the approaches used to design separately the phase lead and the phase lag controller. First, without the phase lag controller, we design the phase lead controller to improve the transient regime. After, we add the phase lag controller to improve the steady state regime while keeping the transient regime as it was improved by the phase lead controller.

The following procedure procedure can be used to design the phase lead-lag controller:
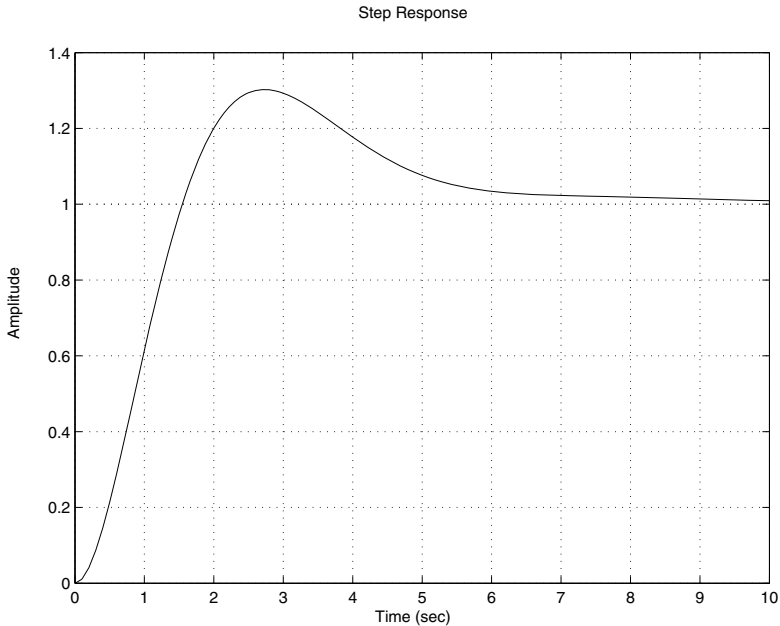
Step Response

Fig. 5.20 Step response of $F(s) = \dfrac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3+\left(2+\frac{1}{T}\right)s^2+\left(\frac{2}{T}+2aK_P\right)s+\frac{2K_P}{T}}$

1. without the phase lead-lag controller, see if with a proportional controller, we can guarantee the desired performances. Analyze the system with a proportional controller and determine how much the transient regime has to be improved

2. design the phase lead controller (gain, pole and zero)

3. analyze the compensated system with a phase lead controller and determine how much the steady state regime has to be improved

4. design the phase lag controller (gain, pole and zero)

5. check if the specifications are similar to the desired ones. In the case of negative answer adjust the placement of the pole and the zero of the controller and repeat the procedure

Example 5.4.7 *To see how the procedure for the design of phase lead-lag controller applies, let us consider the following dynamical system:*

$$G(s) \;=\; \frac{2}{s(s + 2)}$$

For this system with a proportional controller, the best settling time at 5 % we can obtain is equal to 3 s. We can also get an overshot less or equal to 5 %. The steady state error for a step input is equal to zero, while the one for a ramp is constant and
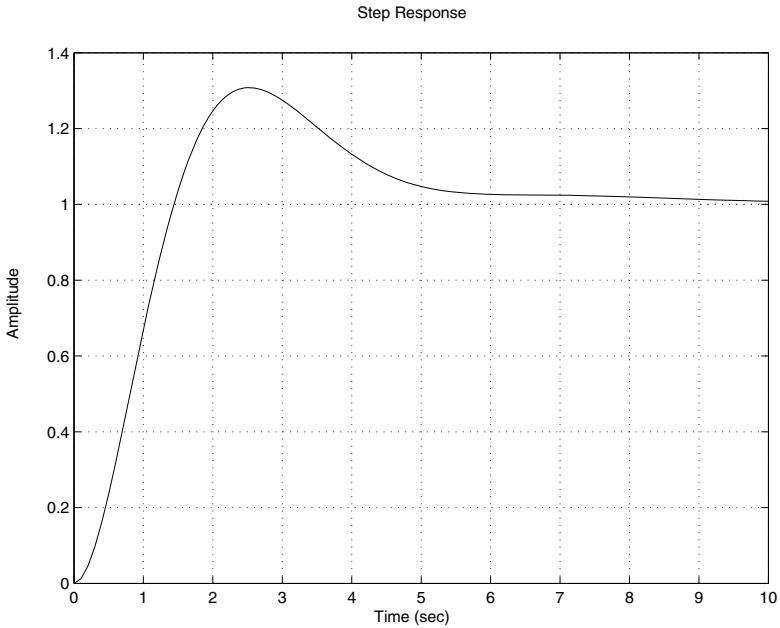
Step Response

Fig. 5.21 Step response of $F(s) = \dfrac{2aK_P\left(s + \frac{1}{aT}\right)}{s^3 + \left(2 + \frac{1}{T}\right)s^2 + \left(\frac{2}{T} + 2aK_P\right)s + \frac{2K_P}{T}}$

can be fixed by acting on the gain controller. A "trade-off" between the overshoot and the steady state error has to be done. It is clear that the proportional controller will not give the good trade-off. The phase lead-lag controller will give the better one.

For this purpose let us assume that we desire the following specifications:

1. stable system in closed loop

2. an overshoot less or equal to 5 %

3. a settling time at 5 % about 2 s

4. a steady state error for a ramp input less or equal to 0.01

To design the phase lead-lag controller that provides the desired performances, let us follow the previous procedure:

1. From Fig. 5.18, it is clear that the settling time requirement can be obtained using a proportional controller. From the specifications, we get the dominant pair of poles that gives what we are looking for:

$$s_d = -1.5 + 1.5j$$

This desired pair of poles can not be obtained by just varying the gain of the proportional controller. A phase lead controller is needed for this purpose. The phase of the transfer function at s_d is given by:

$$\angle G(s_d) = \angle(2) - \angle(0.5000 + 1.5000j) - \angle(-1.5000 + 1.5000j)$$
$$= 0 - 71.5651 - 135 = 206.5651$$

The controller phase lead controller can be used to bring the angle contribution of $206.5651 - 180 = 26.5651$. This can be obtained if we impose that $\angle(s_d + \frac{1}{a_1 T_1}) - \angle(s_d + \frac{1}{T_1}) = 26.5651$.
Following the procedure of the phase lead controller design, if we impose that the zero of the controller is place at the real part of the dominant poles, we get:

$$a_1 T_1 = \frac{1}{1.5}$$

and therefore, we have:

$$\angle(s_d + \frac{1}{T}) = 90 - 26.5651 = 63.4349$$

To get the position of the pole, i.e:, we use the following trigonometric relation:

$$\frac{1}{T_1} = |\Re(s_d)| + \frac{\Im(s_d)}{\tan(11.3099)} = 2.2500$$

which implies $T_1 = 0.4444$. And from the relation $a_1 T_1 = \frac{1}{1.5}$, we get $a_1 = 1.5002$.

2. *for the design of the phase lag controller, notice that the compensated system with a phase lead controller has the following open transfer function:*

$$G_1(s) = a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)}$$

with $k = 2$.
The root locus of this transfer function is illustrated by Fig. 5.22
The gain K_1 that gives the poles that are close to the desired poles is given by:

$$K_1 = 3.87$$

The corresponding poles are $s_d = -1.5 \pm 1.54j$ with an overshoot approximatively equal to 5 %.
The error constant with this controller is given by:

$$K_2 = \lim_{s \to 0} s a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)} = a_1 k K_p \frac{1}{a_1 T_1} \frac{T_1}{2} = K_P$$
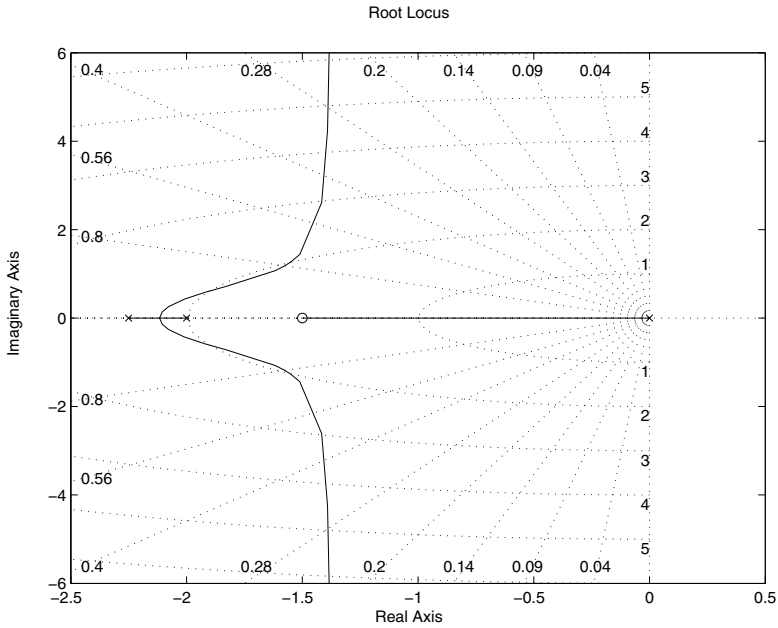
Fig. 5.22 Root locus of $\dfrac{s+\frac{1}{a_1 T_1}}{s(s+2)\left(s+\frac{1}{T_1}\right)}$

To get the desired error we need to fix K_p to 100. This gives the following parameter, a_2 for the phase lag controller:

$$a_2 \;=\; \frac{K_1}{K_2} \;=\; \frac{3.87}{100} = 0.0387$$

Since the procedure for the design of the phase lag controller requires that we have to place the pole and the zero of the controller close each other and close to the origin. A proper choice consists of placing the zero at -0.1. This implies using the relation

$$a_2 \;=\; \frac{p}{z}$$

that the pole is placed at $p = -0.0039$ and since the pole is equal to:

$$p \;=\; -\frac{1}{T_2}$$

we get $T_2 = 256.4103$.

The open loop transfer function of the system with the phase lead-lag controller is give by:

$$G_2(s) \;=\; a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)}$$

with $k = 2$.

The root locus of this transfer function is illustrated by Fig. 5.23
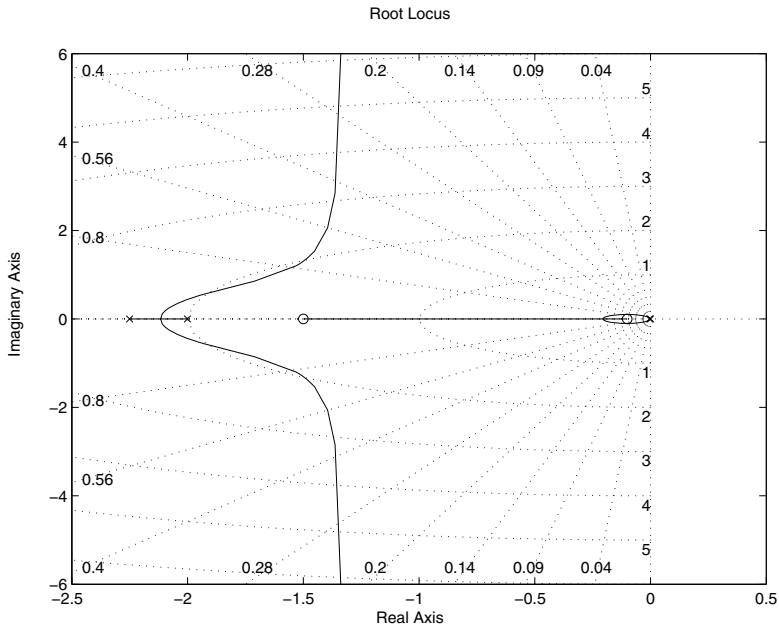
Fig. 5.23 Root locus of $\dfrac{\left(s+\frac{1}{a_1 T_1}\right)\left(s+\frac{1}{a_2 T_2}\right)}{s(s+2)\left(s+\frac{1}{T_1}\right)\left(s+\frac{1}{T_2}\right)}$

From this figure we get the closest poles of the desired ones are:

$$s_d = -1.5 \pm 1.31 j$$

that gives an damping ratio about 0.753 and an overshoot equal to 2.73 %. The gain that gives this pair of poles is equal to:

$$\bar{K}_P = 3.35$$

From this data, we get the following gain for the controller:

$$K_P = \frac{\bar{K}_P}{a_1 a_2 k} = \frac{3.35}{2 \times 1.5002 \times 0.0387} = 28.8506$$

The expression of the designed controller is given by:

$$C(s) = K_P \frac{s + \frac{1}{a_1 T_1}}{s + \frac{1}{T_1}} \frac{s + \frac{1}{a_2 T_2}}{s + \frac{1}{T_2}}$$

with $K_P = 28.8506$, $a_1 = 1.5002$, $T_1 = 0.4444$, $a_2 = 0.0387$ and $T_2 = 256.4103$.

The closed-loop transfer function with this controller is given by:

$$F(s) \; = \; \frac{kK_P\left(a_1T_1a_2T_2s^2 + (a_1T_1 + a_2T_2)s + 1\right)}{b_4s^4 + b_3s^3 + b_2s^2 + b_1s + b_0}$$

with $b_4 = T_1T_2$, $b_3 = (T_1 + T_2 + 2T_1T_2)$, $b_2 = (1 + 2(T_1 + T_2) + kK_pa_1T_1a_2T_2)$, $b_1 = (2 + kK_P(a_1T_1 + a_2T_2))$ and $b_0 = kK_P$.
The behavior of the closed-loop dynamics is illustrated in Fig. 5.24
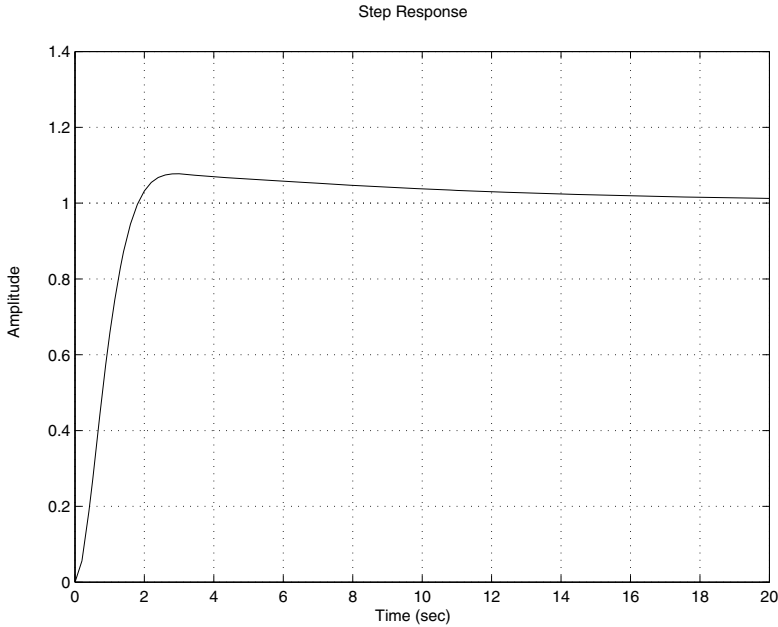
Fig. 5.24 Step response of $F(s)$

It seems that the settling time is a little bit far from the desired one. To overcome this, we play with the positions of the poles and the zeros of the controller and repeat the procedure.

5.5   Design Based on Bode Plot

The design methods we will develop in this section have the advantage over those presented in the previous section by the fact they don't need the knowledge of the mathematical model of the system to be controlled as it required by the techniques based on root locus method. The objective of this section is to cover the methods that we can use for designing the controllers treated in the previous section by using the frequency domain.

The design procedures for the different controllers we will cover here are mainly based on the fact to assure that the closed-loop dynamics of the system will have a phase margin, $\Delta\phi$ satisfying:

$$45^o \le \Delta\phi \le 50^o$$

while the gain margin, ΔG satisfies $\Delta G \ge 8\ db$.

In the rest of this section we assume that the system is described by the following transfer function:

$$G(s) = k\frac{a_m s^m + \cdots + 1}{s^l (a_n s^n + \cdots + 1)}$$

where l is the type of the system, $l + n$ is the degree of the system and $m < n + l$ is the degree of the numerator of the system that we suppose to be causal.

Our goal in this section consists of designing a controller that respond to some given performances. The controllers we consider in this section are those treated in the previous sections. It is important to notice that the idea used in the methods we will cover is based on the deformation of the magnitude and phase curves locally to satisfy the desired performances.

Remark 5.5.1 *It is important to notice that this method doesn't apply for unstable system.*

Let firstly consider the design of the proportional controller ($C(s) = K_P$). This controller has limited actions and can only move vertically the magnitude curve without affecting the phase curve. The open loop transfer function of the compensated system is given by:

$$\begin{aligned}
G_c(s) &= kK_P\frac{a_m s^m + \cdots + 1}{s^l (a_n s^n + \cdots + 1)} \\
&= K\frac{a_m s^m + \cdots + 1}{s^l (a_n s^n + \cdots + 1)}
\end{aligned}$$

The following procedure can be used for the design of the proportional controller that responds to the desired performances:

1. obtain the Bode plot for the compensated system, $G_c(s)$, with $K = 1$

2. determine the frequency, w_c, for which the phase margin is equal to 45°

3. determine the magnitude at this frequency and compute the gain, $\bar{K}_P$ that will move the magnitude curve vertically to get the desired phase margin. A gain greater than one will move the magnitude curve up while a gain less than one will move it down. The controller gain is given by:

$$K_P = \frac{\bar{K}_P}{k}$$

4. draw the Bode diagram of the compensated system, with the computed gain and check that the gain margin is greater than $8\ db$

Example 5.5.1 *To show how the design procedure for the proportional controller works, let us consider the following dynamical system:*

$$G(s) = \frac{2}{(0.1s + 1)(0.2s + 1)(0.5s + 1)}$$

The performances we would like to have for this system are:

1. *stable system in closed-loop*

2. *phase margin about* 45°

3. *gain margin greater than* 10 *db*

To design our proportional controller, let us follow the previous procedure.

1. *the open loop transfer function of the compensated system,* $T(s)$ *is given by:*

$$T(s) = 2K_P \frac{1}{(0.01s + 1)(0.2s + 1)(0.5s + 1)}$$

2. *The Bode plot of this transfer function with* $K = 1$ *is illustrated in Fig. 5.25.*
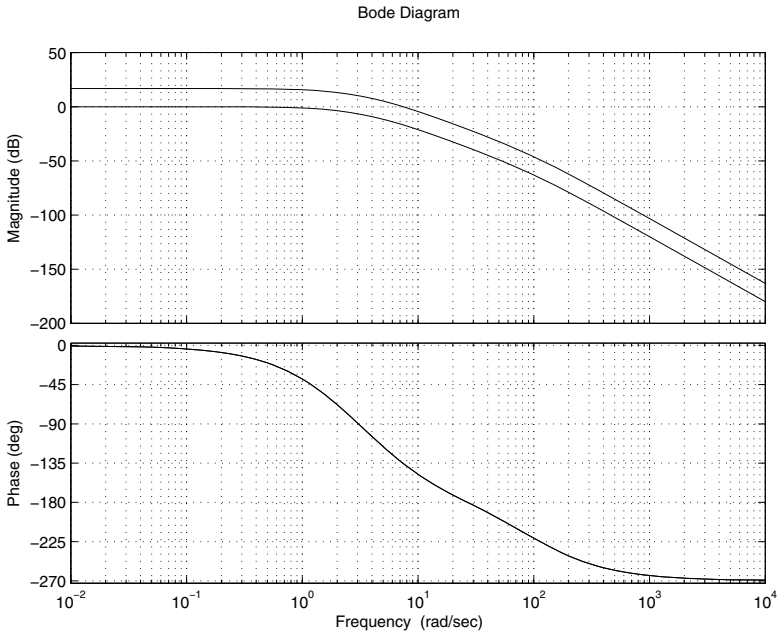
Fig. 5.25 Bode plot of $T(s)$, with $K = 1$, and $K = kK_P$

From this figure we conclude that at the frequency $w_1 = 7.44 \ rd/s$ *we have* $\Delta\phi = 45^o$. *The corresponding magnitude is equal to* $|T(jw_1)| = -16.8$.

3. *The corresponding gain that allows us to move the magnitude curve by* 16.8 *db*
   to get the desired phase margin is given by:

$$\bar{K}_P = 16.8 \ db$$

which implies $\bar{K}_P = 10^{\frac{16.8}{20}} = 6.9183$
Finally, we get the controller gain as follows:

$$K_P = \frac{\bar{K}_P}{k} = \frac{6.9183}{2} = 3.4592$$

The open loop transfer function of the compensated system is given by:

$$T(s) = 2 \times 3.4592 \frac{1}{(0.01s + 1)(0.2s + 1)(0.5s + 1)}$$

*The Bode plot of this transfer function is reported in Fig. 5.25. If we compute
the phase and the gain margins we get:*

$$\Delta G = 10.8363$$
$$\Delta \phi = 44.9849$$

The corresponding frequencies are:

$$w_g = 26.65 \ rd/s, for \ the \ gain \ margin$$
$$w_p = 7.44 \ rd/s, for \ the \ phase \ margin$$

4. *The transfer function of the closed loop with this controller is given by:*

$$F(s) = \frac{kK_P}{0.001s^3 + 0.107s^2 + 0.71s + 1 + kK_P}$$

with $k = 2$.
The behavior of the closed-loop dynamics is illustrated in Fig. 5.26
This response has a steady state error equal to 0.13. *The proportional controller
is unable to make equal to zero but it can be reduced by increasing the gain. This
may degrade the transient regime.*

Remark 5.5.2 *It is important to notice that the system considered in the previous
example is of type zero and therefore, the error for a step input with a proportional
controller is constant and it is given by:*

$$e(\infty) = \frac{1}{1 + kK_p}.$$

*From this expression, it is impossible to make the error equal to zero by increas-
ing the gain of the controller. Incrementing the type of the system is a solution that
can be given by the PI controller.*

Let us now focus on the design of the PI controller using the Bode method. As
we have seen previously increase the type of the system by one and therefore, it
may bring the steady state error to zero. Its disadvantage is that settling time may
increase.
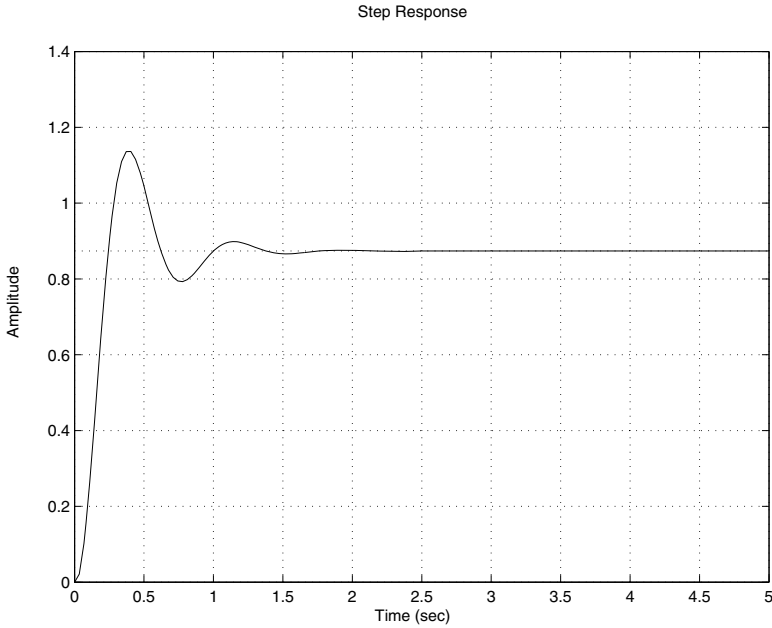
Step Response

Fig. 5.26 Step response of $F(s)$

To design the PI controller, let us assume that its transfer function is described by:

$$C(s) = K_P + \frac{K_I}{s}$$
$$= \frac{1 + \tau_n s}{\tau_i s}$$

with $K_P = \frac{\tau_n}{\tau_i}$ and $K_I = \frac{1}{\tau_i}$.

Using this, the open loop transfer function of the compensated system is given by:

$$T(s) = C(s)G(s) = K(1 + \tau_n s) \frac{b_m s^m + \cdots + b_1 s + 1}{s^{l+1}(a_n s^n + \cdots + a_1 s + 1)}$$

with $K = \frac{k}{\tau_i}$

The following procedure can be used for the design of this controller:

1. determine the slowest pole that is not equal to those at the origin (pole that corresponds the highest time constant) and proceed with a zero/pole cancelation. This will allow us to determine the parameter τ_n by:

$$\tau_n = \max\{\tau_1, \cdots, \tau_\nu\}$$

where τ_j, $j = 1, \cdots, \nu$ are the time constant of the system to be controlled.

2. determine the gain $\bar{K}_P$ that gives the desired phase margin using the Bode plot and obtain:

$$\tau_i = \frac{k}{\bar{K}_P}$$

3. determine the gains K_P and K_I of the controller using:

$$K_P = \frac{1}{\tau_i}$$
$$K_I = \frac{\tau_n}{\tau_i}$$

4. determine the open loop transfer function of the compensated system and check if the desired performances are obtained or not. In case of negative response adjust τ_n and repeat the procedure design.

Example 5.5.2 *To show how this procedure works let us consider the following dynamical system:*

$$G(s) = \frac{1}{(s+1)(s+5)(s+10)}$$

and design a PI controller that gives a steady error equal to zero and a phase margin about 45° and a gain margin greater than 8 db.

To answer these performances, let us follow the previous procedure:

1. *the open transfer function of the system to be controller has 1, 0.2 and 0.1 as time constants. The maximum one is equal to 1 and therefore by canceling the corresponding pole by the controller's zero, we get:*

$$\tau_n = 1 \; s$$

2. *the open loop transfer function with the pole/zero cancellation is given by:*

$$T(s) = \frac{0.02K}{s(0.2s+1)(0.1s+1)}$$

The Bode plot of this transfer function is shown at Fig. 5.27
At $w = 2.8 \; rd/s$, the phase margin is equal to 45° and at this frequency the magnitude is equal to -10.5 db. To get such phase margin we need to translate up the magnitude curve by 17.5 db which implies the use of a gain:

$$\bar{K}_P = 10^{\frac{10.5}{20}} = 3.3497$$

which implies in turn:

$$\tau_i = \frac{0.02}{\bar{K}_P} = \frac{0.02}{3.3497} = 0.0060$$

Bode Diagram
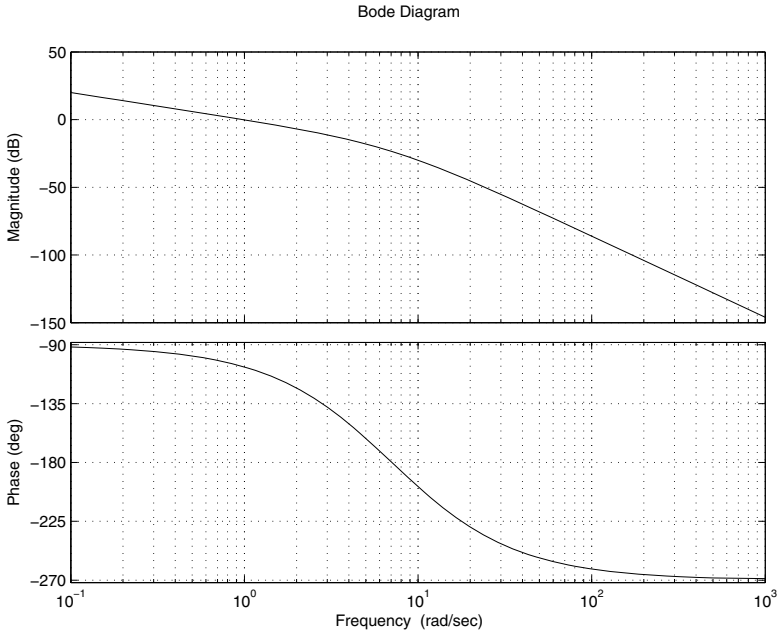
Fig. 5.27 Bode plot of $T(s)$, with $K = 1$

3. the controller gains are given by:

$$K_P = \frac{1}{\tau_i} = \frac{1}{0.0027} = 166.6667$$

$$K_I = \frac{\tau_n}{\tau_i} = \frac{1}{0.0027} = 166.6667$$

4. with this controller we can check that the phase margin is equal to 45.1^o but the gain margin is equal to 4.5 db. The closed loop transfer function with this controller is given by:

$$F(s) = \frac{K_P}{s^3 + 15s^2 50s + K_P}$$

If we accept the gain margin as it is now, the design is complete otherwise we have to modify the value for τ_n and repeat the design
The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.28
The settling time at 5 % is equal to 1.47 s which is acceptable and the error for a step input is equal to zero.

Let us now focus on the design of the PD controller using the Bode method. This controller improves the transient regime. The transfer function of this controller is given by:
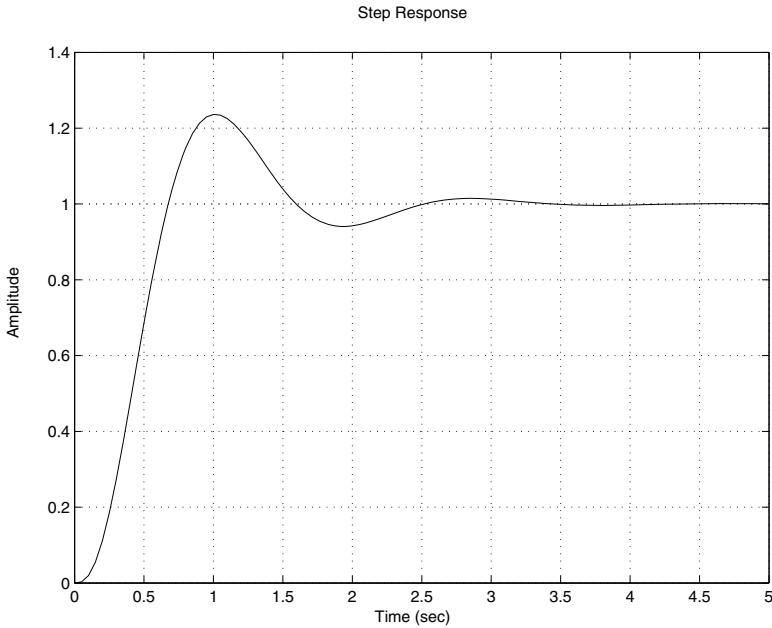
Fig. 5.28 Step response of $F(s)$

$$C(s) = K_P + K_D s = K_P (1 + \tau_D s)$$

with $\tau_D = \frac{K_D}{K_P}$.

The open loop transfer function of the compensated system is given by:

$$T(s) = K \frac{(1 + \tau_D s)(b_m s^m + \cdots, b_1 s + 1)}{s^l (a_n s^n + \cdots, a_1 s + 1)}$$

where $K = k K_P$

The design of the PD controller is brought to the determination of the two gains K_P and K_D. The following procedure can be used for the design of this controller:

1. from the error specifications, determine the gain, $\bar{K}_P$ that gives the desired error

2. draw the Bode diagram of the system:

$$\bar{K}_P \frac{b_m s^m + \cdots, b_1 s + 1}{s^l (a_n s^n + \cdots, a_1 s + 1)}$$

    and determine the frequency, w_m at the which the magnitude is equal to $-20\ db$

3. since the cut frequency of the PD controller is equal to $\frac{1}{\tau_D}$, at the frequency $\frac{10}{\tau_D}$, the contribution of the PD controller to the magnitude and the phase are respectively 20 db and 90°. If we select τ_D such that:

$$\tau_D = \frac{10}{w_m}$$

the phase margin of the compensated system is given:

$$\Delta\phi_c = \Delta\phi + 90$$

where $\Delta\phi$ is the phase margin of the system without the controller at the frequency w_m

If

$$\Delta\phi_c \begin{cases} < 40^o & \text{choose another controller} \\ > 50^o & \text{reduce the parameter, } \tau_D \text{ till } \Delta\phi_c = 45^o \end{cases}$$

4. compute the controller's gains using:

$$K_P = \frac{\bar{K}_P}{k}$$
$$K_D = \bar{K}_P\tau_D$$

5. check if the desired specifications are obtained or not

Example 5.5.3 *To show how the procedure of the design of the PD controller works, let us consider the following dynamical system:*

$$G(s) = \frac{4}{s(0.1s + 1)(4s + 1)}$$

As specifications we consider the following:

1. *stable system*

2. *phase margin equal to 45°*

3. *steady state error equal to 0.1*

To satisfy these specifications a PD controller has to be designed. For this purpose let is follow the previous procedure:

1. *from the error specification, we need to fix $\bar{K}_P$ to 10.*

2. *the Bode diagram of:*

$$\frac{\bar{K}_P}{s(0.1s + 1)(4s + 1)}$$

is illustrated by Fig. 5.29 which shows that magnitude is equal to -20 db when the frequency $w_m = 4.73$ rd/s. The parameter τ_D is then given by:

$$\tau_D = \frac{10}{w_m} = \frac{10}{4.73} = 2.1142$$

Bode Diagram
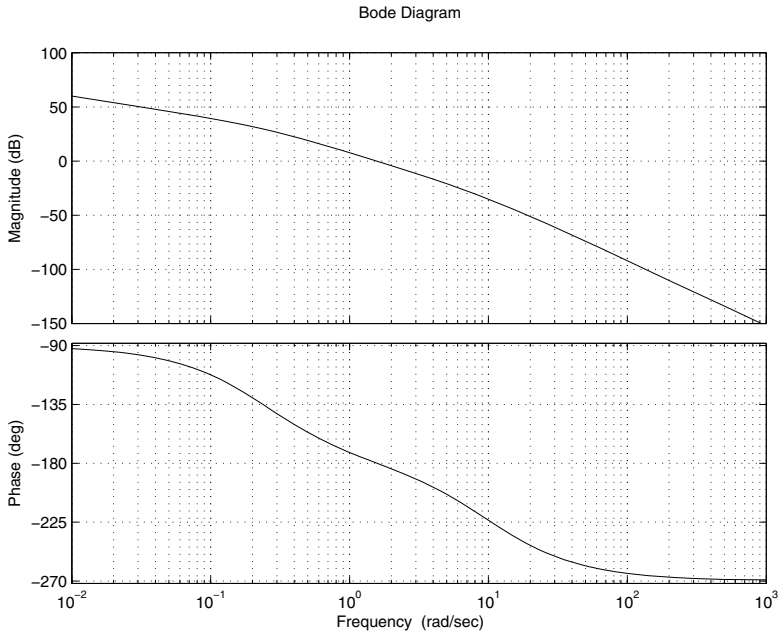
Fig. 5.29 Bode plot of $T(s)$, with $K = 10$

3. *the phase of the system with the controller at $w_m = 4.73 \ rd/s$ is equal to -202°. The phase margin of the compensated system is given by:*

$$\Delta\phi_c = 180 - 202 + 90 = 68^o$$

The phase margin is greater than 45° and we should decrease the parameter τ_D. Therefore if we select $\tau_D = \frac{10}{9.1} = 1.0989$, the phase margin in this case is equal to 49°

4. *the controller gains are give by:*

$$K_P = \frac{\bar{K}_P}{k} = \frac{10}{4} = 2.5$$
$$K_D = \bar{K}_P \tau_D = 2.4 \times 1.0989 = 2.7473$$

5. *the open loop transfer function of the compensated system is given by:*

$$T(s) = \frac{4(K_P + K_D s)}{s(0.1s + 1)(4s + 1)}$$

This controller gives a phase margin about 61.5^o. The closed-loop transfer function is given by:

$$F(s) = \frac{4(K_s + K_p)}{0.1s^3 + 4.1s^2 + (1 + 4K_D)s + 4K_P}$$

The step response of the compensated system is represented in Fig. 5.30.
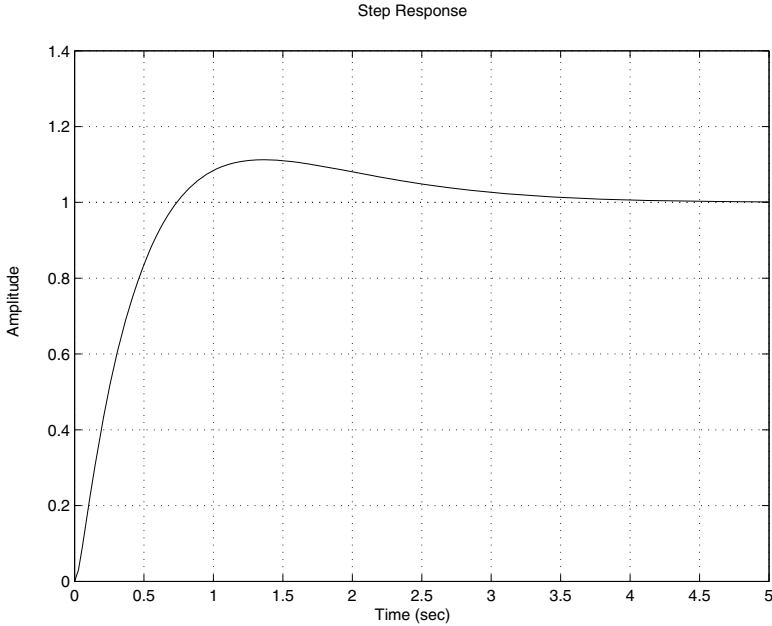
Step Response

Fig. 5.30 Step response of $F(s)$

Let us now focus on the design of the PID controller using the Bode method. This controller acts on the transient and steady state regimes. The transfer function of this controller is given by:

$$C(s) = K_P \frac{K_I}{s} + K_D s = \frac{(1 + \tau_n s)(1 + \tau_v s)}{\tau_i s}$$

where $K_P = \frac{\tau_n + \tau_v}{\tau_i}$, $K_I = \frac{1}{\tau_i}$ and $K_D = \frac{\tau_n \tau_v}{\tau_i}$.

The open loop transfer function of the compensated system is given by:

$$T(s) = K \frac{(1 + \tau_n s)(1 + \tau_i s)(b_m s^m + \cdots + b_1 s + 1)}{s^{l+1}(a_n s^n + \cdots + a_1 s + 1)}$$

with $K = \frac{k}{\tau_i}$.

To design such controller we use the ideas used to design separately the PI and the PD controllers. The procedure to design such controller is based on the fact that a pole is introduced at the origin, the gain, $\bar{K}_P$ that gives the steady error and the use of the maximum phase, 90° (introduced by the PD controller) that corresponds to the frequency when the magnitude is to $-20 \, db$ ($w_m \tau_v = 10$). The following procedure can be used for the design of this controller:

1. determine the slowest pole of the system to controller except those at the origin and proceed with a pole/zero cancellation. This will help to fix, τ_n, i.e.:

$$\tau_n = \max\{\tau_1, \cdots, \tau_v\}$$

2. determine the gain $\bar{K}_P$ that gives the desired error

3. plot the Bode diagram of:

$$T(s) = \bar{K}_P \frac{(1 + \tau_n s)(b_m s^m + \cdots + b_1 s + 1)}{s^{l+1}(a_n s^n + \cdots + a_1 s + 1)}$$

and determine the frequency w_m at which the magnitude is equal to $-20\ db$. Using this frequency we determine τ_v by:

$$\tau_v = \frac{10}{w_m}$$

the phase margin of the compensated system is given:

$$\Delta\phi_c = \Delta\phi + 90$$

where $\Delta\phi$ is the phase margin of the system without the controller at the frequency w_m

If

$$\Delta\phi_c \begin{cases} < 40^o & \text{choose another controller} \\ > 50^o & \text{reduce the parameter, } \tau_D \text{ till } \Delta\phi_c = 45^o \end{cases}$$

4. compute the controller's gains using:

$$K_P = \frac{\tau_n + \tau_v}{\tau_i}$$

$$K_I = \frac{1}{\tau_i}$$

$$K_D = \frac{\tau_n \tau_v}{\tau_i}$$

5. check if the desired specifications are obtained or not

Example 5.5.4 *To show how the design of the PID controller works, let us consider the following dynamical system:*

$$G(s) = \frac{2}{(0.1s + 1)(0.2s + 1)(0.5s + 1)}$$

A steady state error to a unit ramp equal 0.1 is needed.

This system is of type zero and has three time constant, 0.5, 0.2 and 0.1. The maximum time constant is 0.5.

Following the procedure design, we get:

1. *using the maximum time constant of the system we have:*

$$\tau_n = 0.5$$

2. *using the error specification, we get:*

$$\bar{K}_P = \frac{1}{0.1} = 10$$

3. *draw the Bode diagram of:*

$$T(s) = \frac{\bar{K}_P}{s(0.1s + 1)(0.2s + 1)}$$
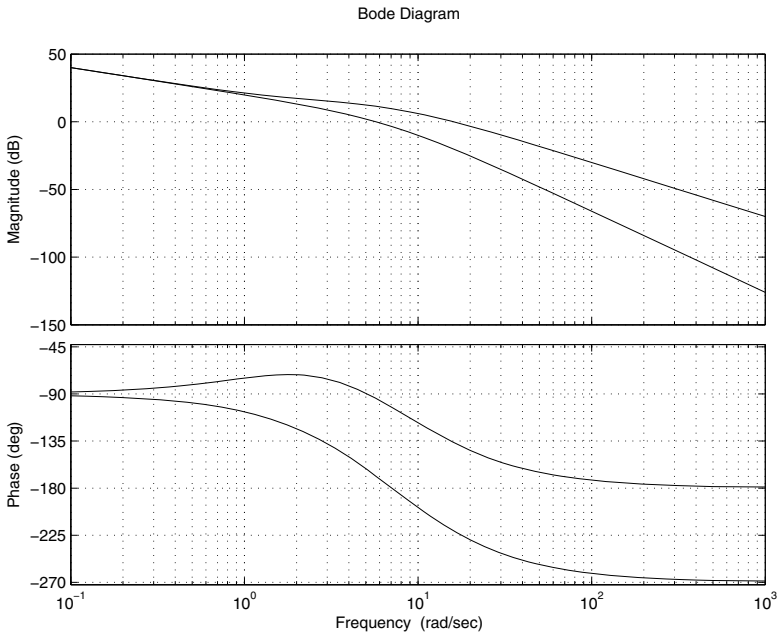
Fig. 5.31 Bode plot of $T(s)$

This diagram is illustrated by Fig. 5.31. The frequency at which the magnitude is equal to -20 db is equal to $w_m = 15.9$. The phase at this frequency is equal to -220°. The phase margin at this frequency is given by:

$$\Delta\phi = 180 + \phi(w_m) + 90 = 180 - 220 + 90 = 50$$

The second parameter, τ_v of the controller is determined by:

$$\tau_v = \frac{10}{w_m} = 0.6289$$

4. *compute the controller's gains using:*

$$\tau_i = \frac{2}{10} = 0.2$$

$$K_P = \frac{\tau_n + \tau_v}{\tau_i} = 5.6447$$

$$K_I = \frac{1}{\tau_i} = 5$$

$$K_D = \frac{\tau_n \tau_v}{\tau_i} = 1.5723$$

5. *The closed-loop transfer function with this controller is given:*

$$F(s) = \frac{\frac{2}{\tau_i}(\tau_v s + 1)}{0.02s^2 + 0.2s^2 + (1 + \frac{2\tau_v}{\tau_i})s + \frac{2}{\tau_i}}$$

The step response of the compensated system is represented in Fig. 5.32.
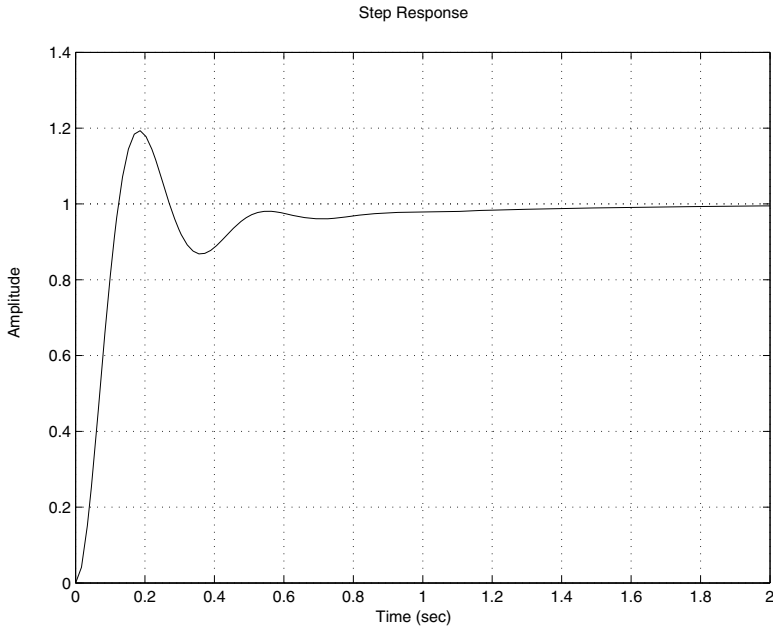
Fig. 5.32 Step response of $F(s)$

Let us now focus on the design of the phase-lead controller using the Bode method. The transfer function of this controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}, a > 1$$

It can be shown that this controller can be deliver a maximum of phase for each value for a. The value of this maximum and the frequency at which this happens are given by:

$$w_m = \frac{1}{T\sqrt{a}}$$

$$\sin(\phi_m) = \frac{a-1}{a+1}$$

The second relation gives also:

$$a = \frac{1+\sin(\phi_m)}{1-\sin(\phi_m)}$$

These relations are of great importance in the design procedure of the phase-lead controller.

The following procedure can be used for the design of this controller:

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. plot the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

and determine the phase and gain margins of the non-compensated system. Then compute the phase margin missing. This value increased by a factor (5°) for safety is considered as ϕ_m, then compute the parameter a by:

$$a = \frac{1+\sin(\phi_m)}{1-\sin(\phi_m)}$$

3. determine the frequency, w_m for which the magnitude of the non-compensated system is equal to $-20\log\sqrt{a}$ and consider it as the crossover of the compensated system. The parameter T of the controller is determined using:

$$T = \frac{1}{w_m\sqrt{a}}$$

4. check if the desired specifications are obtained or not

Example 5.5.5 *Let us consider the following dynamical system:*

$$G(s) = \frac{5(0.125s+1)}{s(2s+1)(0.1s+1)}$$

Our objective in this example is to design a phase-lead controller satisfies the following specifications:

1. *stable system*

2. *steady state error for a ramp input equal to 0.1*

3. *phase margin greater than* 40°

4. *gain margin greater than* 6 *db*

   The design of the phase-lead controller is brought to the determination of the parameters a and T. To accomplish this, we follow the previous procedure.

1. *since the system is of type one, therefore the error for a ramp input is given by:*

$$e(\infty) \ = \ \frac{1}{\bar{K}_P}$$

   which gives in turn:

$$\bar{K}_P \ = \ 10$$

   which gives:

$$K_P \ = \ \frac{\bar{K}_P}{k} = 2$$

2. *with this gain, the open loop transfer function of the system becomes:*

$$T(s) \ = \ \frac{10}{s(2s + 1)(0.1s + 1)}$$

   The Bode diagram of this system is given by Fig. 5.33.
   From this diagram we conclude that the system with a proportional controller has a phase margin equal to 15.67^o and a gain margin equal to ∞ db. To get our desired phase margin we need to add 24.33^o. If we take a 5° safety, the controller should add a phase, ϕ_m equal to 29.33^o. This gives:

$$a \ = \ \frac{1 + \sin(29.33)}{1 - \sin(29.33)} = 2.9201$$

3. *with this value of a we have:*

$$-20log \ \sqrt{a} = -4.6540$$

   From 5.33, we remark that the magnitude curve takes -4.6540 at the frequency $w_m = 2.93$ rd/s. This gives:

$$T \ = \ \frac{1}{w_m \ \sqrt{a}} = 0.1997$$

   The controller is then given by:

$$C(s) \ = \ K_P \frac{aT s + 1}{T s + 1} = 2 \frac{0.5832s + 1}{0.1997s + 1}$$

   The open loop transfer function of the compensated system is given by:

$$T(s) \ = \ 10 \frac{0.5832s + 1}{s(2s + 1)(0.1s + 1)(0.1997s + 1)}$$

Bode Diagram
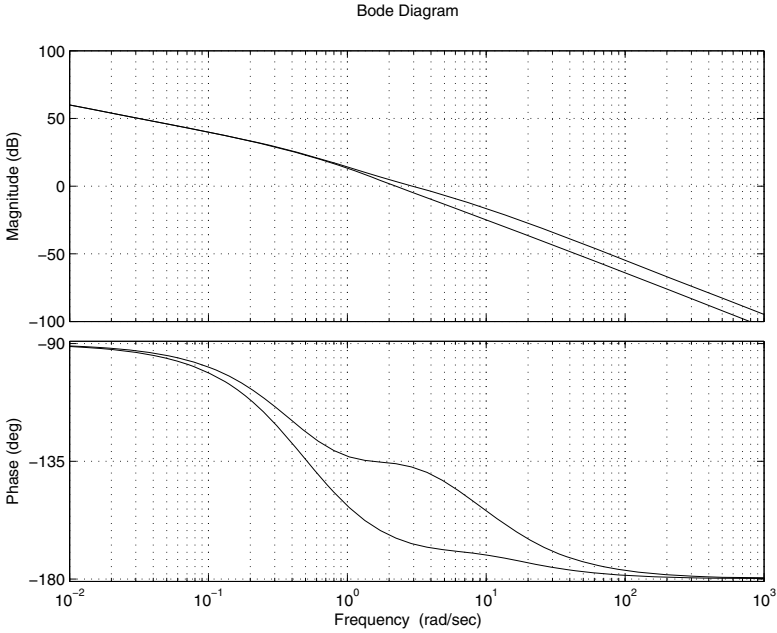
Fig. 5.33 Bode plot of $T(s)$

4. *with controller we get* 42.8^o *and* ∞ *db as phase margin and gain margin respectively.*

   The closed-loop transfer function is given by:

   $$F(s) = \frac{kK_P\left(0.125aTs^2 + (0.125 + aT)s + 1\right)}{b_4s^4 + b_3s^3 + b_2s^2 + b_1s + b_0}$$

   with $k = 5$, $b_4 = 0.2T$, $b_3 = 0.2 + 2.1T$, $b_2 = 2.1 + T + 0.125aTkK_P$, $b_1 = 1 + akK_P(0.125 + aT$ *and* $b_0 = kK_P$.

   The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.34

   The settling time at 5 % is equal to 1.68 *s which is acceptable and the error for a step input is equal to zero while the overshot is about 30 %.*

Let us now focus on the design of the phase-lag controller using the Bode method. The transfer function of this controller is given by:

$$C(s) = K_P\frac{aTs + 1}{Ts + 1}, a < 1$$
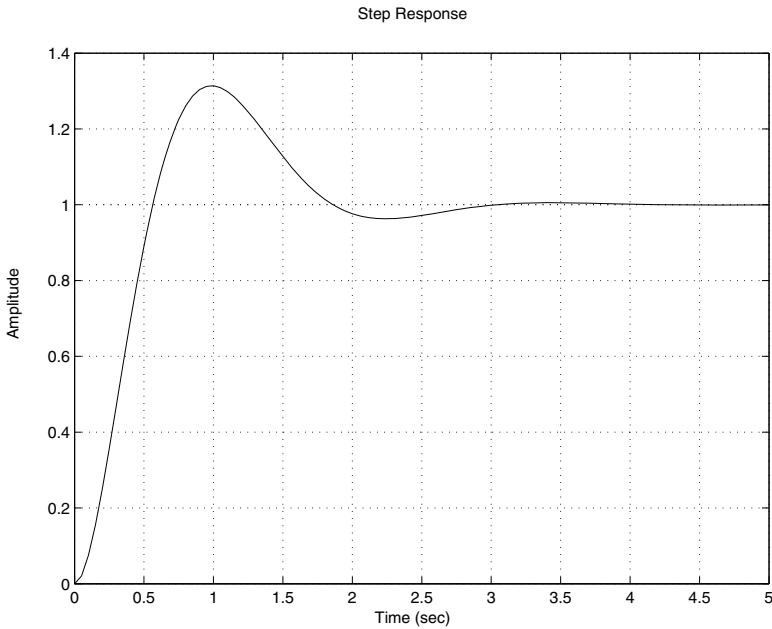
Fig. 5.34 Step response of $F(s)$

The following procedure can be used for the design of this controller:

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. plot the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

and determine the frequency, w_m of the non-compensated system at which we have the desired phase margin. Then compute of how much decibels, m to bring the magnitude to 0 db at w_m. The parameter a of the controller is given by:

$$a = 10^{\frac{m}{20}}$$

3. To get an appreciable change the phase curve, we need to choose, the parameter, T as follows:

$$T = \frac{10}{a w_m}$$

4. check if the desired specifications are obtained or not

Example 5.5.6 *Let us consider the following dynamical system:*

$$G(s) = \frac{2}{s(0.1s + 1)(0.05s + 1)}$$

Our objective in this example is to design a phase-lag controller satisfies the following specifications:

1. *stable system*

2. *steady state error for a ramp input equal to 0.1*

3. *phase margin greater than 40°*

4. *gain margin greater than 4 db*

The design of the phase-lag controller is brought to the determination of the parameters a and T. To accomplish this, we follow the previous procedure.

1. *the system to be controlled is of type one. The steady error to a unit ramp as input is given by:*

$$e(\infty) = \frac{1}{\bar{K}_P}$$

which implies:

$$\bar{K}_P = 10$$

From this we conclude that the gain of the controller is $K_P = 5$.

2. *with this gain, the open loop transfer function of the system becomes:*

$$T(s) = \frac{10}{s(0.1s + 1)(0.05s + 1)}$$

The Bode diagram of this system is given by Fig. 5.35.
From this figure, we conclude, that at $w_m = 5.59$ rd/s, the phase margin is equal to 45°. At this frequency the magnitude is equal to 3.52 db. Using this, the parameter, a is given bY:

$$a = 10^{\frac{-3.52}{20}} = 0.6668$$

Remark 5.5.3 *The fact that we consider -3.52 db means that we want the controller to introduce this amplitude at this frequency.*

3. *the choice of T is done by placing the frequency $\frac{1}{aT}$ at a decade from $w_m = 5.59$ rd/s, i.e.:*

$$w_m = \frac{10}{aT}$$

which implies $T = 2.6828$.
The transfer function of our phase-lag controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}$$
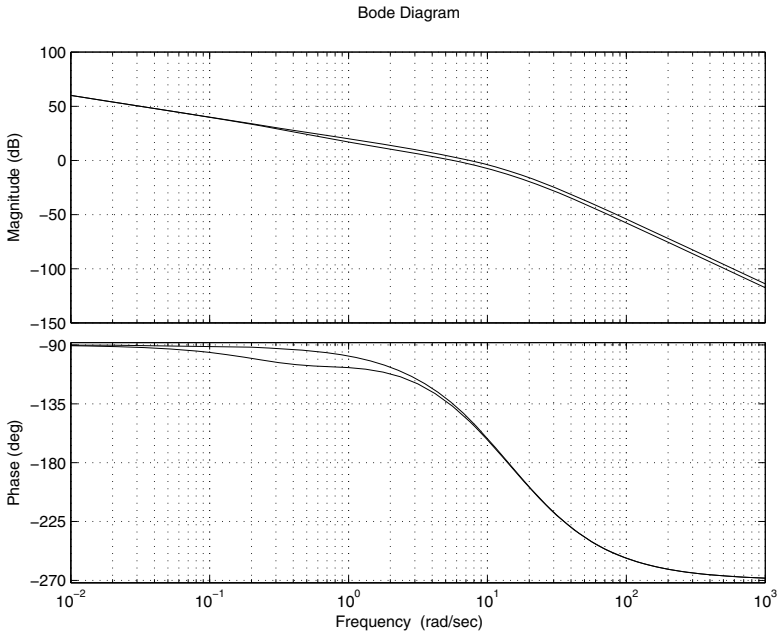
Fig. 5.35 Bode plot of $T(s)$

with $K_P = 5$.
With this controller we get:

$$\Delta\phi \ = 43.13^o$$
$$\Delta G \ = 4.37 \ db$$

The closed-loop transfer function is given by:

$$F(s) \ = \frac{kK_P\,(aT\,s + 1)}{0.005T\,s^4 + (0.005 + 0.15T)s^3 + (0.15 + T)s^2 + (1 + kK_paT)s + kK_P}$$

with $k = 2$.
The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.36
The settling time at 5 % is equal to 0.78 s which is acceptable and the error for a step input is equal to zero while the overshot is about 27 %.

Let us now focus on the design of the phase lead-lag controller using the Bode method. The transfer function of the controller is given by:

$$C(s) \ = K_P\frac{a_1 T_1 s + 1}{T_1 s + 1}\frac{a_2 T_2 s + 1}{T_2 s + 1}, a_1 > 1, a_2 < 1$$

The following procedure can be used for the design of this controller:
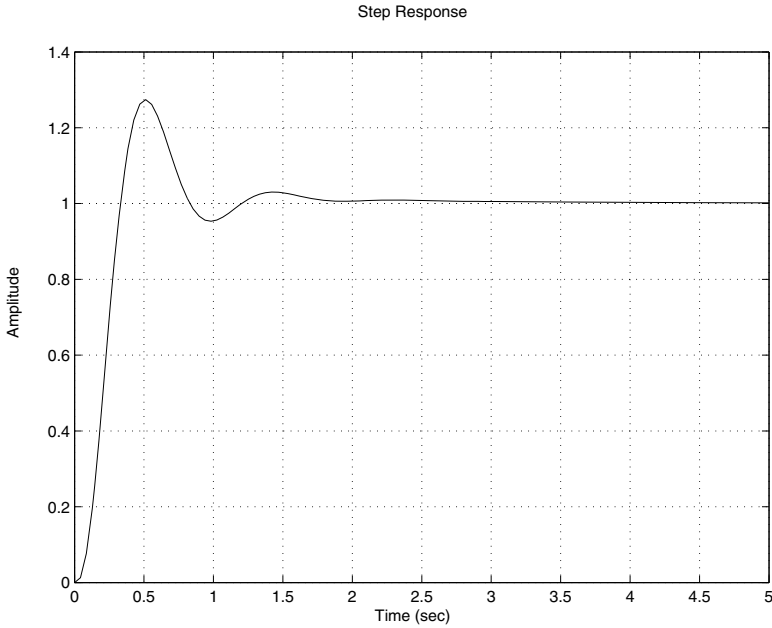
Fig. 5.36 Step response of $F(s)$

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. draw the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

and determine the phase margin of the non-compensated system

3. determine the phase-lead controller's parameters, a_1 and T_1

4. determine the phase-lag controller's parameters, a_2 and T_2

5. check if the desired specifications are obtained or not

Example 5.5.7 *To show how to design a phase lead-lag controller let us consider the following dynamical system:*

$$G(s) = \frac{4(0.125s + 1)}{s(0.1s + 1)(0.2s + 1)}$$

As specifications we search to get the following ones:

1. *stable system*

2. *steady state error to a unit ramp equal to 0.05*

3. *a phase margin greater than 40°*

4. *a gain margin greater than 8 db*

   To design the phase lead-lag controller let us follow the steps of the previous procedure.

1. *to get the desired error a gain $\bar{K}_P$ equal to 20, which corresponds to $K_P = 5$.*

2. *The transfer function of the open loop of the non compensated system with this gain is given by:*

$$T(s) = \frac{20(0.125s + 1)}{s(0.1s + 1)(0.2s + 1)}$$

   The Bode diagram of this system is given by Fig. 5.37.
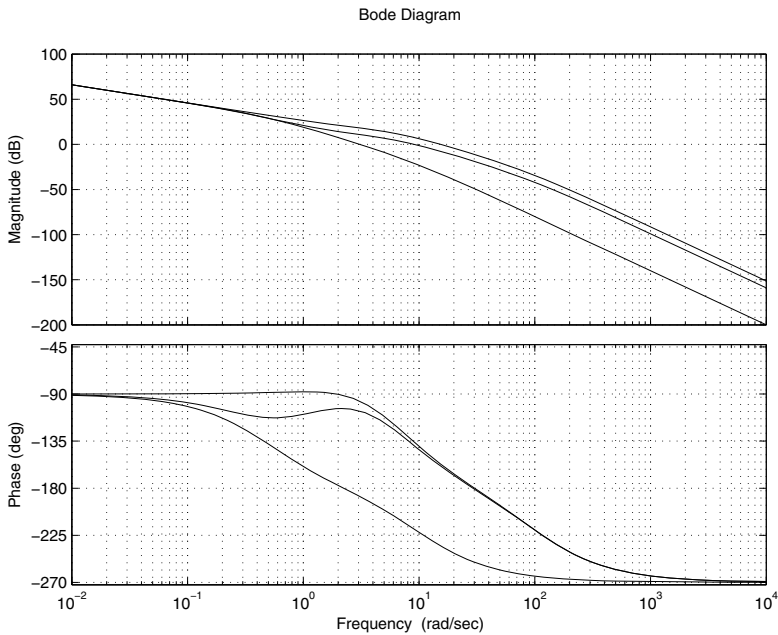
Fig. 5.37 Bode plot of $T(s)$

   With this proportional controller the system has:

$$\Delta\phi = 32.7^o$$
$$\Delta G = \infty \, db$$

3. *to design the phase-lead controller can be done following the previous procedure for this purpose. Notice that to get the desired phase margin, the*

phase-lead controller must bring a phase of $45^o - 32.7^o = 12.3^o$. *Using this, we have:*

$$a_1 = \frac{1 + \sin(12.3)}{1 - \sin(12.3)} = 1.5414$$

Using the value of a_1, *we get:*

$$-20 \log \sqrt{a_1} = -1.8791$$

Now if we refer to the Fig. 5.37, the magnitude will have -1.8791 *at the frequency* $w_m = 11.4 \ rd/s$. *This implies:*

$$T_1 = \frac{1}{w_m \sqrt{a_1}} = 0.0707$$

The transfer function of the phase lead controller is given:

$$C(s) = \frac{0.4231 s + 1}{0.0707 s + 1}$$

The open loop transfer function of the system with controller is given by:

$$T(s) = 20 \frac{a_1 T_1 s + 1}{s(0.2s + 1)(0.01s + 1)(T_1 s + 1)}$$

4. *the system compensated with the phase lead controller has:*

$$\Delta \phi = 10.9624^o$$
$$\Delta G = \infty \ db$$

To get a phase margin equal to 45° *and if we report to the Fig. 5.37, we have this at the frequency* $w_m = 10 \ rd/s$. *Also at this frequency, the magnitude is equal to* $1.76 \ db$. *using this we get the parameter* a_2 *for the phase lag controller:*

$$a_2 = 10^{\frac{-1.76}{20}} = 0.8166$$

The choice of T_2 *is given by:*

$$T_2 = \frac{10}{w_m a_2} = \frac{10}{9.07 \times 0.4154} = 2.6542$$

The transfer function of the phase lead controller is given:

$$C(s) = \frac{1.1026 s + 1}{2.6542 s + 1}$$

5. *The open loop transfer function of the compensated system is given by:*

$$T(s) = 20 \frac{(a_1 T_1 s + 1)(a_2 T_2 s + 1)(0.125 s + 1)}{s(0.2s + 1)(0.1s + 1)(T_1 s + 1)(T_2 s + 1)}$$

The Bode diagram of this transfer function is reported in Fig. 5.37 and from which we get:

$$\Delta \phi = 44.1^o$$
$$\Delta G = \infty \ db$$

The closed-loop transfer function of the compensated system

$$F(s) = kK_P \frac{\alpha_3 s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0}{b_5 s^5 + b_4 s^4 + b_3 s^3 + b_2 s^2 + b_1 s + b_0}$$

with $\alpha_3 = 0.125 a_1 a_2 T_1 T_2$, $\alpha_2 = 0.125(a_1 T_1 + a_2 T_2) + a_1 a_2 T_1 T_2$, $\alpha_1 = 0.125 + a_1 T_1 + a_2 T_2$ *and* $\alpha_0 = 1$; $b_5 = 0.02 T_1 T_2$, $b_4 = 0.3 T_1 T_2 + 0.02(T_1 + T_2)$, $b_3 = 0.02 + T_1 T_2 + 0.3(T_1 + T_2) + 0.125 k K_P a_1 a_2 T_1 T_2$, $b_2 = 0.3 + T_1 + T_2 + k K_P(0.125(a_1 T_1 + a_2 T_2) + a_1 a_2 T_1 T_2)$, $b_1 = 1 + k K_P(0.125 + a_1 T_1 + a_2 T_2)$ *and* $b_0 = k K_P$

5.6   Case Study

The goal of this section is to make the design of different controllers for our dc motor kit using the developed methods and show the reader how things apply in practice. It was shown that the model of this system is given by:

$$G(s) = \frac{K_m}{s(\tau_m s + 1)}$$

where $K_m = 48.5$ is the gain and $\tau_m = 0.060\ s$ is the time constant.

Our objective is to design the proportional controller, the proportional and integral controller, the proportional and derivative controller, the proportional, integral and derivative controller, the phase lead controller, the phase lag controller and the phase lead-lag controller using the three methods and implement them in real-time on our dc motor kit.

Regarding the specifications, we will not fix them but during the design of each controller we will try to get the best specifications that may offer each controller.

5.6.1   Proportional Controller

Let us first of all consider the design of the proportional controller. This controller is assumed to have the following transfer function:

$$C(s) = K_P$$

where K_P is the gain to be determined.

For the empirical methods, it is clear that the time domain methods will not apply since the transfer function of the system has a pole at the origin and will never provide a step response with periodic oscillations.

To compute the gain of the controller, we notice that we have to move up the magnitude by 27.27 db, from Fig. 5.38, which gives a gain equal to:

$$\bar{K}_P = 10^{\frac{27.27}{20}} = 22.9087$$

The gain of the controller is given by:

$$K_P = \frac{\bar{K}_P}{K_m} = \frac{22.9087}{48.5} = 0.4723$$

Bode Diagram
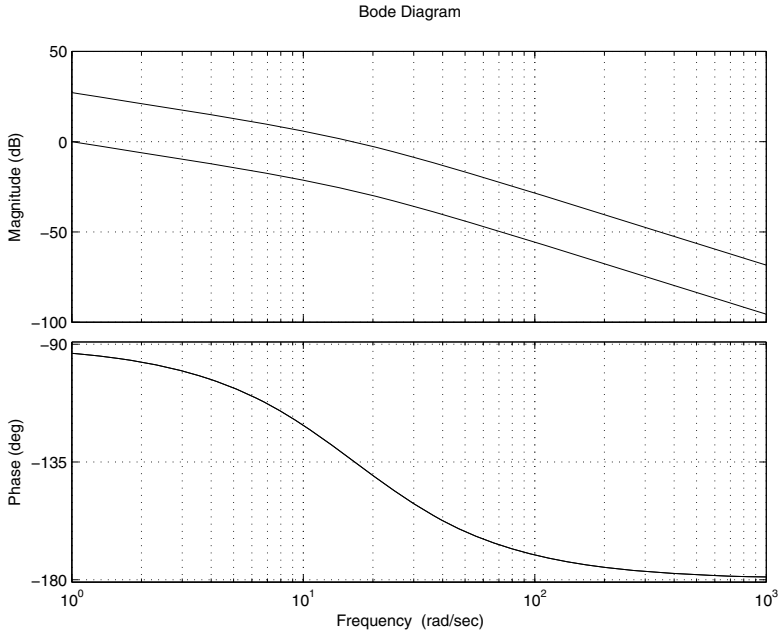
Fig. 5.38 Bode plot of $T(s)\frac{K}{s(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

We can check that with this gain, the closed loop system has a phase margin close to 45° and gain margin equal to infinity. This responds to the general specifications.

For the root locus method, we know that the proportional controller is unable to change the shape of the root locus and the only thing that we can do is to select an appropriate gain for the controller to get best performances. The root locus of the system is given by Fig. 5.39. From this figure with a gain $K = 8.35$ we get a damping ratio equal to 0.707 and the complex poles are $s_{1,2} = -8.33 \pm 8.35 j$. This gives a settling time at 5 % equal to 0.3601 s. The controller's gain in this case is:

$$K_P = \frac{K}{K_m} = \frac{8.35}{48.5} = 0.1722$$

The design of the proportional controller with the Bode method will give the same result as we did for the empirical method. It is important to notice that the methods (empirical method and Bode method and root locus method) give different gains. The step response with the two controllers is plotted in Fig. 5.40. The two methods (empirical and bode) give high value for the controller's gain, which corresponds to a smaller damping ratio and therefore an important overshoot.

As a comparative study of these methods we have the results of the Tab. 5.4. The error for a step input in all the case is equal to zero.
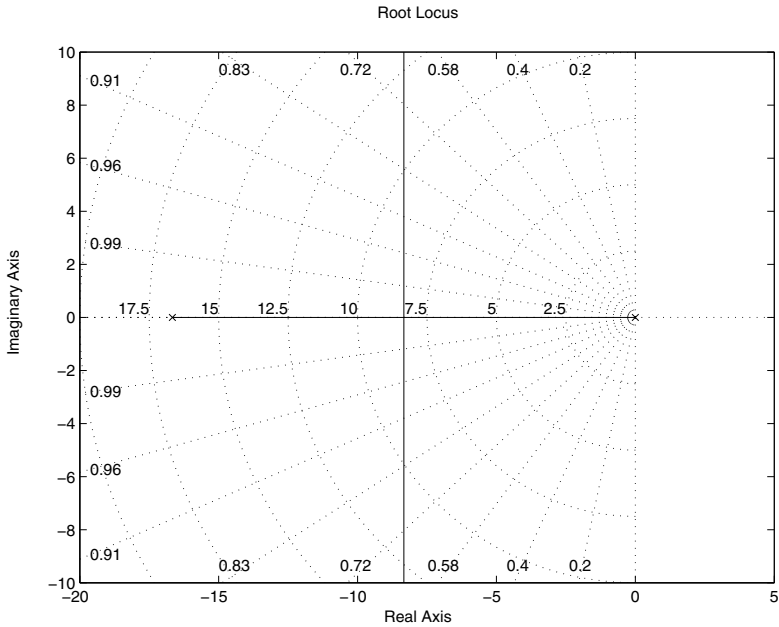
Root Locus

Fig. 5.39 Root locus of $T(s) = \frac{1}{s(\tau_m s+1)}$

Table 5.4 Comparative study of the design of P controller

Method	K_P	t_s	Overshoot	$\Delta\phi$	ΔG
Empirical	0.4723	0.3 s	23 %	45.6^o	∞
Root locus	0.1722	0.3 s	4 %	65.5^o	∞
Bode	0.4723	0.3 s	23 %	45.6^o	∞

5.6.2 Proportional and Integral Controller

Let us now focus on the design of the PI controller using the previous methods that gives the best performances for our dc motor kit. As for the proportional controller, the PI controller can not be designed using the time domain empirical method. While the frequency method can be used. It is important in this case that we can not use our procedure since we can not cancel the pole at the origin but placing the zero at -2 will give good performances. Using this, we get:

$$K_P = 0.0497, \text{ same computations as before}$$
$$K_I = 0.0994$$

The bode diagram of the open-loop transfer function of the compensated system is illustrated at the Fig. 5.41.
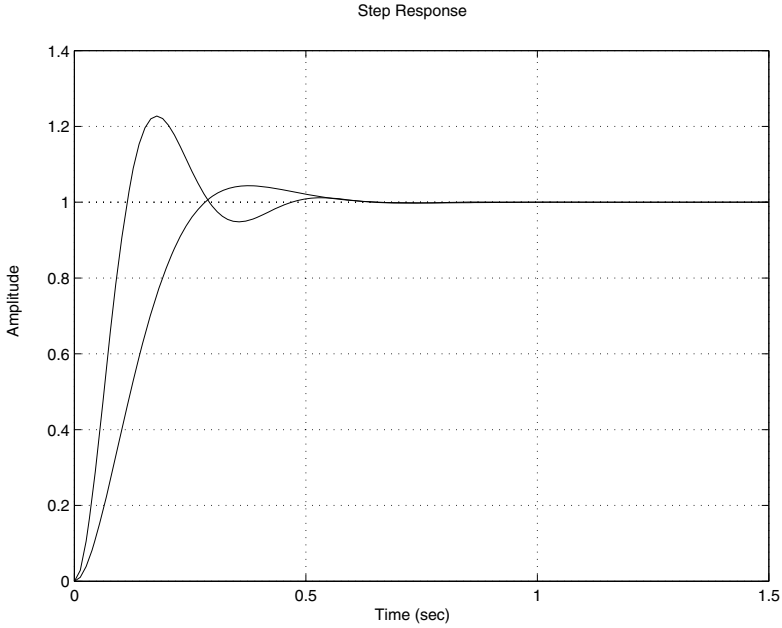
Step Response

Fig. 5.40 Step response of $F(s) = \frac{K_m K_P}{\tau_m s^2 + s + K_m K_P}$

With this controller we get a phase margin equal to 45° but the gain margin is close to zero.

Remark 5.6.1 *It is important to notice the approach used here to design the PI controller is a heuristic that I propose to overcome the problem with the previous procedure.*

If we place the zero at -3, the gain that gives the dominant poles $s_{1,2} = -5.23 \pm 5.95j$ is $\bar{K}_P = 22.9$, which gives $K_P = \frac{22.9}{48.5} = 0.4722$. From this we conclude that $K_I = K_P z = 1.4165$

The Bode method will give the same results as for the Zigeler-Nichols method and we don't repeat the computation again.

The closed-loop transfer function with the PI controller is given by:

$$F(s) = \frac{K_m K_P s + K_m K_I}{\tau_m s^3 + s^2 + K_m K_P s + K_m K_I}$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.43. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical. The settling time for the frequency methods is higher than that the one obtained by the root locus method.
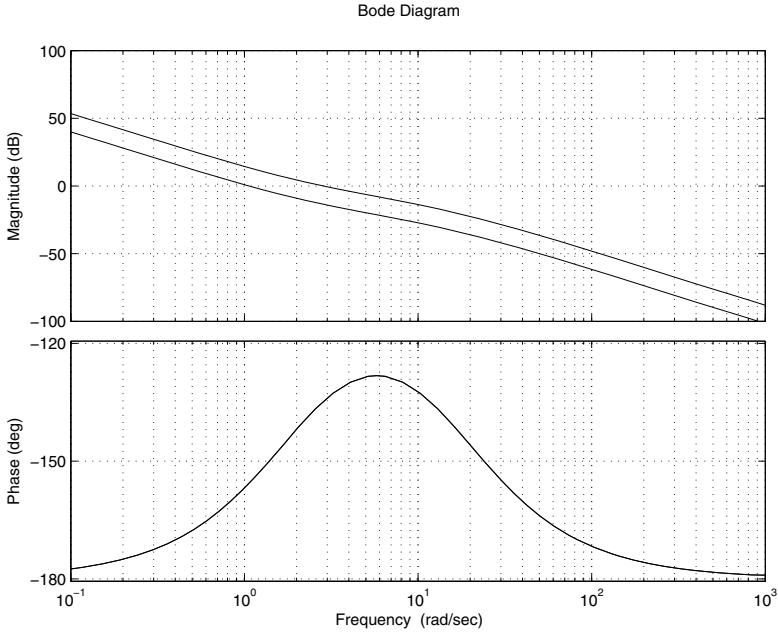
Fig. 5.41 Bode plot of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

5.6.3   Proportional and Derivative Controller

The PD controller can not be designed by any of the proposed Ziegler-Nichols methods. The only methods we can used for this controller are the root locus method and the Bode method. Let us firstly design this controller by the first method. For this controller, we can proceed by pole/zero cancellation or place the zero at the right of the pole of the system. The first case is easy and gives a first order while the second one gives an interesting case. It is important to notice that the damping ratio in this case will be close 1. This doesn't imply that there is no overshoot due the presence of the zero. We will design two cases.

Let the zero be at the position -30, placed at the left of the system pole. The first case case gives the dominant poles $s_{1,2} = -16 \pm 14.2j$ which corresponds to the gain $\bar{K}_P = 0.915$. This gives the gain $K_D = \frac{0.915}{48.5} = 0.0189$. The second gain is $K_P = K_D z = 0.0189 \times 30 = 0.5660$.

The second case case gives the dominant poles $s_{1,2} = -43.2 \pm 15j$ which corresponds to the gain $\bar{K}_P = 4.19$. This gives the gain $K_D = \frac{4.19}{48.5} = 0.0864$. The second gain is $K_P = K_D z = 0.0864 \times 30 = 2.5918$. As it will seen from the Fig. 5.45 this case will gives good performances at least in simulation.

For the design of the PD controller, let us assume that we want to assure a steady state error for a unit ramp input equal to 0.008. This corresponds to a gain $\bar{K}_P = 125$. The Bode diagram of $T(s) = \frac{\bar{K}_P}{s(\tau_m s+1)}$ is represented at the Fig. 5.44. The magnitude
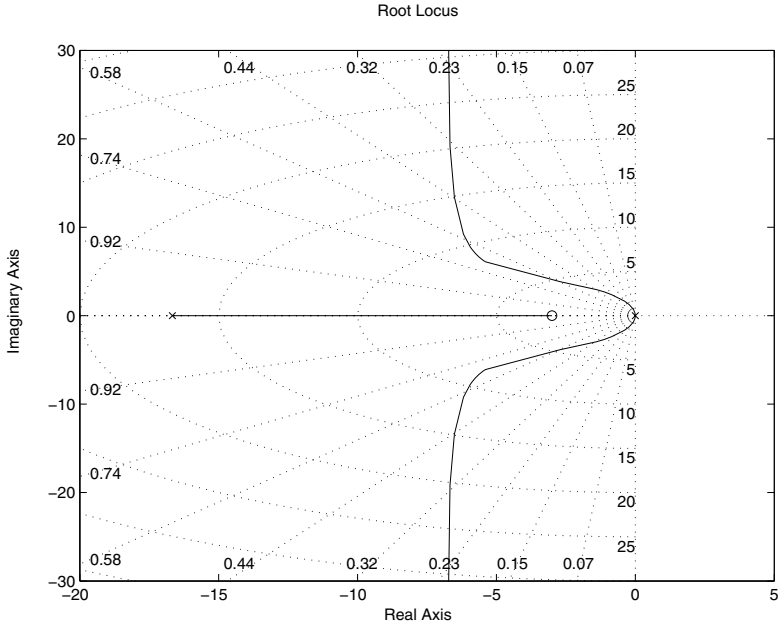
Root Locus

Fig. 5.42 Root locus of $T(s) = \frac{0.25s+1}{s^2(\tau_m s+1)}$

is equal to -20 db at the frequency $w_m = 144 \; rd/s$. The phase at this frequency is equal to -173° which corresponds to a phase margin equal to 7° and it is far from the desired phase margin.

The parameter τ_D is determined by:

$$\tau_D = \frac{10}{200} = 0.05$$

The parameters of the PD controller are given by:

$$K_P = \frac{125}{K_m} = 2.5773$$
$$K_D = K_P\tau_D = 2.5773 \times 0.05 = 0.1289$$

The phase margin of the compensated system is almost equal to 90°.

The closed-loop transfer function with the PD controller is given by:

$$F(s) = \frac{K_m K_D s + K_m K_P}{\tau_m s^2 + (1 + K_m K_D)s + K_m K_P}$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.45. As it can be seen that the two methods give two controllers that are different and the step response are also different. The settling time for the frequency methods is higher than that the one obtained by the root locus method.
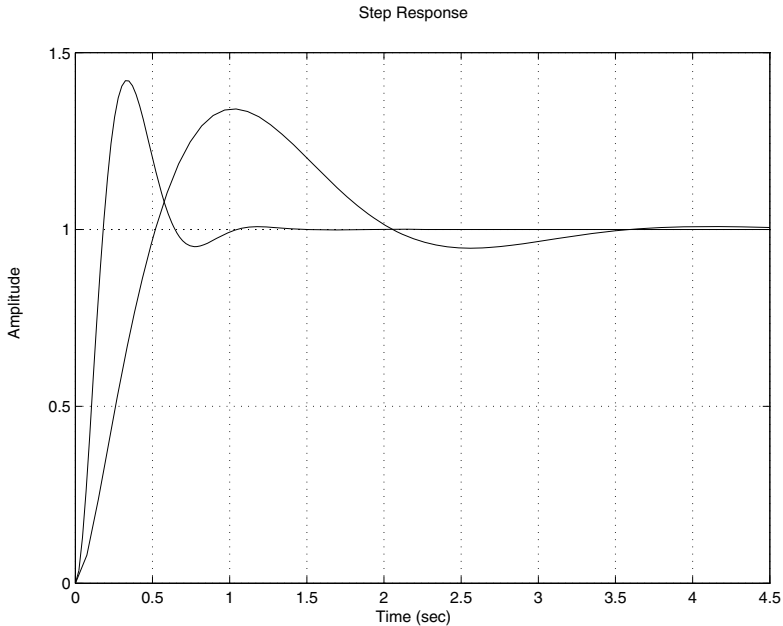
Fig. 5.43 Step of $F(s)$ with two controllers for two design methods

5.6.4   *Proportional Integral and Derivative Controller*

None of the heuristic methods proposed by Ziegler-Nichols can be used to design the PID controller. In the rest of this subsection we focus on the design of this controller using the root locus and Bode methods. The procedures we proposed previous can not be used here and we have to use another heuristic methods for this system.

For the root locus method, since the system has only one pole non equal to zero. The case that consists of placing the zeros between the two poles of the system is interesting since it can give short settling time.

If we place the two zeros of the controllers respectively at -13 and -15, the root locus of the system in this case is represented by Fig. 5.46

From the root locus we see that for the gain $\bar{K}_P = 1.43$, the dominant poles are:

$$s_{1,2} = -11.4 \pm 11.5j$$

If we refer to the procedure used for the design of the PID controller and the expression of the controller, we have:

$$a_1 = 13$$
$$a_2 = 15$$

Bode Diagram
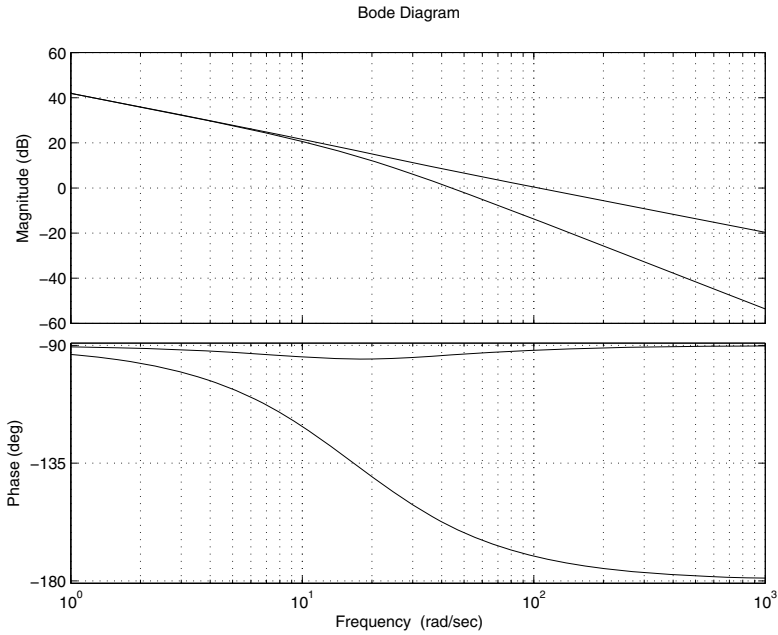
Fig. 5.44 Bode plot of $T(s)$ (compensated and non compensated system

From this we have:

$$K_D = \frac{1.43}{K_m} = 0.0295$$
$$K_P = K_D(a_1 + a_2) = 0.0295(13 + 15) = 0.8260$$
$$K_I = K_D a_1 a_2 = 5.7525$$

For the design of the PID using the Bode method, we will use the same idea of placing the zeros of the controller as we did for the root locus method. Also we would like to have a steady state error to a unit ramp equal to 0.01. To get such error a gain equal to $\bar{K}_P = 100$ is necessary for this purpose.

Now, if we place the two zeros of the controller respectively at -12 and -15, i.e.:

$$\tau_n = \frac{1}{15} = 0.0667$$
$$\tau_v = \frac{1}{12} = 0.0833$$

Step Response
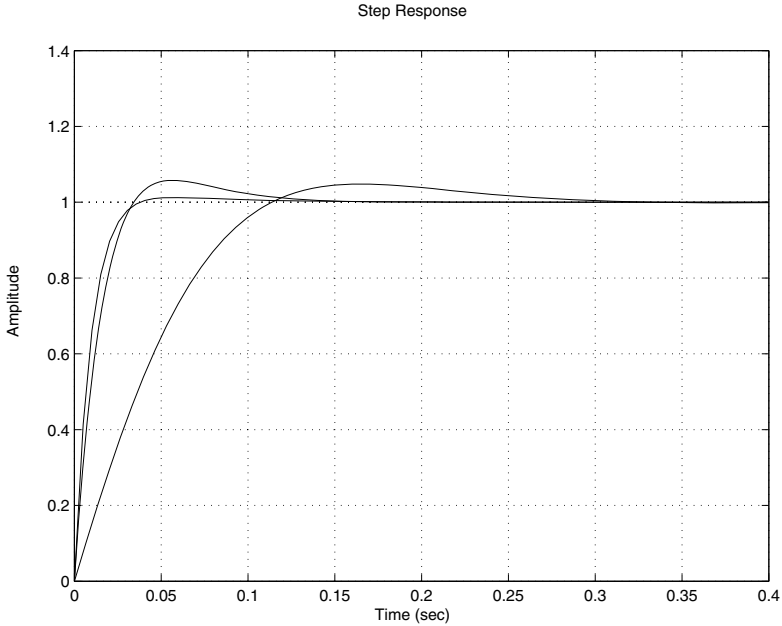
Fig. 5.45 Step of $F(s)$ with two controllers for two design methods

Using these data, we get:

$$\tau_i \;=\; \frac{K_m}{\bar{K}_P} = \frac{48.5}{100} = 0.4850$$

$$K_P \;=\; \frac{\tau_n + \tau_v}{\tau_i} = \frac{0.0667 + 0.0833}{0.4850} = 0.3093$$

$$K_I \;=\; \frac{1}{\tau_i} = \frac{1}{0.4850} = 2.0619$$

$$K_D \;=\; \frac{\tau_n \tau_v}{\tau_i} = \frac{0.0667 \times 0.0833}{0.4850} = 0.0115$$

The Bode diagram of the compensated system is represented at the Fig. 5.47. From this figure we conclude that the phase margin is equal 48°.

The closed loop transfer function of the compensated system is given by:

$$F(s) \;=\; \frac{K(K_D s^2 + K_P s + K_I)}{\tau_m s^3 + (1 + K_m K_D)s^2 + K_m K_P s + K_m K_I}$$

The step responses with the two controllers is illustrated by Fig. 5.48

5.6.5   Phase Lead Controller

Firstly, it is important to mention that this controller can not be designed with the empirical methods. The two other methods are still valid for the design of this con-
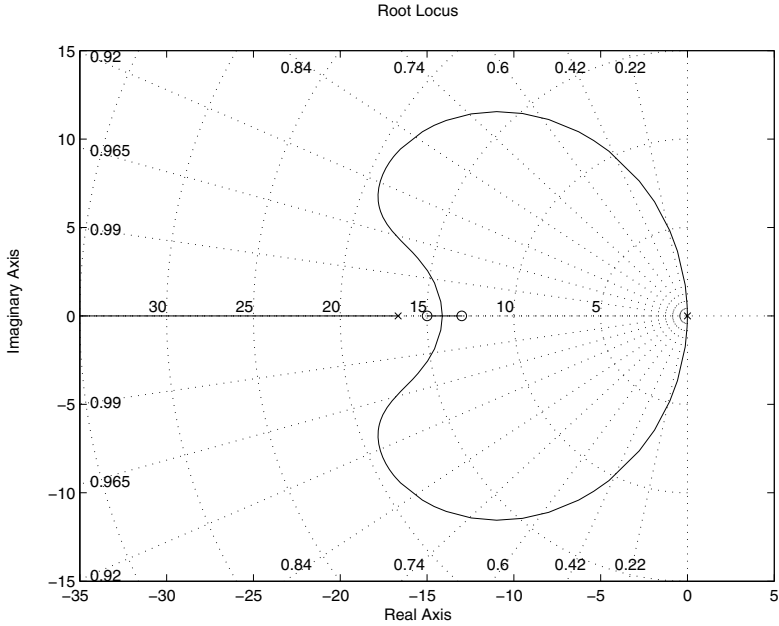
Root Locus

Fig. 5.46 Root locus of $T(s) = \frac{(\frac{1}{13}s+1)(\frac{1}{15}s+1)}{s^2(\tau_m s+1)}$,

troller. Let us firstly focus on the design the phase lead-controller using the root locus method. It is important to notice that the best settling time at 5 % with a proportional controller is about 0.36 s. With the phase lead controller we want to improve this time. Let the desired pole dominant with positive imaginary part be $s_d = -11.3 + 11.3j$ which corresponds to a settling time equal to 0.2655 s and an overshoot equal to 5 %. The phase of the system without the controller is given by:

$$\arg\left(\frac{48.5/0.06}{s_d(s_d + 16.6667)}\right) = 0 - 90 - 64.9830 = -153.9931$$

The phase lead controller must increase the phase with $180 - 153.9931 = 26.0069$ This implies:

$$\beta - \alpha = 26.0069$$

If we place the zero at -15, this implies that $\beta = 72.17^o$ and the pole at -20 gives an angle of 52.89^o. This gives a contribution of 19.27^o by the controller and which close to the desired one.

Bode Diagram
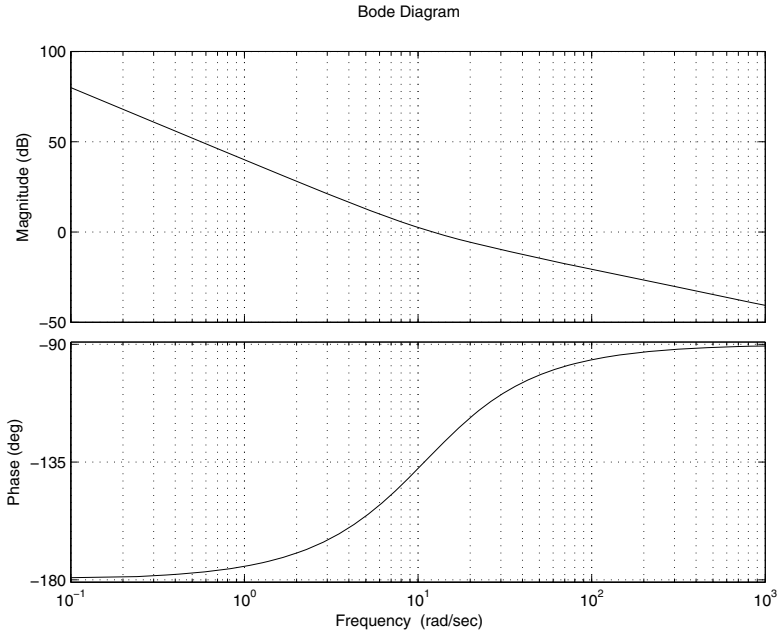
Fig. 5.47 Bode plot of $T(s) = \frac{100(\frac{1}{12}s+1)(\frac{1}{15}s+1)}{s^2(\tau_m s+1)}$

From this, we have:

$$\frac{1}{T} = 20$$

$$\frac{1}{aT} = 15$$

this gives $T = 0.05$ and $a = 1.3333$.

The root locus of the system with the phase lead controller is presented in the Fig. 5.49

The gain that gives the dominant poles is $\bar{K}_P = 10.8$, which gives a gain $K_P = 0.2227$ for the phase lead controller.

The closed-loop transfer function with the controller is given by:

$$F(s) = \frac{K_m K_P(aTs+1)}{T\tau_m s^3 + (\tau_m + T)s^2 + (1 + aTK_m K_P)s + K_m K_P}$$

The behavior of the step response of the system with this controller is illustrated in Fig. 5.51.

Using the Bode method, we design a controller that provides the following specifications:

1. stable system

2. steady state error to a unit ramp is less than 0.01
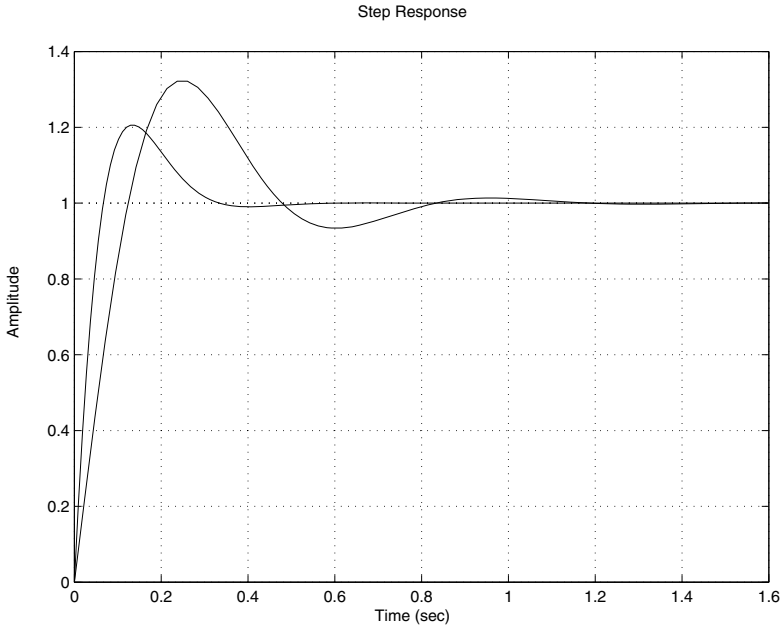
Step Response

Fig. 5.48 Step response of $F(s)$ with the two controllers

3. phase margin greater than 40°

4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.50. From this figure we have:

$$\Delta\phi = 23.1^o$$
$$\Delta G = \infty$$

For the design of phase lead controller notice that this controller should bring $45^o - 23.1^o = 22.9^o$, which gives:

$$a = \frac{1 + \sin(22.9)}{1 - \sin(22.9)} = 2.2740.$$

Using this values the magnitude will take the value $-20log(\sqrt{a}) = -3.5679$ at the frequency $w_m = 48.9 \ rd/s$. This implies:

$$T = \frac{1}{w_m \sqrt{a}} = 0.0136$$
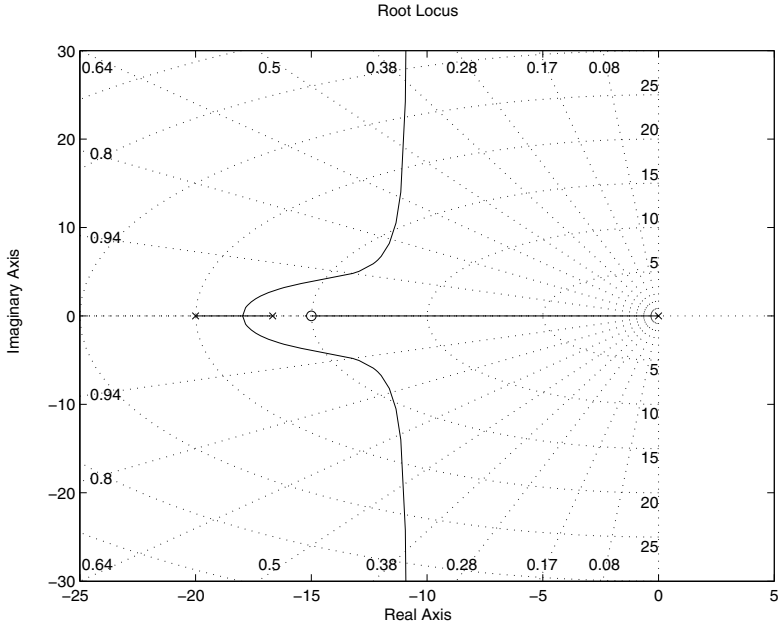
Root Locus

Fig. 5.49 Root locus of $T(s) = \frac{aTs+1}{s(\tau_m s+1)(Ts+1)}$

The phase lead controller is then given by the following transfer function:

$$C_1(s) \;=\; \frac{aTs+1}{Ts+1}$$

With this controller, the compensated system has:

$$\Delta\phi \;=\; 41.8^o$$
$$\Delta G \;=\; \infty$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.51. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.

5.6.6   Phase Lag Controller

As it was the case for the phase lead controller, the empirical methods can not help in the design of the phase lag controller. Here we will design this controller using the two other methods. For the root locus technique, we will assume that we want the following specifications:

1. stable system

2. a steady state error to a unit ramp input equal to 0.01

Bode Diagram
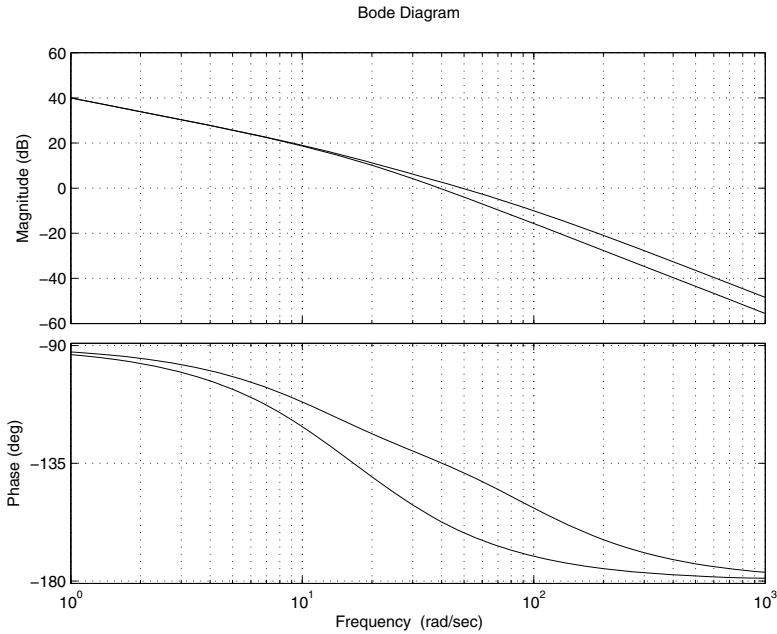
Fig. 5.50 Bode plot of $T(s)\frac{100}{s(\tau_m s+1)}$

3. an overshoot about 5 %

4. a settling time at 5 % equal to 0.36 s

Using the settling and the overshoot specifications, we conclude that the domi-
nant poles are $s_{1,2} = -8.33 \pm 8.35j$ and from the root locus of the system, we get
that the gain K_1 that gives these poles is $K_1 = 8.35$

Using now the steady state specifications, we conclude that K_2 is equal to 100.
From the values of these two gains, we get the parameter, a of the controller:

$$a = \frac{K_1}{K_2} = \frac{8.35}{100} = 0.0835$$

It is also important to notice that $a = \frac{p}{z}$, where p and z are respectively the pole
and the zero of the controller. Now, if we place the zero at -1.5, we get:

$$p = az = 0.1253$$

and since $p = \frac{1}{T}$, we get: $T = 7.9808$.
For the controller gain, it is given by:

$$K_P = \frac{100}{48.5} = 2.0619$$
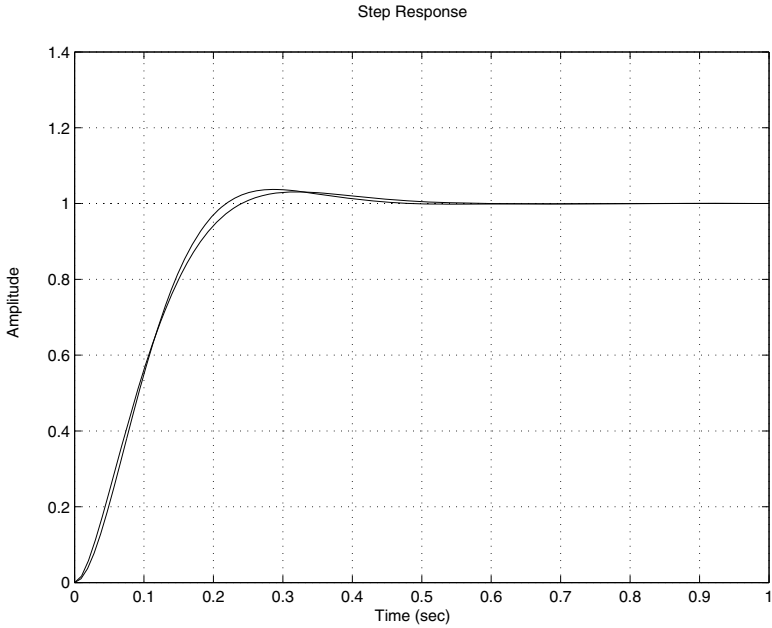
Fig. 5.51 Step of $F(s)$ with two controllers for two design methods

Finally, the transfer function of the controller is given by:

$$C(s) \ = K_P \frac{aT s + 1}{T s + 1}, a < 1$$

Using the Bode method, we design a controller that provides the following specifications:

1. stable system

2. steady state error to a unit ramp is less than 0.01

3. phase margin greater than 40°

4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead-lag controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.50. From this figure we have:

$$\Delta \phi \ = 23.1^o$$
$$\Delta G \ = \infty$$

The open loop transfer of the system with this controller is illustrated at the Fig. 5.52. The system will have a phase margin equal to 45° at the frequency $w_m = 16.9 \ rd/s$.

Bode Diagram
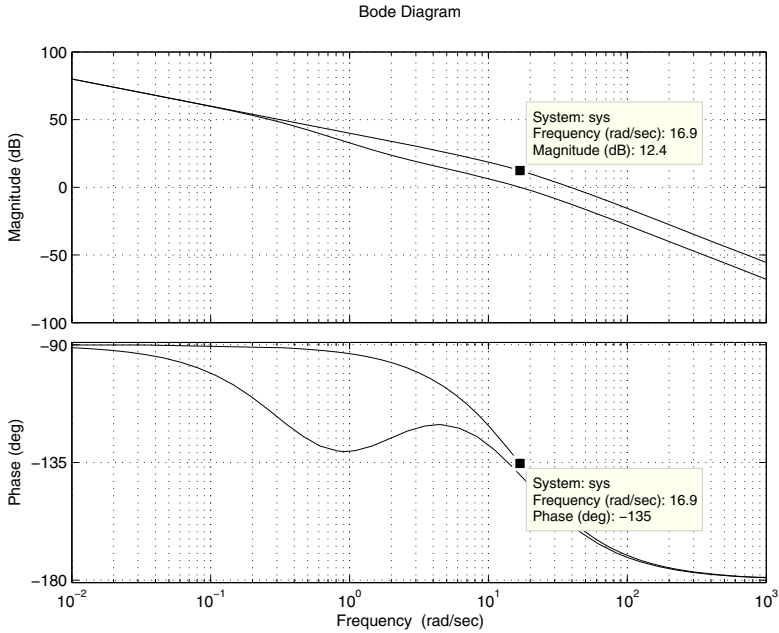
Fig. 5.52 Bode plot of $T(s)\frac{100}{(\tau_m s+1)}$

For the design of the phase lag controller, notice that at $w_m = 16.9\ rd/s$, the magnitude is equal to $12.4\ db$. Therefore,

$$a = 10^{\frac{-12.4}{20}} = 0.2399$$

The parameter T is given by:

$$T = \frac{10}{aw_m} = 2.4667$$

The controller phase-lag is given by the following transfer function:

$$C_2(s) = \frac{aTs+1}{Ts+1}$$

Combining now the two controllers, the open loop transfer function is given by:

$$T(s) = \frac{59.1716s + 100}{s(0.1480s^2 + 2.5267s + 1)}$$

The Bode diagram of this transfer function is represented at the 5.52. The specifications are:

$$\Delta\phi = 40.3^o$$
$$\Delta G = \infty$$

which are acceptable.

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.53. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.
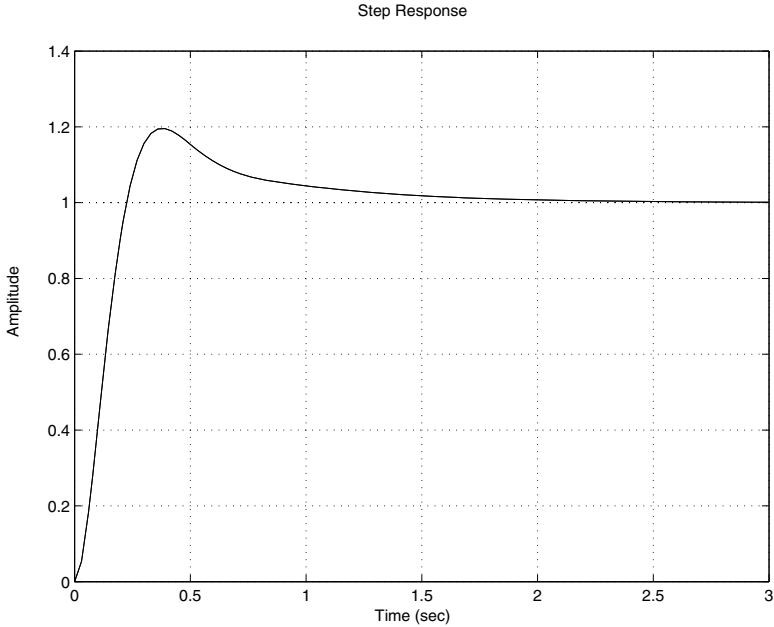
Fig. 5.53 Step of $F(s)$ with two controllers for two design methods

5.6.7 *Phase Lead-Lag Controller*

For this controller, we can use only the root-locus and the Bode methods to design it. Let us first of start the design of the controller using the root-locus method. It is important to notice that the best settling time at 5 % with a proportional controller is about 0.36 s. With the phase lead controller we want to improve this time. Let the desired pole dominant with positive imaginary part be $s_d = -11.5 + 11.6j$ which corresponds to a settling time equal to 0.27 s and an overshoot equal to 5 %. The phase of the system without the controller is given by:

$$\arg\left(\frac{48.5/0.06}{s_d(s_d + 16.6667)}\right) = 0 - 90 - 65.9917 = -155.9917$$

The phase lead controller must increase the phase with $180 - 155.9917 = 24.0083$

This implies:

$$\beta - \alpha = 24.0083$$

If we place the zero at -20, this implies that $\beta = 53.7676^o$ and the pole at -30 gives an angle of 52.89^o. This gives a contribution of 21.6788^o by the controller and which close to the desired one.

From this, we have:

$$\frac{1}{T_1} = 30$$

$$\frac{1}{a_1 T_1} = 20$$

this gives $T_1 = 0.0333$ and $a_1 = 1.5$.

For the phase lag controller design using the root locus technique, we will assume that we want the following specifications:

1. stable system

2. a steady state error to a unit ramp input equal to 0.01

3. an overshoot about 5 %

4. a settling time at 5 % equal to 0.27 s

Using the settling and the overshoot specifications, we conclude that the dominant poles are $s_{1,2} = -11.5 \pm 11.5j$ and from the root locus of the system, we get that the gain K_1 that gives these poles is $K_1 = 12.5$

Using now the steady state specifications, we conclude that K_2 is equal to 100.

From the values of these two gains, we get the parameter, a_2 of the controller:

$$a_2 = \frac{K_1}{K_2} = \frac{12.5}{100} = 0.125$$

It is also important to notice that $a_2 = \frac{p}{z}$, where p and z are respectively the pole and the zero of the controller. Now, if we place the zero at -0.1, we get:

$$p = a_2 z = 0.0125$$

and since $p = \frac{1}{T_2}$, we get: $T_2 = 80$.

For the controller gain, it is given by:

$$K_P = \frac{100}{48.5} = 2.0619$$

Finally, the transfer function of the controller is given by:

$$C(s) = K_P \frac{(a_1 T_1 s + 1)(a_2 T_2 s + 1)}{(T_1 s + 1)(T_2 s + 1)}, a_1 > 1, a_2 < 1$$

Using the Bode method, we design a controller that provides the following specifications:

1. stable system

Bode Diagram
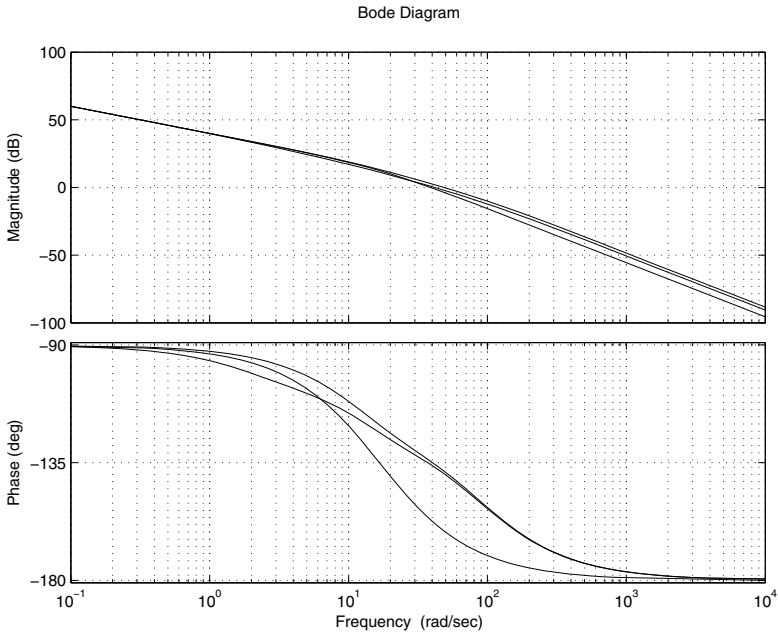
Fig. 5.54 Root locus of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

2. steady state error to a unit ramp is less than 0.01

3. phase margin greater than 40°

4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead-lag controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.55. From this figure we have:

$$\Delta\phi \ = 23.1^o$$
$$\Delta G \ = \infty$$

For the design of phase lead controller notice that this controller should bring $45^o - 23.1^o = 22.9^o$, which gives:

$$a_1 \ = \frac{1 + \sin(22.9)}{1 - \sin(22.9)} = 2.2740.$$

Using this values the magnitude will take the value $-20log(\sqrt{a_1}) = -3.5679$ at the frequency $w_m = 48.9 \ rd/s$. This implies:

$$T_1 \ = \frac{1}{w_m \sqrt{a_1}} = 0.0136$$

Bode Diagram
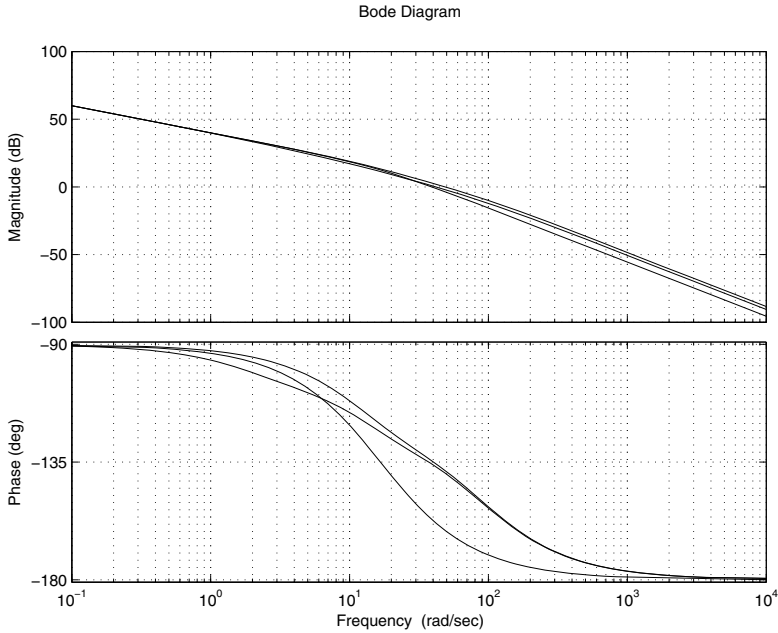
Fig. 5.55 Bode plot of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

The phase-lead controller is then given by the following transfer function:

$$C_1(s) = \frac{a_1 T_1 s + 1}{T_1 s + 1}$$

With this controller, the compensated system has:

$$\Delta\phi = 41.8^o$$
$$\Delta G = \infty$$

The open loop transfer of the system with this controller is illustrated at the Fig. 5.55. The system will have a phase margin equal to 45° at the frequency $w_m = 41.3\ rd/s$.

For the design of the phase lag controller, notice that at $w_m = 41.3\ rd/s$, the magnitude is equal to $2.13\ db$. Therefore,

$$a_2 = 10^{\frac{-2.13}{20}} = 0.7825$$

The parameter T_2 is given by:

$$T_2 = \frac{10}{a_2 w_m} = 0.3094$$

The controller phase-lag is given by the following transfer function:

$$C_2(s) = \frac{a_2 T_2 s + 1}{T_2 s + 1}$$

Combining now the two controllers, the open loop transfer function is given by:

$$T(s) \;=\; \frac{0.7467 s^2 + 27.2969 s + 100}{s(0.0003 s^3 + 0.02830 s^2 + 0.3830 s)}$$

The Bode diagram of this transfer function is represented at the 5.55. The specifications are:

$$\Delta\phi \;=\; 43.3^o$$
$$\Delta G \;=\; \infty$$

which are acceptable.

The closed-loop transfer function of the system with this controller is given by:

$$F(s) \;=\; \frac{K_m K_P \left(a_1 a_2 T_1 T_2 s^2 (a_1 T_1 + a_2 T_2) s + 1 \right)}{b_4 s^4 + b_3 s^3 + b_2 s^2 + b_1 s + b_0}$$

with $b_4 = \tau_m T_1 T_2$, $b_3 = \tau_m (T_1 + T_2) + T_1 T_2$, $b_2 = \tau_m + T_1 + T_2 + K_m K_P a_1 a_2 T_1 T_2$, $b_1 = 1 + K_m K_P (a_1 T_1 + a_2 T_2)$ and $b_0 = K_m K_P$.

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.56. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.
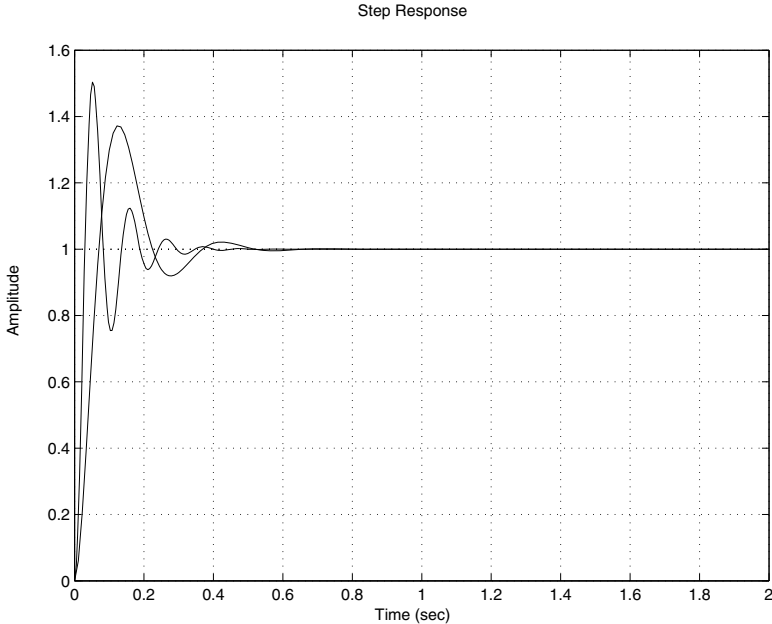
Fig. 5.56 Step of $F(s)$ with two controllers for two design methods

Remark 5.6.2 *From this section, and the previous two ones, we can conclude for a given system, the phase lead-lag controller can not be obtained by the multiplication of the of the phase lead controller and the phase lag controller transfer function designed separately.*

The implementation of the different algorithms we developed in this case study can be done as it will be done in the implementation part. Tab. 5.5 gives the different difference equations to program in each controller. To get these equations we have used the trapezoidal schema and by denoting the sampling period by T_s.

Table 5.5 Difference equations for the different controllers: dc motor kit

Controller	Algorithm
P	$u(k) = K_P e(k)$
PI	$u(k) = u(k-1) + ae(k) + be(k-1)$
	$a = K_P + \frac{K_I T_s}{2}, b = -K_P + \frac{K_I T_s}{2}$
PD	$u(k) = -u(k-1) + ae(k) + be(k-1)$
	$a = K_P + \frac{2K_D}{T_s}, b = K_P - \frac{2K_D}{T_s}$
PID	$u(k) = u(k-2) + ae(k) + be(k-1) + ce(k-2)$
	$a = K_P + \frac{K_I T_s}{2} + 2\frac{K_D}{T_s}, b = K_I T_s - \frac{4K_D}{T_s}, c = \frac{K_I T_s}{2} + \frac{2K_D}{T_s} - K_P$
Lead	$u(k) = -a_0 u(k-1) + be(k) + ce(k-1)$
	$a_0 = \frac{T_s-2T}{T_s+2T}, b = K_P \frac{T_s+2aT}{T_s+2T}, c = K_P \frac{T_s-2aT}{T_s+2T}$
Lag	$u(k) = -a_0 u(k-1) + be(k) + ce(k-1)$
	$a_0 = \frac{T_s-2T}{T_s+2T}, b = K_P \frac{T_s+2aT}{T_s+2T}, c = K_P \frac{T_s-2aT}{T_s+2T}$
Lead-Lag	$u(k) = -a_0 u(k-1) - bu(k-2) + ce(k) + de(k-1) + fe(k-2)$
	$a_0 = \frac{(T_s-2T_1)(T_s+2T_2)+(T_s+2T_1)(T_s-2T_2)}{(T_s+2T_1)(T_s+2T_2)}, b = \frac{(T_s-2T_1)(T_s-2T_2)}{(T_s+2T_1)(T_s+2T_2)},$
	$c = K_P \frac{(T_s+2a_1T_1)(T_s+2a_2T_2)}{(T_s+2T_1)(T_s+2T_2)}, d = K_P \frac{(T_s-2a_1T_1)(T_s+2a_2T_2)+(T_s+2a_1T_1)(T_s-2a_2T_2)}{(T_s+2T_1)(T_s+2T_2)},$
	$f = K_P \frac{(T_s-2a_1T_1)(T_s-2a_2T_2)}{(T_s+2T_1)(T_s+2T_2)}$

5.7 Conclusion

Practical systems when designed need in general the design of a controller that improves the performances of such systems. The performances give an idea on the transient and transient regimes. Mostly, the overshoot, the settling time, the steady state error are considered as for the design of controllers. This chapter covers the design of the classical controllers like proportional, integral and derivative actions. Procedures using the empirical methods, root-locus technique and Bode plot technique are proposed and illustrated by numerical examples.

5.8   Problems

1. In this problem we consider the control of a small satellite. The mathematical model of this dynamical system is given by:

$$G(s) = \frac{\Theta(s)}{U(s)} = \frac{k}{s^2}$$

where $\Theta(s)$ is the angle to be controlled, $U(s)$ is the force to apply to the satellite and $k = 2$ is the gain of the satellite that depends on many parameters of the system.

Using the three techniques developed in this chapter to design the controller that gives the best performances and stabilizes the system.

2. Consider the following dynamical system:

$$G(s) = \frac{4}{s(0.1s + 1)(s - 1)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

3. Consider the following dynamical system:

$$G(s) = \frac{4}{s(0.2s + 1)^2}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

4. Consider the following dynamical system:

$$G(s) = \frac{5}{s(0.1s + 1)(0.2s + 1)}$$

Design a controller that assures the following performances:

(a) stable system
(b) steady state error to a unit ramp equal to 0.1
(c) settling time at 5 % less than 1 s
(d) overshoot less than 5 %

5. A dynamical system is described by the following dynamics:

$$G(s) = \frac{10}{(s + 1)(s + 5)(s + 10)}$$

Using the Ziegler-Nichols methods design the different controllers that we can design for this system and compare their performances

Using now the root locus and Bode methods design the controllers that gives good performances for this system. Make a comparative study of these controllers.

6. Consider a dynamical system with the following dynamics:

$$G(s) = \frac{5(s + 2)}{s(s + 1)(s + 5)(s + 10)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

7. A dynamical system is described by the following transfer function:

$$G(s) = \frac{0.4(0.2s + 1))}{(0.1s + 1)(0.4s + 1)(0.5s + 1)(0.8s + 1)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

Part IV

State Space Approaches

6
Analysis Based on State Space

After reading this chapter the reader will:

1. master the concept of state space and its relationship with the transfer function concept

2. be able to perform the analysis of any LTI system and determine the specifications that the system has

3. be able to compute the time response of any LTI system for a given input

4. be able to check the stability, the controllability and the observability of any dynamical system

6.1   Introduction

As it was seen before the state space representation is one of the ways to model dynamical systems (see [5, 4, 1]). Previously, we showed for instance that the model of the behavior of the position of a mechanical system driven by a dc motor can be described by the following state space representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0 \\ y(t) = Cx(t) \end{cases} \tag{6.1}$$

where $x(t)$ and $u(t)$ are respectively the state vector and the control input and the matrices A, B and C are given by:

$$A = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} & 0 \\ \frac{K_t}{J} & -\frac{b}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{L_m} \\ 0 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}.$$

The control input $u(t)$ is the voltage that we send to the dc motor and the state vector $x(t)$ is composed of:

- the current of the armature, $i(t)$

- the speed of the mechanical part, $w(t)$

- the position of the part, $\theta(t)$

Before going further in the design of this simple system, we should examine what are the actual specifications the system has and if they are not acceptable we must improve them by using appropriate design tools that will be presented later at the next chapter.

In this chapter we are interested to know what are the specifications that any system under study has and how we can proceed to determine them. We are mainly interested to know if the transient and the steady state regimes are acceptable or not for the considered system and if this system is stable, controllable and observable. The goal of this chapter is to show how we can check these properties and compute other performances. Some numerical examples are provided to reinforce the understanding of the different concepts developed in this chapter.

6.2   State Space Concept

Most of the systems that are computer controlled are in general considered to evolve continuously in time. To perform their analysis and design, they are sampled and converted to sampled data systems and then, the appropriate tools for analysis and synthesis are used. To show how this is done, let us consider a system described by the following state space equations:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + B_1 v(t) \\ y(t) = Cx(t) \end{cases} \tag{6.2}$$

where $x(t)$, $u(t)$ and $v(t)$ represent respectively the state, the control input and the disturbance of the system.

Remark 6.2.1 *Notice that the dynamics (6.2) can be rewritten as the one in (6.1) by redefining the control as* $\begin{bmatrix} u(t) \\ v(t) \end{bmatrix}$ *and the control matrix as* $\begin{bmatrix} B & B_1 \end{bmatrix}$

The solution of the state space equation is given by [1]:

$$x(t) = \Phi(t - t_0)x(t_0) + \int_{t_0}^{t} \Phi(t - \sigma)Bu(\sigma)d\sigma + \int_{t_0}^{t} \Phi(t - \sigma)B_1v(\sigma)d\sigma \quad (6.3)$$

where t_0 is the initial time and $\Phi(t)$ is the transition matrix ($\Phi(t) = \mathcal{L}^{-1}(s\mathbb{I} - A)^{-1}$ with $\mathcal{L}$ is the Laplace transform).

Let $t_0 = kT$ and $t = (k + 1)T$, where T is the sampling period of the system. With a zero-order-hold the control $u(\sigma)$ is supposed to be constant and equal to the value taken at period kT, i.e. $u(\sigma) = u(kT)$, for $kT < \sigma < (k + 1)T$. Defining $\Psi(T)$ and $W(kT)$ as:

$$\Psi(T) = \int_{kT}^{(k+1)T} \Phi\left((k + 1)T - \sigma\right) Bd\sigma \quad (6.4)$$

$$W(kT) = \int_{kT}^{(k+1)T} \Phi\left((k + 1)T - \sigma\right) B_1v(\sigma)d\sigma \quad (6.5)$$

we obtain the following state space representation in the discrete-time domain:

$$x((k + 1)T) = \Phi(T)x(kT) + \Psi(T)u(kT) + W(kT) \quad (6.6)$$

$$y(kT) = Cx(kT) \quad (6.7)$$

After the choice of the sampling period, T, the matrices $\Phi(T)$ and $\Psi(T)$ become constant known matrices.

It is customary to use the representation $x(k)$ in place of $x(kT)$. Therefore, $x(k)$ means the vector $x(t)$ at time $t = kT$. The state space representation of a linear time invariant discrete system when the external disturbance is equal to zero for all $k \geq 0$ is given by:

$$\begin{cases} x(k + 1) = \Phi x(k) + \Psi u(k) \\ y(k) = Cx(k) \end{cases} \quad (6.8)$$

where $x(k)$, $u(k)$ and $y(k)$ are respectively the state vector, the input vector and the output vector at time $t = kT$, $k = 0, 1, 2, \cdots$ and T is the sampling period.

Remark 6.2.2 *In Eq. (6.8) we omit to make the matrices Φ and Ψ depend on the sampling period T since it is fixed and these matrices are constant.*

The more general form of the discrete-time state space representation is given by:

$$\begin{cases} x(k + 1) = Gx(k) + Hu(k) \\ y(k) = Cx(k) + Du(k) \end{cases} \quad (6.9)$$

where $G = \Phi$, $H = \Psi$, C and D are constant matrices with appropriate dimensions.
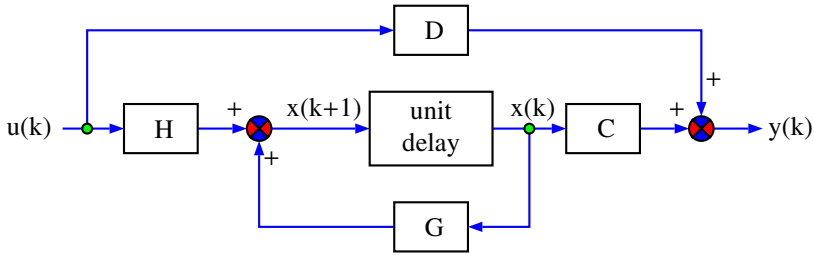
Fig. 6.1 Block diagram of discrete-time linear system

The block diagram of this system is illustrated by Fig. 6.1. Notice that the state at the time instant kT is obtained from the one at $kT + 1$ by a delay of one period time, T.

Example 6.2.1 *Consider a system with the following dynamics for the output y(t):*

$$Y(s) = \frac{10}{s(10s + 1)} U(s) + \frac{1}{s(10s + 1)} V(s)$$

where

- *u(t) is the reference input*

- *v(t) a unit-step disturbance*

Find the the difference equation of this system.

The solution of this question can be obtained using the formulas presented earlier. In fact, the corresponding differential equation of the system is:

$$10\ddot{y}(t) + \dot{y}(t) \; = 10u(t) + v(t)$$

By choosing $y(t) = x_1(t)$, $\dot{y}(t) = \dot{x}_1(t) = x_2(t)$ we get:

$$\ddot{y}(t) = \dot{x}_2(t) = -0.1x_2(t) + u(t) + 0.1v(t)$$

The state space equations are:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -0.1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} [u(t)] + \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} [v(t)]$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

First of all notice that the fast dynamics in this system is linked to the pole -0.1, which corresponds to a constant time $\tau = 10s$. Therefore an appropriate choice of the sampling period T is equal $1s$.

The transition matrix, $\Phi(t)$ of this system is given by:

$$\Phi(t) = \mathcal{L}^{-1}\left((sI - A)^{-1}\right) = \mathcal{L}^{-1}\left(\frac{1}{s(s+0.1)}\begin{bmatrix} s+0.1 & 1 \\ 0 & s \end{bmatrix}\right) =$$

$$\Phi(t) = \mathcal{L}^{-1}\left(\begin{bmatrix} \dfrac{1}{s} & \dfrac{1}{s(s+0.1)} \\ 0 & \dfrac{1}{s+0.1} \end{bmatrix}\right) = \begin{bmatrix} 1 & 10(1 - e^{-0.1t}) \\ 0 & e^{-0.1t} \end{bmatrix}$$

We also have:

$$\Psi(t) = \int_0^t \Phi(t-\sigma)Bd\sigma = \int_0^t \begin{bmatrix} 10(1 - e^{-0.1(t-\sigma)}) \\ e^{-0.1(t-\sigma)} \end{bmatrix} d\sigma$$

$$= \begin{bmatrix} 10t - 100 + 100e^{-0.1t} \\ 10 - 10e^{-0.1t} \end{bmatrix}$$

Therefore:

$$\Phi(T) = \begin{bmatrix} 1 & 10(1 - e^{-0.1T}) \\ 0 & e^{-0.1T} \end{bmatrix},$$

$$\Psi(T) = \begin{bmatrix} 10T - 100 + 100e^{-0.1T} \\ 10 - 10e^{-0.1T} \end{bmatrix}.$$

Since $v(t) = 1$ for $t \geq 0$, then we have :

$$W(t) = \int_0^t \Phi(t-\sigma)B_1d\sigma = \begin{bmatrix} t - 10 + 10e^{-0.1t} \\ 1 - e^{-0.1t} \end{bmatrix}$$

$$W(T) = \begin{bmatrix} T - 10 + 10e^{-0.1T} \\ 1 - e^{-0.1T} \end{bmatrix}$$

If $T = 1$ second, then we obtain:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0.95 \\ 0 & 0.91 \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.48 \\ 0.95 \end{bmatrix}u_k + \begin{bmatrix} 0.05 \\ 0.1 \end{bmatrix}v_k$$

It is well known that a physical system may have many state space representations. Most of the cases, we consider the canonical forms (controllable from, observable form and the Jordan form). The one, we just presented, is the controllable form. The other forms will be developed here.

For the Jordan form, remark that:

$$\frac{1}{s(s+0.1)} = \frac{10}{s} - \frac{10}{(s+0.1)}$$

which implies that:

$$Y(s) = \left[\frac{10}{s} - \frac{10}{(s+0.1)}\right][U(s) + 0.1V(s)]$$

Letting now:

$$X_1(s) = \frac{10}{s} [U(s) + 0.1V(s)]$$

$$X_2(s) = \frac{10}{(s+0.1)} [U(s) + 0.1V(s)]$$

which implies:

$$\dot{x}_1(t) = 10u(t) + v(t)$$
$$\dot{x}_2(t) = -0.1x_2(t) + 10u(t) + v(t)$$
$$y(t) = x_1(t) - x_2(t)$$

From this we get the following state space representation:

$$\dot{x}(t) = Ax(t) + Bu(t) + B_1v(t)$$
$$y(t) = Cx(t)$$

with

$$A = \begin{bmatrix} 0 & 0 \\ 0 & -0.1 \end{bmatrix},$$

$$B = \begin{bmatrix} 10 \\ -10 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & -1 \end{bmatrix}$$

For the observable canonical form notice that from:

$$Y(s) = \frac{1}{s(10s+1)} [10U(s) + V(s)]$$

we get:

$$10s^2 Y(s) + sY(s) = 10U(s) + V(s)$$

From this we obtain:

$$Y(s) = \frac{1}{s}\left[-0.1Y(s) + \frac{1}{s}[U(s) + 0.1V(s)] \right]$$

By letting:

$$Y(s) = X_1(s)$$
$$X_2(s) = \frac{1}{s} [U(s) + 0.1V(s)]$$
$$X_1(s) = \frac{1}{s} [-0.1Y(s) + X_2(s)]$$

that give in turn:

$$\dot{x}_1(t) = -0.1x_1(t) + x_2(t)$$
$$\dot{x}_2(t) = u(t) + 0.1v(t)$$
$$y(t) = x_1(t)$$

Finally we get the following description for our system:

$$\dot{x}(t) = Ax(t) + Bu(t) + B_1v(t)$$
$$y(t) = Cx(t)$$

where

$$A = \begin{bmatrix} -0.1 & 1 \\ 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Remark 6.2.3 *Another description can be obtained by letting:*

$$Y(s) = X_2(s)$$
$$X_1(s) = \frac{1}{s}[U(s) + 0.1V(s)]$$
$$X_2(s) = \frac{1}{s}[-0.1Y(s) + X_1(s)]$$

The computations of all the matrices for the discrete-time description can be done in a similar way as we did for the controllable form.

Remark 6.2.4 *In this example, for a given dynamical system, we developed a corresponding discrete-time state space representation for a given continuous-time state one. In some application we may have the recursive recurrent difference equation or its equivalent transfer function and we would like to establish the corresponding state space representation.*

Sometimes the model of the system under study can be given in $\mathscr{Z}$-transform with its transfer function between the output $Y(z)$ and the input $U(z)$ as follows:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{z^4 + 2z^2 + 3z + 4}{z^5 - 1} = \frac{z^4\left(1 + 2z^{-2} + 3z^{-3} + 4z^{-4}\right)}{z^5 - 1}$$

To establish the state space representation of this system, let us first of all rewrite the transfer function as follows:

$$G(z) = \frac{Y(z)}{X(z)}\frac{X(z)}{U(z)}$$

with

$$\frac{X(z)}{U(z)} = \frac{z^4}{z^5 - 1}$$

$$\frac{Y(z)}{X(z)} = 1 + 2z^{-2} + 3z^{-3} + 4z^{-4}$$

From the first relation we get:

$$z^5 X(z) - X(z) = z^4 U(z)$$

that gives in turn:

$$x(k + 1) = x(k - 4) + u(k)$$

An alternate choice for the state variable consists for instance to consider

$$
\begin{aligned}
x_1(k) &= x(k - 4) \\
x_2(k) &= x(k - 3) \\
x_3(k) &= x(k - 2) \\
x_4(k) &= x(k - 1) \\
x_5(k) &= x(k)
\end{aligned}
$$

From this we get:

$$
\begin{aligned}
x_1(k + 1) &= x(k - 3) = x_2(k) \\
x_2(k + 1) &= x(k - 2) = x_3(k) \\
x_3(k + 1) &= x(k - 1) = x_4(k) \\
x_4(k + 1) &= x(k) = x_5(k) \\
x_5(k + 1) &= x(k + 1) = x_1(k) + u(k)
\end{aligned}
$$

In matrix form we get:

$$x(k + 1) = Ax(k) + Bu(k)$$

with

$$
A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0
\end{bmatrix},
$$

$$
B = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
1
\end{bmatrix}
$$

From the other side, the second equation gives:

$$y(k) = x(k) + 2x(k - 2) + 3x(k - 3) + 4x(k - 4)$$

Using now, the definition of the state variables, we get:

$$y(k) = Cx(k)$$

with

$$C = \begin{bmatrix} 4 & 3 & 2 & 0 & 1 \end{bmatrix}.$$

This idea can be generalized and for this purpose let:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_{n-1}z^{n-1} + b_{n-2}z^{n-2} + \cdots + b_0}{z^n + a_{n-1}z^{n-1} + \cdots + a_0}$$

$$= \frac{z^{n-1}\left(b_{n-1} + b_{n-2}z^{-1} + b_{n-3}z^{-2} + \cdots + b_0z^{-(n-1)}\right)}{z^n + a_{n-1}z^{n-1} + \cdots + a_0}$$

This can be rewritten as follows:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{X(z)}{U(z)}\frac{Y(z)}{X(z)}$$

with

$$\frac{X(z)}{U(z)} = \frac{z^{n-1}}{z^n + a_{n-1}z^{n-1} + \cdots + a_0}$$

$$\frac{Y(z)}{X(z)} = b_{n-1} + b_{n-2}z^{-1} + b_{n-3}z^{-2} + \cdots + b_0z^{-(n-1)}$$

Using the first relation we get:

$$x(k + n) + a_{n-1}x(k + n - 1) + \cdots + a_0x(k) = u(k + n - 1)$$

That we can rewrite as:

$$x(k + 1) + a_{n-1}x(k) + \cdots + a_0x(k - n + 1) = u(k)$$

Let now:

$$
\begin{aligned}
x_1(k) &= x(k - n + 1) \\
x_2(k) &= x(k - n + 2) \\
x_3(k) &= x(k - n + 3) \\
&\vdots \\
x_n(k) &= x(k)
\end{aligned}
$$

which implies:

$$
\begin{aligned}
x_1(k + 1) &= x(k - n + 2) = x_2(k) \\
x_2(k + 1) &= x(k - n + 3) = x_3(k) \\
x_3(k + 1) &= x(k - n + 4) = x_4(k) \\
&\vdots \\
x_n(k + 1) &= x(k + 1) = -a_0x_1(k) - \cdots - a_{n-1}x_n(k) + u(k)
\end{aligned}
$$

The other relation implies:

$$
\begin{aligned}
y(k) &= b_{n-1}x(k) + b_{n-2}x(k-1) + \cdots + b_0 x(k-n+1) \\
&= b_0 x_1(k) + \cdots + b_{n-1}x_n(k)
\end{aligned}
$$

In matrix form we get for the two relations:

$$
\begin{aligned}
x(k+1) &= Ax(k) + Bu(k) \\
y(k) &= Cx(k)
\end{aligned}
$$

where

$$
A = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-a_0 & -a_1 & \cdots & -a_{n-2} & -a_{n-1}
\end{bmatrix},
$$

$$
B = \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix},
$$

$$
C = \begin{bmatrix} b_0 & \cdots & b_{n-1} \end{bmatrix}.
$$

When the degree of the numerator is equal to the one of the denominator of the transfer function, i.e:

$$
G(z) = \frac{b_n z^n + b_{n-1}z^{n-1} + \ldots + b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \ldots + a_1 z + a_0}
$$

we can firstly rewrite the transfer function as follows:

$$
G(z) = b_n + \frac{\bar{b}_{n-1}z^{n-1} + \ldots + \bar{b}_1 z + \bar{b}_0}{z^n + a_{n-1}z^{n-1} + \ldots + a_1 z + a_0}
$$

with

$$
\begin{aligned}
\bar{b}_{n-1} &= b_{n-1} - b_n a_{n-1} \\
\bar{b}_{n-2} &= b_{n-2} - b_n a_{n-2} \\
&\vdots \\
\bar{b}_0 &= b_0 - b_n a_0
\end{aligned}
$$

Following the same steps as before, we get the following state space description:

$$
\begin{aligned}
x(k+1) &= Ax(k) + Bu(k) \\
y(k) &= Cx(k) + b_n u(k)
\end{aligned}
$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & \cdots & -a_{n-2} & -a_{n-1} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} b_0 - a_0 b_n & \cdots & b_{n-1} - a_{n-1} b_n \end{bmatrix}.$$

For the observable canonical form, let us assume that the dynamics of the system with input $u(k)$ and output $y(k)$ is given by:

$$G(z) = \frac{b_{n-1} z^{n-1} + \ldots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \ldots + a_1 z + a_0}$$

To establish the observable canonical form, let us rewrite the dynamics as follows:

$$z^n Y(z) + a_{n-1} z^{n-1} Y(z) + \ldots + a_1 z Y(z) + a_0 Y(z) = b_{n-1} z^{n-1} U(z) + \ldots + b_1 z U(z) + b_0 U(z)$$

where $Y(z)$ and $U(z)$ and the $\mathcal{Z}$-transform of $y(k)$ and $u(k)$ respectively.

This relation can be rewritten as:

$$Y(z) = \frac{1}{z^n} \left[-a_{n-1} z^{n-1} Y(z) - \ldots - a_1 z Y(z) - a_0 Y(z) + b_{n-1} z^{n-1} U(z) + \ldots + b_1 z U(z) + b_0 U(z) \right]$$

$$= \frac{1}{z} \left[-a_{n-1} Y(z) + b_{n-1} U(z) + \frac{1}{z} \left[\ldots + \frac{1}{z} \left[-a_1 Y(z) + b_1 U(z) + \frac{1}{z} [-a_0 Y(z) + b_0 U(z)] \right] \right] \right]$$

Define $X_1(k_1), X_2(k), \cdots, X_{n-1}(k), X_n(k)$ by:

$$X_n(k) = \frac{1}{z} [-a_0 Y(z) + b_0 U(z)]$$

$$X_{n-1}(k) = \frac{1}{z} [-a_1 Y(z) + b_1 U(z) + X_n(z)]$$

$$\vdots$$

$$X_2(k) = \frac{1}{z} [-a_{n-2} Y(z) + b_{n-2} U(z) + X_3(z)]$$

$$X_1(k) = \frac{1}{z} [-a_{n-1} Y(z) + b_{n-1} U(z) + X_2(z)] = Y(z)$$

which gives:

$$
\begin{aligned}
x_n(k+1) &= -a_0 x_1(k) + b_0 u(k) \\
x_{n-1}(k+1) &= -a_1 x_1(k) + b_1 u(k) + x_n(k) \\
&\quad \vdots \\
x_2(k+1) &= -a_{n-2} x_1(k) + b_{n-2} u(k) + x_3(k) \\
x_1(k+1) &= -a_{n-1} x_1(k) + b_{n-1} u(k) + x_2(k)
\end{aligned}
$$

In matrix form we get:

$$
x(k+1) = \begin{bmatrix}
-a_{n-1} & 1 & 0 & \cdots & 0 \\
-a_{n-2} & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & 0 \\
-a_1 & 0 & 0 & \cdots & 1 \\
-a_0 & 0 & 0 & \cdots & 0
\end{bmatrix} x(k) + \begin{bmatrix}
b_{n-1} \\
b_{n-2} \\
\vdots \\
b_1 \\
b_0
\end{bmatrix} u(k)
$$

$$
y(k) = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} x(k)
$$

Remark 6.2.5 *It is important to notice that if we define* $X_1(k_1), X_2(k), \cdots, X_{n-1}(k),$ $X_n(k)$ *by:*

$$
\begin{aligned}
X_1(k) &= \frac{1}{z}[-a_0 Y(z) + b_0 U(z)] \\
X_2(k) &= \frac{1}{z}[-a_1 Y(z) + b_1 U(z) + X_n(z)] \\
&\quad \vdots \\
X_{n-1}(k) &= \frac{1}{z}[-a_{n-2} Y(z) + b_{n-2} U(z) + X_3(z)] \\
X_n(k) &= \frac{1}{z}[-a_{n-1} Y(z) + b_{n-1} U(z) + X_2(z)] = Y(z)
\end{aligned}
$$

which gives:

$$
\begin{aligned}
x_1(k+1) &= -a_0 x_1(k) + b_0 u(k) \\
x_2(k+1) &= -a_1 x_1(k) + b_1 u(k) + x_n(k) \\
&\quad \vdots \\
x_{n-1}(k+1) &= -a_{n-2} x_1(k) + b_{n-2} u(k) + x_3(k) \\
x_n(k+1) &= -a_{n-1} x_1(k) + b_{n-1} u(k) + x_2(k)
\end{aligned}
$$

In matrix form we get:

$$x(k+1) = \begin{bmatrix} 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & \cdots & 0 & -a_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_{n-1} \end{bmatrix} x(k) + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-2} \\ b_{n-1} \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix} x(k)$$

which gives another observable canonical form:

For more general transfer function with the following expression:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_n z^n + b_{n-1} z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}$$

This transfer function can be rewritten as follows:

$$G(z) = b_n + \frac{Y(z)}{U(z)} = b_n + \frac{(b_{n-1} - a_{n-1}b_n)z^{n-1} + \cdots + (b_1 - a_1 b_n)z + (b_0 - a_0 b_n)}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}$$

That gives the following state space description:

$$x(k+1) = \begin{bmatrix} -a_{n-1} & 1 & 0 & \cdots & 0 \\ -a_{n-2} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ -a_1 & 0 & 0 & \cdots & 1 \\ -a_0 & 0 & 0 & \cdots & 0 \end{bmatrix} x(k) + \begin{bmatrix} b_{n-1} - a_{n-1}b_n \\ b_{n-2} - a_{n-2}b_n \\ \vdots \\ b_1 - a_1 b_n \\ b_0 - a_0 b_n \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} x(k) + b_n u(k)$$

For the Jordan canonical form, notice that we can have poles with multiplicity equal to one or greater than one. Let us firstly treat the case of poles with multiplicity equal to one. For this purpose, let us assume that the transfer function that describes the system has $z_1, \cdots, z_n$

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_{n-1} z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_1 z^{n-1} + \cdots + a_n}$$

This transfer function can be rewritten as follows:

$$G(z) = \frac{K_1}{z - z_1} + \cdots + \frac{K_n}{z - z_n}$$

where

$$K_i = \lim_{z \to z_i} G(z)(z - z_i)$$

Define $X_i(z) = \frac{K_i}{z - z_i}$, we get:

$$x_i(k+1) = z_i x_i(k) + K_i u(k)$$

which gives also:

$$y(k) = x_1(k) + \cdots + x_n(k)$$

This gives the following state space description:

$$x(k+1) = \begin{bmatrix} z_1 & 0 & \cdots & 0 \\ 0 & z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & z_n \end{bmatrix} x(k) + \begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_n \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} x(k)$$

Remark 6.2.6 *It is important to notice that the gain K_i, $i = 1, 2, \cdots, n$, can also be put in the C matrix of the state space description.*

When the transfer function is given by:

$$G(z) = \frac{b_n z^n + b_{n-1} z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}$$

we can firstly proceed with a polynomial division which will give the following:
Then proceeding in the same way as we did before, and ntoticing that

$$Y(z) = \frac{(b_{n-1} - a_{n-1}b_n)z^{n-1} + \cdots + (b_1 - a_1 b_n)z + (b_0 - a_0 b_n)}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0} U(z) + b_n U(z)$$

$$= X_1(z) + \cdots + X_n(z) + b_n U(z)$$

This gives the following state space description:

$$x(k+1) = \begin{bmatrix} z_1 & 0 & \cdots & 0 \\ 0 & z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & z_n \end{bmatrix} x(k) + \begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_n \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} x(k) + b_n u(k)$$

Example 6.2.2 *In this example, we will show how to establish canonical forms for a dynamical system described by a transfer function. For this purpose, let us assume that we have a system with the following transfer function between the output $y(k)$ and the input $u(k)$:*

$$G(z) = \frac{Y(z)}{U(z)} = \frac{z + 2}{z^2 - 1.7z + 0.72}$$

- *controllable canonical form: to establish this form, let us rewrite the transfer function as follows:*

$$G(z) = \frac{Y(z)}{U(z)} = \frac{W(z)}{U(z)} \frac{Y(z)}{W(z)} = \frac{z + 2}{z^2 - 1.7z + 0.72}$$

with

$$\frac{W(z)}{U(z)} = \frac{z}{z^2 - 1.7z + 0.72}$$

$$\frac{Y(z)}{W(z)} = 1 + 2z^{-1}$$

From $\frac{W(z)}{U(z)} = \frac{z}{z^2 - 1.7z + 0.72}$, *we get:*

$$w(k+2) - 1.7w(k+1) + 0.72w(k) = u(k+1)$$

that we can rewrite as follows:

$$w(k+1) - 1.7w(k) + 0.72w(k-1) = u(k)$$

Let us now define:

$$x_1(k) = w(k-1)$$
$$x_2(k) = w(k)$$

which implies in turn:

$$x_1(k+1) = w(k) = x_2(k)$$
$$x_2(k+1) = w(k+1) = 1.7w(k) - 0.72w(k-1) + u(k)$$
$$= -0.72x_1(k) + 1.7x_2(k) + u(k)$$

In matrix form we have:

$$x(k+1) = \begin{bmatrix} 0 & 1 \\ -0.72 & 1.7 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$

Using now the second relation $\frac{Y(z)}{W(z)} = 1 + 2z^{-1}$ *and the definition of the state variables, we get:*

$$y(k) = w(k) + 2w(k-1) = 2x_1(k) + x_2(k) = \begin{bmatrix} 2 & 1 \end{bmatrix} x(k)$$

Finally, we get the following description:

$$x(k+1) = \begin{bmatrix} 0 & 1 \\ -0.72 & 1.7 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$
$$y(k) = \begin{bmatrix} 2 & 1 \end{bmatrix} x(k)$$

Remark 6.2.7 *It is important to notice that we can get another controllable form with respect to the first raw.*

- *Observable canonical form: from the transfer function we get:*

$$z^2 Y(z) - 1.7zY(z) + 0.72Y(z) = zU(z) + 2U(z)$$

that gives in turn:

$$Y(z) = \frac{1}{z^2} [1.7zY(z) - 0.72Y(z) + zU(z) + 2U(z)]$$
$$= \frac{1}{z} \left[1.7Y(z) + U(z) + \frac{1}{z} [-0.72Y(z) + 2U(z)] \right]$$

Let us define $X_1(k)$ and $X_2(z)$ as follows:

$$X_2(z) = \frac{1}{z}[-0.72Y(z) + 2U(z)]$$

$$X_1(z) = \frac{1}{z}[1.7Y(z) + U(z) + X_2(z)] = Y(z)$$

which gives in turn:

$$x_1(k+1) = 1.7x_1(k) + x_2(k) + u(k)$$
$$x_2(k+1) = -0.72x_1(k) + 2u(k)$$
$$y(k) = x_1(k)$$

Finally, we get the following state space description:

$$x(k+1) = \begin{bmatrix} 1.7 & 1 \\ -0.72 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k)$$

Remark 6.2.8 *It is important to notice that we can get another observable form with respect to the last column.*

- *Jordan canonical form: from the transfer function it can be verified that we have the following poles:*

$$z_1 = 0.8$$
$$z_2 = 0.9$$

Based on this, the transfer function can be decomposed as follows:

$$G(z) = \frac{C_1}{z - 0.8} + \frac{C_2}{z - 0.9}$$

where C_1 and C_2 are given by:

$$C_1 = \lim_{z \to 0.8} \frac{z+2}{z-0.9} = -28$$
$$C_2 = \lim_{z \to 0.9} \frac{z+2}{z-0.8} = 29$$

Now, let us define $X_1(z)$ and $X_2(z)$ as follows:

$$X_1(z) = \frac{C_1}{z - 0.8}$$
$$X_2(z) = \frac{C_2}{z - 0.9}$$

The output $Y(z)$ is given by:

$$Y(z) = X_1(z) + X_2(z)$$

Using these relations, we get:

$$x_1(k+1) = 0.8x_1(k) - 28u(k)$$
$$x_2(k+1) = 0.9x_2(k) + 29u(k)$$
$$y(k) = x_1(k) + x_2(k)$$

which gives the following state space description:

$$x(k+1) = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.9 \end{bmatrix} x(k) + \begin{bmatrix} -28 \\ 29 \end{bmatrix} u(k)$$
$$y(k) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(k)$$

Remark 6.2.9 *It is important to notice that we can get different state space descriptions either by putting the coefficient -28 and 29 at the C matrix and by permuting the poles 0.8 and 0.9.*

In some circumstances we need to compute the transfer function from the state space description. In the next lines, we will show how to obtain it. For this purpose consider the state space description (6.9). Applying the $\mathscr{Z}$-transform, we get:

$$zX(z) - zx(0) = GX(z) + HU(z)$$
$$Y(z) = CX(z) + DU(z)$$

where $X(z)$, $U(z)$ and $x(0)$ represent respectively the $\mathscr{Z}$-transform of $x(k)$ and $u(k)$ and the initial condition of the state vector.

From these relations we get:

$$Y(z) = \left[C\,[z\mathbb{I} - G]^{-1}\,H + D \right] U(z) + z\,[z\mathbb{I} - G]^{-1}\,x(0)$$

When the initial condition are equal to zero this condition becomes:

$$Y(z) = \left[C\,[z\mathbb{I} - G]^{-1}\,H + D \right] U(z)$$

Example 6.2.3 *To show how to compute the transfer function from a state space description, let us consider the following dynamics:*

$$x(k+1) = Gx(k) + Hu(k)$$
$$y(k) = Cx(k)$$

where

$$G = \begin{bmatrix} 0 & 1 \\ -0.1 & 0.2 \end{bmatrix},$$
$$H = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$
$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Using the formula for the transfer function we get:

$$
\begin{aligned}
G(z) &= \frac{Y(z)}{U(z)} = C\,[z\mathbb{I} - G]^{-1}\,H \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix} \left[z \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ -0.1 & 0.2 \end{bmatrix} \right]^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= \frac{1}{z\,(z - 0.2) + 0.1} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z - 0.2 & 1 \\ -0.1 & z \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= \frac{1}{z\,(z - 0.2) + 0.1} \\
&= \frac{1}{z^2 - 0.2z + 0.1}
\end{aligned}
$$

Example 6.2.4 *As another example, let us consider the different models in state space description we developed for the dc motor driving the small disk presented earlier and compute the corresponding transfer function. For this purpose, we assume that the external disturbance is absent. For this system, we will consider the output as the position, θ(t), and we should act on the voltage input, u(t), to move the output. The mathematical model of this system is given by the following transfer function:*

$$
G(s) = \frac{K}{s\,(\tau s + 1)}
$$

where K is the gain and τ is the constant time.
From this transfer function we get:

$$
\ddot{\theta}(t) + \frac{1}{\tau}\dot{\theta}(t) = \frac{K}{\tau}u(t)
$$

For the controllable form, we can choose:

$$
\begin{aligned}
x_1(t) &= \theta(t) = y(t) \\
x_2(t) &= \dot{\theta}(t)
\end{aligned}
$$

From this, we get:

$$
\begin{aligned}
\dot{x}_1(t) &= \dot{\theta}(t) = x_2(t) \\
\dot{x}_2(t) &= \ddot{\theta}(t) = -\frac{1}{\tau}x_2(t) + \frac{K}{\tau}u(t)
\end{aligned}
$$

That gives in matrix form:

$$
\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t)
\end{aligned}
$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

To compute the corresponding discrete-time form, let us compute the transition matrix $\phi(s)$. Using the controllable description, we have

$$\phi(s) = (s\mathbb{I} - A)^{-1} = \begin{bmatrix} s & -1 \\ 0 & s + \frac{1}{\tau} \end{bmatrix}^{-1} = \frac{1}{s\left(s + \frac{1}{\tau}\right)} \begin{bmatrix} s + \frac{1}{\tau} & 1 \\ 0 & s \end{bmatrix} = \begin{bmatrix} \frac{1}{s} & \frac{\tau}{s} - \frac{\tau}{s + \frac{1}{\tau}} \\ 0 & \frac{1}{s + \frac{1}{\tau}} \end{bmatrix}$$

which gives using the Laplace transform table:

$$\phi(t) = \begin{bmatrix} 1 & \tau\left[1 - e^{-\frac{t}{\tau}}\right] \\ 0 & e^{-\frac{t}{\tau}} \end{bmatrix}$$

$$\Psi(T) = \int_0^T \begin{bmatrix} 1 & 1 - e^{-\frac{T-\sigma}{\tau}} \\ 0 & e^{-\frac{T-\sigma}{\tau}} \end{bmatrix} \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix} d\sigma$$

$$= \begin{bmatrix} K\left[T - \tau\left[1 - e^{-\frac{T}{\tau}}\right]\right] \\ K\left[1 - e^{-\frac{T}{\tau}}\right] \end{bmatrix}$$

It is important to notice that the corresponding discrete-time description is not in controllable form.

For the controllable form with respect to the first row, we can choose:

$$x_1(t) = \dot{\theta}(t)$$
$$x_2(t) = \theta(t) = y(t)$$

From this, we get:

$$\dot{x}_1(t) = \ddot{\theta}(t) = -\frac{1}{\tau}x_2(t) + \frac{K}{\tau}u(t)$$
$$\dot{x}_2(t) = \dot{\theta}(t) = x_1(t)$$

that gives in matrix form:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & -\frac{1}{\tau} \\ 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{K}{\tau} \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix}.$$

The corresponding discrete-time form of this controllable form, can be obtained in a similar way.

For the observable form, notice that from the transfer function we have:

$$\Theta(s) = \frac{1}{s} \left[-\frac{1}{\tau} \Theta(s) + \frac{1}{s} \left[\frac{K}{\tau} U(s) \right] \right]$$

Let us choose:

$$X_1(s) = \frac{1}{s} \left[-\frac{1}{\tau} \Theta(s) + X_2(s) \right] = y(t)$$

$$X_2(s) = \frac{1}{s} \left[\frac{K}{\tau} U(s) \right]$$

From this, we get the following matrix form:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} -\frac{1}{\tau} & 1 \\ 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

To compute the corresponding discrete-time form, let us compute the transition matrix $\phi(s)$. Using the controllable description, we have

$$\phi(s) = (s\mathbb{I} - A)^{-1} = \begin{bmatrix} s + \frac{1}{\tau} & -1 \\ 0 & s \end{bmatrix}^{-1} = \frac{1}{s\left(s + \frac{1}{\tau}\right)} \begin{bmatrix} s & 1 \\ 0 & s + \frac{1}{\tau} \end{bmatrix} = \begin{bmatrix} \frac{1}{s+\frac{1}{\tau}} & \frac{\tau}{s} - \frac{\tau}{s+\frac{1}{\tau}} \\ 0 & \frac{1}{s} \end{bmatrix}$$

which gives using the Laplace transform table:

$$\phi(t) = \begin{bmatrix} e^{-\frac{t}{\tau}} & \tau\left[1 - e^{-\frac{t}{\tau}}\right] \\ 0 & 1 \end{bmatrix}$$

$$\Psi(T) = \int_0^T \begin{bmatrix} e^{-\frac{T-\sigma}{\tau}} & \tau\left[1 - e^{-\frac{T-\sigma}{\tau}}\right] \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix} d\sigma$$

$$= \begin{bmatrix} K\left[T - \tau\left[1 - e^{-\frac{T}{\tau}}\right]\right] \\ \frac{K}{\tau}T \end{bmatrix}$$

It is important to notice that the corresponding discrete-time description is not in observable form.

For the observable form with respect to the last column, we can choose:

$$X_1(s) = \frac{1}{s}\left[\frac{K}{\tau}U(s)\right]$$

$$X_2(s) = \frac{1}{s}\left[-\frac{1}{\tau}\Theta(s) + X_2(s)\right] = y(t)$$

From this, we get the following matrix form:

$$\dot{x}(t) \quad Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 0 \\ 1 & -\frac{1}{\tau} \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{K}{\tau} \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix}.$$

For the Jordan form, notice that from the transfer function we have:

$$\Theta(s) = \frac{K}{s}U(s) + \frac{-K}{s + \frac{1}{\tau}}U(s)$$

Let us choose:

$$X_1(s) = \frac{K}{s}U(s)$$

$$X_2(s) = \frac{-K}{s + \frac{1}{\tau}}U(s)$$

$$y(t) = x_1(t) + x_2(t)$$

From this, we get the following matrix form:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{\tau} \end{bmatrix},$$

$$B = \begin{bmatrix} K \\ -K \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

To compute the corresponding discrete-time form, let us compute the transition matrix $\phi(s)$. Using the controllable description, we have

$$\phi(s) = (s\mathbb{I} - A)^{-1} = \begin{bmatrix} s & 0 \\ 0 & s + \frac{1}{\tau} \end{bmatrix}^{-1} = \frac{1}{s\left(s + \frac{1}{\tau}\right)}\begin{bmatrix} s + \frac{1}{\tau} & 0 \\ 0 & s \end{bmatrix} = \begin{bmatrix} \frac{1}{s} & 0 \\ 0 & \frac{1}{s+\frac{1}{\tau}} \end{bmatrix}$$

which gives using the Laplace transform table:

$$\phi(t) = \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{t}{\tau}} \end{bmatrix}$$

$$\Psi(T) = \int_0^T \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{T-\sigma}{\tau}} \end{bmatrix}\begin{bmatrix} K \\ -K \end{bmatrix} d\sigma$$

$$= \begin{bmatrix} KT \\ -K\tau\left[1 - e^{-\frac{T}{\tau}}\right] \end{bmatrix}$$

A second Jordan form can be obtained. In fact if we can choose:

$$X_1(s) = \frac{-K}{s + \frac{1}{\tau}}U(s)$$

$$X_2(s) = \frac{K}{s}U(s)$$

$$y(t) = x_1(t) + x_2(t)$$

From this, we get the following matrix form:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix},$$

$$A = \begin{bmatrix} -\frac{1}{\tau} & 0 \\ 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} -K \\ K \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

In the next section, we will treat how we can compute the time response of the sampled data system when the input is fixed.

6.3   Time Response and Its Computation

Previously, we presented a method based on the transfer function concept to compute the time response of any system for a given input. In this section, we will develop another method that uses the state space description.

Consider the state difference equation :

$$x(k + 1) = Gx(k) + Hu(k)$$

Before giving the solution of this difference equation, let us show that

$$\phi(kT) = G^k = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z \right]$$

In fact, the z-transform of the transition matrix, $\phi(k) = \phi(kT)$, is given by:

$$\Phi(z) = \sum_{k=0}^{\infty} \phi(kT)z^{-k}$$

Pre-multiplying both side of this relation by $\phi(T)z$ $(G = \phi(T))$ and subtracting the result from this relation, we get:

$$\left[\mathbb{I} - \phi(T)z^{-1} \right] \Phi(z) = \mathbb{I}$$

which can be rewritten as:

$$\Phi(z) = \left[\mathbb{I} - \phi(T)z^{-1} \right]^{-1} = \left[z\mathbb{I} - \phi(T) \right]^{-1} z$$

Taking now the inverse z-transform on both sides of this relation, we get:

$$\phi(kT) = G^k = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z \right]$$

Another approach can be used to show this. In fact, the $\mathscr{Z}$-transform of the previous state decsription gives :

$$zX(z) - zx(0) = GX(z) + HU(z)$$
$$(zI - G)X(z) = zx(0) + HU(z)$$

Multiplying by $(zI - G)^{-1}$, we get:

$$X(z) = (zI - G)^{-1}zx(0) + (zI - G)^{-1}HU(z)$$

Taking now the $\mathscr{Z}$-inverse transform, we obtain:

$$x(k) = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z\right]x(0) + \mathscr{Z}^{-1}\left[(zI - G)^{-1}HU(z)\right]$$

Notice that $G^k = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z\right]$ is the transition matrix.

Finally, we get the following expression for the solution of the difference equation:

$$x(k) = G^k x(0) + \sum_{l=0}^{k-1} G^{k-l-1}Hu(l)$$

It is also important to note that the solution, can be obtained using a recursive approach. In fact, for $k = 0$, we have:

$$x(T) = Gx(0) + Hu(0)$$

and for $k = 1$ we have:

$$\begin{aligned}
x(2T) &= Gx(T) + Hu(T) \\
&= G\left[Gx(0) + Hu(0)\right] + Hu(T) \\
&= G^2 x(0) + GHu(0) + Hu(T)
\end{aligned}$$

For $k = (N - 1)T$, we have

$$x(NT) = Gx((N - 1)T) + Hu((N - 1)T)$$

For $k = 2$, we have

$$x(3T) = Gx(2T) + Hu(2T)$$

Substituting the $(N - 1)$ equations for $x((N - 1)T), x((N - 2)T)\cdots, x(T)$, we get:

$$x(NT) = G^N x(0) + \sum_{l=0}^{N-1} G^{N-l-1}Hu(l)$$

For $N = k$ we get:

$$x(kT) = G^k x(0) + \sum_{l=0}^{k-1} G^{k-l-1}Hu(l)$$

The characteristic equation is:

$$|zI - G| = 0$$

Recall that a discrete system is stable if and only if the roots of the characteristic equation lie inside the unit circle centered at the origin.

Example 6.3.1 *Find the solution of the following discrete-time system when* $u(k) =$ *1,* $k = 0, 1, 2, \cdots$

$$x(k + 1) = Gx(k) + Hu(k)$$

$$G = \begin{bmatrix} 0 & 1 \\ -0.16 & -1 \end{bmatrix}, \quad H = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad x(0) = \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The transition matrix is defined by :

$$\Phi(k) = G^k = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z\right]$$

Calculate $(zI - G)^{-1}$

$$(zI - G)^{-1} = \begin{bmatrix} z & -1 \\ 0.16 & z+1 \end{bmatrix} = \begin{bmatrix} \dfrac{z+1}{(z+0.2)(z+0.8)} & \dfrac{1}{(z+0.2)(z+0.8)} \\ \dfrac{-0.16}{(z+0.2)(z+0.8)} & \dfrac{z}{(z+0.2)(z+0.8)} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{4}{3}\left(\dfrac{z}{z+0.2}\right) - \dfrac{1}{3}\left(\dfrac{z}{z+0.8}\right) & \dfrac{5}{3}\left(\dfrac{z}{z+0.2}\right) - \dfrac{5}{3}\left(\dfrac{z}{z+0.8}\right) \\ -\dfrac{0.8}{3}\left(\dfrac{z}{z+0.2}\right) + \dfrac{0.8}{3}\left(\dfrac{z}{z+0.8}\right) & -\dfrac{1}{3}\left(\dfrac{z}{z+0.2}\right) + \dfrac{4}{3}\left(\dfrac{z}{z+0.8}\right) \end{bmatrix}$$

The transition matrix

$$\Phi(k) = G^k = \mathscr{Z}^{-1}\left[(zI - G)^{-1}z\right]$$

$$= \mathscr{Z}^{-1} \begin{bmatrix} \dfrac{4}{3}\left(\dfrac{z}{z+0.2}\right) - \dfrac{1}{3}\left(\dfrac{z}{z+0.8}\right) & \dfrac{5}{3}\left(\dfrac{z}{z+0.2}\right) - \dfrac{5}{3}\left(\dfrac{z}{z+0.8}\right) \\ -\dfrac{0.8}{3}\left(\dfrac{z}{z+0.2}\right) + \dfrac{0.8}{3}\left(\dfrac{z}{z+0.8}\right) & -\dfrac{1}{3}\left(\dfrac{z}{z+0.2}\right) + \dfrac{4}{3}\left(\dfrac{z}{z+0.8}\right) \end{bmatrix}$$

$$\Phi(k) = \begin{bmatrix} \dfrac{4}{3}(-0.2)^k - \dfrac{1}{3}(-0.8)^k & \dfrac{5}{3}(-0.2)^k - \dfrac{5}{3}(-0.8)^k \\ -\dfrac{0.8}{3}(-0.2)^k + \dfrac{0.8}{3}(-0.8)^k & -\dfrac{1}{3}(-0.2)^k + \dfrac{4}{3}(-0.8)^k \end{bmatrix}$$

We know that:

$$X(z) = (zI - G)^{-1}\left[zx(0) + HU(z)\right]$$
$$U(z) = \frac{z}{z - 1}$$

therefore

$$zx(0) + HU(z) = \begin{bmatrix} z \\ -z \end{bmatrix} + \begin{bmatrix} \dfrac{z}{z-1} \\ \dfrac{z}{z-1} \end{bmatrix} = \begin{bmatrix} \dfrac{z^2}{z-1} \\ \dfrac{-z^2 + 2z}{z-1} \end{bmatrix}$$

and finally

$$X(z) = (zI - G)^{-1}\left[zx(0) + HU(z)\right]$$

$$X(z) = \begin{bmatrix} \dfrac{(z^2 + 2)z}{(z + 0.2)(z + 0.8)(z - 1)} \\[4mm] \dfrac{(-z^2 + 1.84z)z}{(z + 0.2)(z + 0.8)(z - 1)} \end{bmatrix} = \begin{bmatrix} -\dfrac{17}{6}z \\ \dfrac{}{z + 0.2} + \dfrac{-\dfrac{22}{9}z}{z + 0.8} + \dfrac{\dfrac{25}{18}z}{z - 1} \\[4mm] \dfrac{3.4}{6}z \\ \dfrac{}{z + 0.2} + \dfrac{-\dfrac{17.6}{9}z}{z + 0.8} + \dfrac{\dfrac{7}{18}z}{z - 1} \end{bmatrix}$$

Finally, we get the desired result:

$$x(k) = \mathscr{Z}^{-1}\{X(z)\} = \begin{bmatrix} -\dfrac{17}{6}(-0.2)^k + \dfrac{22}{9}(-0.8)^k + \dfrac{25}{18} \\[4mm] \dfrac{3.4}{6}(-0.2)^k - \dfrac{17.6}{9}(-0.8)^k + \dfrac{7}{18} \end{bmatrix}$$

Most of the time the system under control evolves continuously in time and their outputs take continuous values. Their specifications are defined in a similar way as we did previously when using the transfer concepts. For more details on the specifications, we refer the reader to the appropriate chapter. The discrete-time description is obtained in a similar was that we did for previous examples.

6.4   Stability

Previously we presented methods to check if a linear time-invariant given system is stable or not. These methods are based on the transfer function concepts and due to Jury and Raible and the methods are referred in the literature to as Jury's criteria and Raible's criteria respectively. In this section we present another approach for the stability analysis that was developed by Lyapunov. This method is powerful since it can be applied to linear and nonlinear systems and it is referred in the literature to as second method of Lyapunov.

For the analysis of stability in the sense of Lyapunov, we assume that the system is unforced ($u(t) = 0, \forall t \geq 0$) and responds only to initial conditions. The second method of Lyapunov has the disadvantages that gives only sufficient condition only and it relies of the choice of a Lyapunov function which is more complex for nonlinear systems.

A linear discrete-time system $x(k + 1) = Gx(k)$, with $x(k)$ its solution at period k, is stable if it exists a scalar function $V(x(k))$, called Lyapunov function, that satisfies the following conditions:

1. $V(x(k))$ must be positive definite

2. and satisfying the following:

$$V(x(k)) \begin{cases} = 0 & \text{for } x = 0, \\ > 0 & \text{for } x \neq 0. \end{cases}$$

The variation of $V(x(k))$ between two consecutive values $(k + 1)$ and (k) of $x(k)$ must be negative definite, i.e.

$$\Delta V(x(k)) = V(x(k + 1)) - V(x(k))$$

that must satisfy the following:

$$\Delta V(x(k)) \begin{cases} = 0 & \text{for } x = 0, \\ < 0 & \text{for } x \neq 0. \end{cases}$$

For the choice of the Lyapunov function $V(x(k))$, there exist several possibilities to find an adequate function $V(x(k))$. For linear systems, we generally choose the following form:

$$V(x(k)) = x^\top(k)Px(k)$$

where P is an appropriate matrix with appropriate dimension.

1. In order for $V(x(k))$ to be positive definite, it is sufficient that P is a symmetric and positive-definite matrix.

2. Regarding the condition $\Delta V(x(k))$, since $x(k + 1) = Gx(k)$, we have :

$$\begin{aligned} \Delta V(x(k)) &= x^\top(k + 1)Px(k + 1) - x^\top(k)Px(k) \\ &= x^\top(k)\underbrace{\left[G^\top PG - P\right]}_{-Q}x(k) = -x^\top(k)Qx(k) \end{aligned}$$

One solution for $\Delta V(x(k))$ to be negative definite is that Q is symmetric and positive-definite matrix.

Theorem 6.4.1 *Consider a linear time-invariant system with the following description:*

$$x(k + 1) = Gx(k)$$

The equilibrium point $\tilde{x} = 0$ is asymptotically stable if and only if for any given symmetric and positive matrix Q, there exists a symmetric and positive-definite matrix P solution of the following:

$$G^\top PG - G = -Q \tag{6.10}$$

Then $V(x(k)) = x^\top(k)Px(k)$ is a Lyapunov, and $\Delta V(x(k)) = -x^\top(k)Qx(k)$.

Remark 6.4.1 *It is important to notice that the stability of linear systems depends only on the system itself and not on the inputs and this is shown by Eq. (6.10) since the matrix G represents this system.*

Proof: Let us prove the sufficiency only. For this purpose, let us assume the existence a symmetric and positive-definite matrix $P > 0$ that it is the unique solution of $A^\top PA - P = -Q$, for a given $Q > 0$, and consider the following Lyapunov function:

$$V(x) = x^\top Px$$

First of all notice that $V(x) > 0$ for all $x \neq 0$ and $V(0) = 0$.
The difference is given by:

$$\Delta V(x(k)) = V(x(k+1)) - V(x_k) = x^\top(k+1)Px(k+1) + x^\top(k)Px(k)$$
$$= x^\top(k)\left[A^\top PA - P\right]x(k) = -x^\top(k)Qx(k)$$

Since $Q > 0$ is symmetric and positive-definite matrix, then $\Delta V(x) < 0$ is negative-definite, which implies that the system is stable.

For the proof of the necessity we refer the reader to [7]   □

We can show the stability of a given system doesn't depend on the used description. Let us assume that we have a stable system and consider a transformation, T, that puts it in a Jordan form. For this purpose, notice that:

$$\bar{G} = T^{-1}GT$$

Since our system is stable, this means that for a given symmetric and positive-definite matrix Q, there exists a symmetric and positive-definite matrix P solution of the following Lyapunov equation:

$$G^\top PG - P = -Q$$

The new description will be stable if for a given symmetric and positive-definite matrix $\bar{Q}$ there exists a symmetric and positive definite matrix $\bar{P}$ such that the following holds:

$$\bar{G}^\top \bar{P}\bar{G} - \bar{G} = -\bar{Q}$$

Using now the expression of $\bar{G}$ we have:

$$T^\top G^\top T^{-\top} \bar{P} \bar{T}^1 GT - T^{-1}GT = -\bar{Q}$$

that gives

$$T^{-1}\left[G^\top PG - G\right]T = -\bar{Q}$$

From this we get:

$$G^\top PG - P = -T^{-1}\bar{Q}T = -Q$$

which is equivalent to the stability of the initial description.

Example 6.4.1 *Let us now consider a system with the following dynamics:*

$$x(k+1) = Gx(k)$$

with

$$G = \begin{bmatrix} 0.768 & -0.416 \\ 1.184 & 0.192 \end{bmatrix}$$

Let's choose the simplest symmetric and positive-definite matrix:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

we calculate : $Q = P - G^\top PG$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} - \begin{bmatrix} 0.768 & 1.184 \\ -0.416 & 0.192 \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} \begin{bmatrix} 0.768 & -0.416 \\ 1.184 & 0.192 \end{bmatrix}$$

From this, we obtain :

$$0.410 p_{11} - 0.909 p_{12} - 0.909 p_{21} - 1.402 p_{22} = 1$$
$$0.319 p_{11} + 0.853 p_{12} + 0.492 p_{21} - 0.227 p_{22} = 0$$
$$0.319 p_{11} + 0.492 p_{12} + 0.853 p_{21} - 0.227 p_{22} = 0$$
$$-0.173 p_{11} + 0.080 p_{12} + 0.080 p_{21} + 0.963 p_{22} = 1$$

which gives in turn:

$$P = \begin{bmatrix} 5.715 & -0.978 \\ -0.978 & 2.227 \end{bmatrix}$$

The determinant of P is equal to 11.771. *The matrix P is then symmetric and positive-definite and therefore, the system is stable in the sense of Lyapunov.*

Example 6.4.2 *Let us consider the following system for this example:*

$$G(z) = \frac{2}{(z - 0.1)(z - 0.2)} = \frac{Y(z)}{U(z)}$$

where $Y(z)$ and $U(z)$ represent respectively the $\mathscr{Z}$-transform of the output and input. The canonical forms for this system are:

- *controllable form: Letting $x_1(k)$ and $x_2(k)$ be defined as follows:*

$$x_1(k) = y(k)$$
$$x_2(k) = y(k + 1)$$

  and noticing that we have:

$$y(k + 2) - 0.3y(k + 1) + 0.02y(k) = 2u(k)$$

  we get:

$$x(k + 1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 \\ -0.02 & 0.3 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 2 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

- *observable form: Notice that we can write the following:*

$$Y(z) = \frac{1}{z}\left[0.3Y(z) + \frac{1}{z}[-0.02Y(z) + 2U(z)]\right]$$

Letting $x_1(k)$ and $x_2(k)$ be defined as follows:

$$X_1(z) = \frac{1}{z}[0.3Y(z) + X_2(z)] = Y(z)$$

$$X_2(z) = \frac{1}{z}[-0.02Y(z) + 2U(z)]$$

we get:

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

$$A = \begin{bmatrix} 0.3 & 1 \\ -0.02 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 2 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

- *Jordan form: Notice that we can write the following:*

$$Y(z) = \frac{-20}{z - 0.1}U(z) + \frac{20}{z - 0.2}U(z)$$

Letting $x_1(k)$ and $x_2(k)$ be defined as follows:

$$X_1(z) = \frac{1}{s - 0.1}U(s)$$

$$X_2(z) = \frac{1}{z - 0.2}U(s)$$

we get:

$$\dot{x}(k) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

$$A = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.2 \end{bmatrix},$$

$$B = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} -20 & 20 \end{bmatrix}.$$

Remark 6.4.2 *Let us show that a transformation T of a system description will not change the system stability. For this purpose, let the initial description be given by:*

$$x(k+1) = Ax(k) + Bu(k)$$

The characteristic equation for this system is given by:

$$\Delta(z) = det\,[z\mathbb{I} - A] = 0.$$

Now, if we use the transformation T, i.e.: $x(k) = Tz(k)$ to get:

$$z(k+1) = \bar{A}z(k) + \bar{B}u(k)$$

with $\bar{A} = T^{-1}AT$ and $\bar{B} = T^{-1}B$.

The corresponding characteristic equation is given:

$$\bar{\Delta}(s) = det\left[z\mathbb{I} - \bar{A}\right] = 0.$$

Using now the expressions of $\bar{A}$ and $\bar{B}$, we have:

$$\bar{\Delta}(s) = det\left[T^{-1}\,[z\mathbb{I} - A]\,T\right]$$
$$= det\,[z\mathbb{I} - A] = 0.$$

which is the same as for the original description. To get the last relation, we used the fact that the transformation is nonsingular (det(T) and det(T^{-1}) are not equal to zero)

In this example, we will consider the canonical description of the previous example and show that the stability of the system is not affected. We will do the analysis in the continuous-time.

- *controllability form*

$$\Delta(z) = det\,[z\mathbb{I} - A] = \left\|\begin{matrix} z & -1 \\ 0.02 & z - 0.3 \end{matrix}\right\| = s(s - 0.3) + 0.02 = 0.$$

- *observability form*

$$\Delta(z) \ = det\,[z\mathbb{I} - A] = \left\| \begin{matrix} z - 0.3 & -1 \\ -0.02 & z \end{matrix} \right\| = s(s - 0.3) + 0.02 = 0.$$

- *Jordan form*

$$\Delta(z) \ = det\,[z\mathbb{I} - A] = \left\| \begin{matrix} z - 0.1 & 0 \\ 0 & z - 0.2 \end{matrix} \right\| = (s - 0.1)(s - 0.2) = 0.$$

For these description we get the same poles and therefore, the stability is not affected the transformation we consider.

6.5   Controllability and Observability

The concepts of controllability and observability are important issues in modern control theory. These two concepts play an important role in the stabilization problem of any dynamical system. The controllability is in some sense related to the possibility of driving the state of the system into a particular state, like the origin for instance, by using an appropriate control signal in a finite time. Therefore, when a state is not controllable, then no way to reach such goal, and consequently no signal will be able to control the state. The fact that the state is not controllable will cause a problem if the system is not stable. The concept of observability is related to the possibility of observing, through output measurements, the state of a system that we may use for control purpose for instance. Therefore, when a state is not observable, the controller will never be able to determine the behavior of an unobservable state and consequenlty can not use it to stabilize the system. In the rest of this section we will show how to evaluate if a given system is controllable and observable or not.

Let us consider the following dynamics:

$$\begin{cases} x((k + 1)T) = Gx(kT) + Hu(kT) \\ y(kT) = Cx(kT) \end{cases} \tag{6.11}$$

with $x_k \in \mathbb{R}^{n \times 1}$, $G \in \mathbb{R}^{n \times n}$, $H \in R^{n \times 1}$ and $C \in \mathbb{R}^{1 \times n}$.

In the rest of this section we will see how to determine if a given dynamical system is controllable or not and we do the same for the observability. Let us firstly start by the controllability.

Definition 6.5.1 *The system (6.11) is state controllable if there exists a piecewise-constant control signal u(kT) defined over a finite sampling interval $0 \leq kT < nT$ such that starting from any initial state, the state x(kT) can be made zero for $kT \geq nT$.*

Definition 6.5.2 *If every state is controllable, then the system (6.11) is said to be completely state controllable.*

Definition 6.5.3 *A system*

$$x(k + 1) = Gx(k) + Hu(k), x(0) = x_0$$

is controllable provided that there exists a sequence of inputs $u(0), u(1), \cdots, u(N)$
with finite values that transfers the system from any initial state $x(0)$ *to any final*
state $x(N)$ *with N finite.*

In fact, notice that:

$$
\begin{aligned}
x(1) \ &= Gx(0) + Hu(0) \\
x(2) \ &= Gx(1) + Hu(1) \\
&= G^2x(0) + GHu(0) + Hu(1) \\
x(3) \ &= Gx(2) + Hu(2) \\
&= G\left[G^2x(0) + GHu(0) + Hu(1)\right] + Hu(2) \\
&= G^3x(0) + \left[H \ GH \ G^2H\right]\begin{bmatrix} u(2) \\ u(1) \\ u(0) \end{bmatrix}
\end{aligned}
$$

$$\vdots$$

$$
\begin{aligned}
x(N) \ &= G^Nx(0) + G^{N-1}Gu(0) + \cdots + GHu(N-2) + Hu(N-1) \\
&= G^Nx(0) + \left[H \ GH \ \cdots \ G^{N-1}H\right]\begin{bmatrix} u(N-1) \\ u(N-2) \\ \vdots \\ u(0) \end{bmatrix}
\end{aligned}
$$

Therefore, for given $x(0)$ and $x(N)$, we get:

$$x(N) - G^Nx(0) \ = \left[H \ GH \ \cdots \ G^{N-1}H\right]\begin{bmatrix} u(N-1) \\ u(N-2) \\ \vdots \\ u(0) \end{bmatrix}$$

Since $x(N) \in \mathbb{R}^n$, this algebraic equation will give a solution only if the rank of
the matrix

$$\left[H \ GH \ \cdots \ G^{N-1}H\right]$$

is equal to n.

This matrix in known as the controllability matrix and it is defined by:

$$\mathscr{C} = \left[H \ GH \ \cdots \ G^{n-1}H\right]$$

Theorem 6.5.1 *The system (6.11) is completely controllable if* $\mathscr{C}$ *is of rank n.*

Remark 6.5.1 *The controllability of a given depends only the pair* (G, H) *and*
doesn't depend on the system description we use.

Theorem 6.5.2 *The system controllability is invariant under an equivalent transformation of the system description.*

Proof To prove this, let us firstly consider that the system is described by:

$$\begin{cases} x(k+1) = Gx(k) + Hu(k) \\ y(k) = Cx(k) \end{cases}$$

The pair (G, H) is controllable if the rank of the controllability matrix:

$$\mathscr{C} = \begin{bmatrix} H & GH & \cdots & G^{n-1}H \end{bmatrix}$$

is equal to n.

Let us consider a transformation, $\eta(k) = Px(k)$ that put that the system description as follows:

$$\begin{cases} \eta(k+1) = \bar{G}\eta(k) + \bar{H}u(k) \\ y(k) = \bar{C}x(k) \end{cases}$$

with $\bar{G} = PGP^{-1}$, $\bar{H} = PH$ and $\bar{C} = CP^{-1}$.

The pair $(\bar{G}, \bar{H})$ is controllable if the rank of the controllability matrix:

$$\mathscr{C} = \begin{bmatrix} \bar{H} & \bar{G}\bar{H} & \cdots & \bar{G}^{n-1}\bar{H} \end{bmatrix}$$

is equal to n.

Replacing $\bar{G}$ and $\bar{H}$ by their expression and noticing that $\left(PGP^{-1}\right)^n = PGP^{-1}PGP^{-1}\cdots PGP^{-1} = PG^nP^{-1}$, we get:

$$\begin{aligned} \bar{\mathscr{C}} &= \begin{bmatrix} PH & PGP^{-1}PH & \cdots & PG^{n-1}P^{-1}PH \end{bmatrix} \\ &= P\begin{bmatrix} H & GH & \cdots & G^{n-1}H \end{bmatrix} \\ &= P\mathscr{C} \end{aligned}$$

Since P is nonsingular, its rank will not affect the results. Therefore the rank of $\bar{\mathscr{C}}$ is equal to the one of $\mathscr{C}$, which implies that the controllability is not affected by the equivalent transformation. $\qquad \square$

Theorem 6.5.3 *The following statements are equivalent:*

1. *the pair (A, B) is controllable;*

2. *the matrix of dimension $n \times n$*

$$W_c(k-1) = \sum_{l=0}^{k-1} A^l BB^\top \left(A^\top\right)^l$$

   is nonsingular;

3. *the controllability matrix $\mathscr{C} = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}$ has rank n;*

4. *the matrix $\begin{bmatrix} A - \lambda\mathbb{I} & B \end{bmatrix}$ is full rank (raw) for every eigenvalue, λ, of the matrix A;*

5. *moreover if all the eigenvalues of the matrix A are inside the unit circle, then the unique solution $W_c - AW_cA^\top = BB^\top$ is positive-definite and the solution is called the controllability gramian . The expression of the controllability gramian is given by:*

$$W_c = \sum_{l=0}^{\infty} A^l BB^\top \left(A^\top\right)^l$$

Let us focus on the observability of the system (6.11). First of all using the dual system, the observability of the original system can be seen as the controllability of its dual. The dual system of the system (6.11) is described by the following dynamics:

$$\eta(k+1) = A^\top \eta(k) + C^\top v(k) \qquad (6.12)$$
$$w(k) = B^\top \eta(k) \qquad (6.13)$$

The controllability of this system implies the observability of the system (refdynamics) and vice versa.

Definition 6.5.4 *The system (6.11) is said to be observable if every initial state $x(0)$ can be determined from the observation of the output over a finite k sampling periods.*

Definition 6.5.5 *The system (6.11) is completely observable is every state is observable.*

For simplicity, let us consider that the input is equal to zero for all $k \geq 0$. In this case, we have:

$$y(0) = Cx(0)$$
$$y(1) = Cx(1) = CGx(0)$$
$$y(2) = Cx(2) = CGx(1) = CG^2x(0)$$
$$y(3) = Cx(3) = CG^3x(0)$$
$$\vdots$$
$$y(N-1) = Cx(N-1) = CG^{N-1}x(0)$$

Notice that:

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N-1) \end{bmatrix} = \begin{bmatrix} C \\ CG \\ \vdots \\ CG^{N-1} \end{bmatrix} x(0)$$

Since $x(0) \in \mathbb{R}^n$, this algebraic equation will have a solution only when the matrix

$$
\begin{bmatrix}
C \\
CG \\
\vdots \\
CG^{N-1}
\end{bmatrix}
$$

has a rank equal to n.

Observability matrix is defined by:

$$
\mathscr{O} =
\begin{bmatrix}
C \\
CG \\
\vdots \\
CG^{N-1}
\end{bmatrix}
$$

Theorem 6.5.4 *The system (6.11) is completely observable if $\mathscr{O}$ is of rank n.*

Remark 6.5.2 *The controllability of a given depends only the pair (C, G) and doesn't depend on the system description we use.*

Theorem 6.5.5 *The system observability is invariant under an equivalent transformation of the system description.*

Proof: The proof of this theorem can be done either by using the dual system or by following the same steps as for the controllability. □

Theorem 6.5.6 *The following statements are equivalent:*

1. *the pair (A, C) is observable;*

2. *the matrix of dimension $n \times n$*

$$
W_o(n-1) = \sum_{l=0}^{n-1} \left(A^\top\right)^l C^\top C A^l
$$

is nonsingular;

3. *the observability matrix $\mathscr{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$ has rank n;*

4. *the matrix $\begin{bmatrix} A - \lambda\mathbb{I} \\ C \end{bmatrix}$ has full rank (column) for every eigenvalue, λ, of the matrix A;*

5. *moreover if the all the eigenvalues of the matrix A are inside the unit circle, then the unique solution of $W_o - A^\top W_o A = C^\top C$ is positive-definite., that is called the controllability gramien and its expression is given by:*

$$W_o = \sum_{l=0}^{\infty} \left(A^\top\right)^l C^\top A^l$$

Example 6.5.1 *In this example, we consider a system with the dynamics as in (6.11) with*

$$G = \begin{bmatrix} 1 & 0.905 \\ 0 & 0.819 \end{bmatrix}, H = \begin{bmatrix} 0.475 \\ 0.905 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Let us study the controllability and the observability of this system.

- *Controllability Matrix:*

$$\mathscr{C} = [H|GH] = \begin{bmatrix} 0.475 & 1.294 \\ 0.905 & 0.741 \end{bmatrix}.$$

The determinant of this matrix, $\det \mathscr{C} = -0.819$, which means that its rank, $\mathrm{rank}(\mathscr{C}) = 2$ therefore the system is completely controllable.

- *Observability matrix:*

$$\mathscr{O} = \begin{bmatrix} C \\ CG \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0.905 \end{bmatrix}$$

The determinant of this matrix, $\det \mathscr{O} = 0.905$, which means that its rank, $\mathrm{rank}(\mathscr{C}) = 2$ therefore the system is completely observable

Example 6.5.2 *In this example we consider a dynamical system with the following dynamics between the input $u(k)$ and output $y(k)$:*

$$G(z) = \frac{6z^3 + 5z^2 + 4z + 1}{z^3 - 6z^2 + 11z - 6}$$

Our objective is to study the controllability and the observability of this system and show that these properties are not affected by the state space description we used. We will restrict ourselves to the case of canonical forms.

Following the same steps as we did earlier to establish the canonical forms, we can establish the following:

$$x(k + 1) = Gx(k) + Hu(k)$$
$$y(k) = Cx(k) + Du(k)$$

where the matrices G, H, C and D are given in each form as:

- *controllable form:*

$$G = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 6 & -11 & 6 \end{bmatrix},$$

$$H = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 37 & -62 & 41 \end{bmatrix},$$

$$D = 6$$

- *observable form:*

$$G = \begin{bmatrix} 6 & 1 & 0 \\ -11 & 0 & 1 \\ 6 & 0 & 0 \end{bmatrix},$$

$$H = \begin{bmatrix} 41 \\ -62 \\ 37 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix},$$

$$D = 6$$

- *Jordan form:*

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 6 & 0 & 3 \end{bmatrix},$$

$$H = \begin{bmatrix} 8 \\ -77 \\ 110 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix},$$

$$D = 6$$

The controllability matrix in each form is given

- *controllable form:*

$$\mathscr{C} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 6 \\ 1 & 6 & 36 \end{bmatrix},$$

- *observable form:*

$$\mathscr{C} = \begin{bmatrix} 41 & 184 & 690 \\ -62 & -414 & -1778 \\ 37 & 246 & 1104 \end{bmatrix},$$

- *controllable form:*

$$\mathscr{C} = \begin{bmatrix} 8 & 8 & 8 \\ -77 & -154 & -308 \\ 110 & 330 & 990 \end{bmatrix},$$

The rank of the controllability matrix in each form is equal to 3 and therefore, the system is controllable and as it can be seen, the chosen form will not affect the controllability of the system.

The observability matrix in each form is given

- *controllable form:*

$$\mathscr{O} = \begin{bmatrix} 37 & -62 & 41 \\ 246 & -414 & 184 \\ 1104 & -1178 & 690 \end{bmatrix},$$

- *observable form:*

$$\mathscr{O} = \begin{bmatrix} 1 & 0 & 0 \\ 6 & 1 & 0 \\ 25 & 6 & 1 \end{bmatrix},$$

- *controllable form:*

$$\mathscr{O} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 4 & 9 \end{bmatrix},$$

The rank of the observability matrix in each form is equal to 3 and therefore, the system is observable and as it can be seen, the chosen form will not affect the observability of the system.

Previously we presented the canonical forms for single input single output systems. In the rest of this section we will cover how we can establish the multi-input multi-output case. For this purpose, let us assume that the system we consider is described by the following:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$ and $y(k) \in \mathbb{R}^p$; $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{p \times n}$,

To establish the canonical forms, we need to determine a transformation P such that:

$$x(k) = P\eta(k)$$

where P is a nonsingular matrix.

For the Jordan form, this matrix can be obtained in the following manners:

- eigenvalues of the matrix A are distinct and with multiplicity equal to one: For this case, assume that the eigenvalues of the matrix A are $\lambda_1, \cdots, \lambda_n$ (i.e: they are solution of the equation $z\mathbb{I} - A = 0$), and their corresponding eigenvectors

are $v_1, \cdots, v_n$ (i.e: they are solution of $Av_i = \lambda_i v_i, i = 1, \cdots, n$). For this case the matrix P is given by:

$$P = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

The new state description is obtained as follows. In fact using the transformation $x(k) = P\eta(k)$, we get:

$$\begin{cases} \eta(k + 1) = P^{-1}AP\eta(k) + P^{-1}Bu(k) \\ y(k) = CP\eta(k) \end{cases}$$

Since P is nonsingular, which means that P^{-1} exists, we get:

$$\begin{cases} \eta(k + 1) = \bar{A}\eta(k) + \bar{B}u(k) \\ y(k) = \bar{C}\eta(k) \end{cases}$$

with

$$\begin{aligned} \bar{A} &= P^{-1}AP \\ &= \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \\ \bar{B} &= P^{-1}B \\ \bar{C} &= CP \end{aligned}$$

- eigenvalues with multiplicity greater than one: For eigenvalues with multiplicity greater than one, the method of determining the matrix P is similar to the previous one, except the computation of the eigenvectors is different. To show how this works, let us assume that we have eigenvalue with multiplicity equal to n. The eigenvectors in this case are solution of the following:

$$\begin{aligned} Av_1 &= \lambda v_1 \\ Av_2 &= \lambda v_2 + v_1 \\ &\vdots \\ Av_n &= \lambda v_n + v_{n-1} \end{aligned}$$

where λ is the eigenvector of multiplicity equal to n of the matrix A.
The matrix P in this case is given by:

$$P = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

The new description is similar to the last case with the following structure for $\bar{A}$:

$$\bar{A} = P^{-1}AP$$
$$= \begin{bmatrix} \lambda & 1 & 0 & \cdots & 0 \\ 0 & \lambda & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda & 1 \\ 0 & 0 & \cdots & 0 & \lambda \end{bmatrix}$$

Example 6.5.3 *To show how we establish the Jordan form in case of distinct eigen-values with multiplicity equal to one. For this purpose, let us consider the following state description:*

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

with

$$A = \begin{bmatrix} 0 & 1 \\ -0.2 & 0.9 \end{bmatrix},$$
$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

The eigenvalues of the matrix A are 0.4 and 0.5. The corresponding eigenvectors are:

$$v_1 = \begin{bmatrix} 5 \\ 2 \end{bmatrix},$$
$$v_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix},$$

which are the solution of $Av_1 = 0.4v_1$ and $Av_2 = 0.5v_2$ respectively.
The corresponding matrix P is given by:

$$P = \begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix}.$$

Its inverse is given by:

$$P^{-1} = \frac{1}{1}\begin{bmatrix} 1 & -2 \\ -2 & 5 \end{bmatrix}.$$

The corresponding Jordan form is given by:

$$\begin{cases} \eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k) \\ y(k) = \bar{C}\eta(k) \end{cases}$$

with

$$\bar{A} = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.5 \end{bmatrix},$$

$$\bar{B} = \begin{bmatrix} 1 & -2 \\ -2 & 5 \end{bmatrix},$$

$$\bar{C} = \begin{bmatrix} 7 & 3 \\ 2 & 1 \end{bmatrix}$$

Example 6.5.4 *To show how we establish the Jordan form in case of eigenvalues with multiplicity greater than one, let us consider the following state description:*

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

with

$$A = \begin{bmatrix} 0 & 1 \\ -0.25 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

The eigenvalue of the matrix A is 0.5 with multiplicity equal to two. The corresponding eigenvectors are:

$$v_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix},$$

$$v_2 = \begin{bmatrix} 2 \\ 3 \end{bmatrix},$$

which are the solution of $Av_1 = 0.5v_1$ and $Av_2 = 0.5v_2 + v_1$ respectively.
The corresponding matrix P is given by:

$$P = \begin{bmatrix} 2 & 2 \\ 1 & 3 \end{bmatrix}.$$

Its inverse is given by:

$$P^{-1} = \begin{bmatrix} 0.57 & -0.5 \\ -0.25 & 0.5 \end{bmatrix}.$$

The corresponding Jordan form is given by:

$$\begin{cases} \eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k) \\ y(k) = \bar{C}\eta(k) \end{cases}$$

with

$$\bar{A} = \begin{bmatrix} 0.5 & 1 \\ 0 & 0.5 \end{bmatrix},$$

$$\bar{B} = \begin{bmatrix} 0.75 & -0.5 \\ -0.25 & 0.5 \end{bmatrix},$$

$$\bar{C} = \begin{bmatrix} 3 & 5 \\ 1 & 3 \end{bmatrix}$$

Let us now focus on how we determine the matrix P of the transformation $\eta(k) = Px(k)$ that put the system description in the controllable canonical form. We will firstly treat the case of single input, which means that $B \in \mathbb{R}^{n \times 1}$. Our objective is to determine a transformation:

$$\eta(k) = Px(k)$$

with P a nonsingular matrix.

Since P is nonsingular, this means that it has an inverse and $PP^{-1} = \mathbb{I}$. One way to construct this matrix is to use the controllability matrix which is given for this case by:

$$\mathscr{C} = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}.$$

If the system is controllable, then $\mathscr{C}$ is nonsingular and $\mathscr{C}^{-1}$ exists. Let the expression of this matrix be as follows:

$$\mathscr{C}^{-1} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}.$$

where q_i is a row of $\mathscr{C}^{-1}$.

The matrix P of the transformation that gives the controllable canonical form can be constructed as follows:

$$P = \begin{bmatrix} q_n \\ q_n A \\ \vdots \\ q_n A^{n-1} \end{bmatrix}.$$

By this construction of the matrix P, it is important now to show that it is non-singular, or to show that its rows are linearly independent which means that there exists scalars $a_0, \cdots, a_{n-1}$ such:

$$a_0 q_n + a_1 q_n A + \cdots + a_{n-1} q_n A^{n-1} = 0$$

with $a_0 = a_1 = \cdots = a_{n-1} = 0$

To show this, let us use the following relationship between $\mathscr{C}$ and its inverse:

$$\mathscr{C}^{-1}\mathscr{C} = \mathbb{I}$$

$$= \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}$$

From which we get:

$$\begin{bmatrix} q_1B & q_1AB & \cdots & q_1A^{n-1}B \\ q_2B & q_2AB & \cdots & q_2A^{n-1}B \\ \vdots & \vdots & \ddots & \vdots \\ q_nB & q_nAB & \cdots & q_nA^{n-1}B \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Using this, we obtain for the last row:

$$q_nB = q_nAB = \cdots = q_nA^{n-2}B = 0$$
$$q_nA^{n-1}B = 1$$

Now if we multiply from the right the following relation:

$$a_0q_n + a_1q_nA + \cdots + a_{n-1}q_nA^{n-1} = 0$$

by B, we get:

$$a_0q_nB + a_1q_nAB + \cdots + a_{n-1}q_nA^{n-1}B = 0$$

From this, we obtain:

$$a_{n-1} = 0$$

since

$$q_nB = q_nAB = \cdots = q_nA^{n-2}B = 0$$
$$q_nA^{n-1}B = 1$$

Using now the fact that $a_{n-1} = 0$, we get:

$$a_0q_n + a_1q_nA + \cdots + a_{n-2}q_nA^{n-2} = 0$$

Repeating the same idea and by multiplying this relation by AB, and by using this time the $n - 2$ row, we get:

$$a_{n-2} = 0$$

since

$$q_nB = q_nAB = \cdots = q_nA^{n-3}B = 0$$
$$q_nA^{n-2}AB = 1$$

Proceeding similarly for the rest of the term we can prove that $a_0 = a_1 = \cdots = a_{n-3} = 0$ and therefore the matrix P is nonsingular.

Let P^{-1} be given by:

$$P^{-1} = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

Notice that

$$PAP^{-1} = \begin{bmatrix} q_n \\ q_nA \\ \vdots \\ q_nA^{n-1} \end{bmatrix} A \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

$$= \begin{bmatrix} q_nA \\ q_nA^2 \\ \vdots \\ q_nA^n \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

$$= \begin{bmatrix} q_nAv_1 & q_nAv_2 & \cdots & q_nAv_n \\ q_nA^2v_1 & q_nA^2v_2 & \cdots & q_nA^2v_n \\ \vdots & \vdots & \ddots & \vdots \\ q_nA^nv_1 & q_nA^nv_2 & \cdots & q_nA^nv_n \end{bmatrix}$$

Using now the fact that:

$$PP^{-1} = \begin{bmatrix} q_n \\ q_nA \\ \vdots \\ q_nA^{n-1} \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix}$$

$$= \begin{bmatrix} q_nv_1 & q_nv_2 & \cdots & q_nv_n \\ q_nAv_1 & q_nAv_2 & \cdots & q_nAv_n \\ \vdots & \vdots & \ddots & \vdots \\ q_nA^{n-1}v_1 & q_nA^{n-1}v_2 & \cdots & q_nA^{n-1}v_n \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

we remark that the first $(n-1)$ rows of the matrix PAP^{-1} are identical to the last $(n-1)$ last rows of the matrix PP^{-1} and therefore equal to the identity matrix, while the last row of PAP^{-1} can be composed of any numbers. Therefore, the matrix PAP^{-1} has the controllable canonical form with respect to the last row.

To show that $\bar{B}$ has the following form:

$$\bar{B} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

notice that:

$$\bar{B} = PB = \begin{bmatrix} q_n \\ q_n A \\ \vdots \\ q_n A^{n-1} \end{bmatrix} B$$

$$= \begin{bmatrix} q_n B \\ q_n AB \\ \vdots \\ q_n A^{n-1} B \end{bmatrix}$$

Using now the fact that:

$$q_n B = q_n AB = \cdots = q_n A^{n-2} B = 0$$
$$q_n A^{n-1} B = 1$$

we get:

$$\bar{B} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

Remark 6.5.3 *We just present a procedure to determine the controllable canonical form with respect to the last row. This form is obtained using the last row of the inverse of the controllability matrix. Here we will give another procedure that is based on the observability matrix $\mathcal{O}$. For this purpose, let us assume that the dynamics of the system is described by: The system*

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

where $x(k) \in \mathbb{R}^n$, $y(k) \in \mathbb{R}$ and $u(k) \in \mathbb{R}$.

Now if we let P, ($\eta(k) = Px(k)$) be given by: The system

$$P = \mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-2} \\ CA^{n-1} \end{bmatrix}$$

and following the same idea as we did previously we can show that

$$\bar{A} = PAP^{-1} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ \times & \times & \times & \cdots & \times \end{bmatrix}$$

$$\bar{B} = PB = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

$$\bar{C} = CP^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

Using the transformation, the new one is given by:

$$\eta(k+1) = P[Ax(k) + Bu(k)] = PAP^{-1}Px(k) + PBu(k) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) = CP^{-1}Px(k) = CP^{-1}\eta(k)$$

with $\bar{A} = PAP^{-1}$, $\bar{B} = PB$ and $\bar{C} = CP^{-1}$.

Example 6.5.5 *To show how to get the controllable canonical description of system, let us consider the following dynamics:*

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

The system is controllable and its controllability matrix is given by:

$$\mathscr{C} = \begin{bmatrix} 1 & 2 & 2 & 6 \\ -1 & 2 & -2 & 6 \\ 0 & 1 & 1 & 5 \\ 1 & 0 & 4 & 0 \end{bmatrix}$$

and its inverse is:

$$\mathscr{C}^{-1} = \begin{bmatrix} 1 & -1 & 0 & 1 \\ 0.25 & 1 & -1.5 & 0.75 \\ -0.25 & 0.25 & 0 & 0.5 \\ 0 & -0.25 & 0.25 & -0.25 \end{bmatrix}$$

From this we get:

$$q_4 = \begin{bmatrix} 0 & -0.25 & 0.25 & -0.25 \end{bmatrix}$$

and the matrix P is given by:

$$P = \begin{bmatrix} q_4 \\ q_4 A \\ \vdots \\ q_4 A^3 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -0.25 & 0.5 & -0.25 \\ -0.25 & 0 & 0.5 & 0.25 \\ 0 & 0.25 & 0.5 & 0.25 \\ 0.25 & 0 & 0.5 & 0.75 \end{bmatrix}$$

Its inverse is:

$$P^{-1} = \begin{bmatrix} 1 & -3 & 1 & 1 \\ -1 & -1 & 3 & -1 \\ 1 & 0 & 1 & 0 \\ -1 & 1 & -1 & 1 \end{bmatrix}$$

From this we get the following controllable canonical form:

$$\eta(k+1) = A\eta(k) + Bu(k)$$
$$y(k) = C\eta(k)$$

with

$$\bar{A} = PAP^{-1}$$

$$= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -3 & 3 & 1 \end{bmatrix}$$

$$\bar{B} = PB$$

$$= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\bar{C} = CP^{-1}$$

$$= \begin{bmatrix} 0 & -2 & 0 & 2 \\ 0 & 2 & -4 & 2 \end{bmatrix}$$

Let us now generalize this idea to the multi-input multi-output case. Before this, we will introduce the concept of the controllability index. For this purpose, let us first of all assume that the matrix has full column rank, which means that all the columns in the matrix $B \in \mathbb{R}^{n \times m}$ are linearly independent. It is important to notice that the presence of a linearly dependent column to another one, means that the corresponding input is redundant and this column can be removed without affecting the results.

Firstly let us write B as follows:

$$B = \begin{bmatrix} b_1 & b_2 & \cdots & b_m \end{bmatrix}$$

where b_i is the ith column that is assumed to be linearly independent of all the other columns of B.

Notice also that the controllability matrix can be rewritten as follows using this:

$$\mathscr{C} = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}$$
$$= \begin{bmatrix} b_1 & \cdots b_m & Ab_1 & \cdots & Ab_m & \cdots & A^{n-1}b_1 & \cdots A^{n-1}b_m \end{bmatrix}$$

To search for the number of columns that are linearly independent starting from the left side, notice that when the column $A^\nu b_l$ depends on the left hand side columns, the columns $A^{\nu+1}b_l, \cdots A^{n-1}b_l$ will also depend on the left hand side columns. This means that once a column associated with b_l becomes linearly dependent, the rest of the columns associate with b_l are also linearly dependent.

Let us again rewrite again the controllability matrix as follows:

$$\mathscr{C} = \begin{bmatrix} b_1 & Ab_1 & \cdots & A^{n-1}b_1 & \cdots & b_m & Ab_m & \cdots & A^{n-1}b_m \end{bmatrix}$$

Let us denote by ρ_l the number of columns associated with the column b_l that are linearly independent in the controllability matrix $\mathscr{C}_\rho$ which means also that

$$\mathscr{C}_{\rho_l} = \begin{bmatrix} b_l & Ab_l & \cdots & A^{\rho_l-1}b_l \end{bmatrix}$$

are linearly independent and all the columns $A^{\rho_l+k}b_l$, $k = 0, 1, \cdots, n-1$ are linearly dependent.

This is true for any l, $l = 1, 2, \cdots, m$ and if the rank of $\mathscr{C}$ is equal to n, we have:

$$\rho_1 + \rho_2 + \cdots + \rho_m = n$$

where ρ_l is the controllability index associated with the column b_l. The controllability index of the controllability matrix (i.e: the pair (A, b)) is defined by

$$\rho = \max(\rho_1, \rho_2, \cdots, \rho_m)$$

It is important to notice that the controllability index , ρ, which is in some sense the largest integer such that the corresponding controllability matrix,

$$\begin{bmatrix} B & AB & \cdots & A^{\rho-1}B \end{bmatrix}$$

has a rank equal to n, satisfies the following:

$$\frac{n}{m} \leq \rho \leq \min(n_A, n - m + 1)$$

where n_A represents the degree of the minimal polynomial of A.

Theorem 6.5.7 *The system*

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned}$$

with $x(k) \in \mathbb{R}^n$, $y(k) \in \mathbb{R}^p$ and $u(k) \in \mathbb{R}^m$, is controllable (B is full column rank) if and only if the rank of the following matrix:

$$\mathscr{C}_r = \begin{bmatrix} B & AB & \cdots & A^{n-p}B \end{bmatrix}$$

is equal to n or the $n \times n$ matrix $\mathscr{C}_r^\top \mathscr{C}_r$ is nonsingular

Example 6.5.6 *To show how to compute the controllability indices, let us consider the following system:*

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned}$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Firstly, it is important to notice that the two columns are independent. In this case, we have $n = 4$, $m = 2$ and $p = 2$.

The controllability matrix of this system using the last theorem is given by:

$$\mathcal{C} = \begin{bmatrix} b_1 & b_2 & Ab_1 & Ab_2 & A^2b_1 & A^2b_2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 2 & 2 & 4 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 & 2 \\ 0 & 1 & 1 & 2 & 2 & 4 \end{bmatrix}$$

This matrix is has a rank equal to 4. From this we extract:

$$\mathcal{C}_o = \begin{bmatrix} b_1 & b_2 & Ab_1 & Ab_2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

This implies that the controllability indices are respectively $\rho_1 = 2$ and $\rho_2 = 2$, and the controllability index of the system is:

$$\rho = \max(2, 2) = 2$$

and satisfies:

$$\frac{n}{p} = \frac{4}{2} \leq \rho \leq \min(4, n - p + 1) = 3$$

Let us now focus on the transformation that gives the controllable canonical form. This transformation is based on the controllability indices associated with the columns, $b_1, \cdots, b_m$. Let us denote by $\mathcal{C}_o$ the number of columns that are linearly independent that we can extract from the controllability matrix starting from the left side. $\mathcal{C}_o$ is defined by:

$$\mathcal{C}_o = \begin{bmatrix} b_1 & \cdots & A^{\rho_1 - 1}b_1 & b_2 & \cdots & A^{\rho_2 - 1}b_2 & \cdots & b_m & \cdots & A^{\rho_m - 1}b_m \end{bmatrix}$$

such that $\sum_{l=1}^{m} \rho_l = n$. Define ϱ_i by:

$$\varrho_i = \sum_{l=1}^{i} \rho_l$$

Let us denote by q_i, $i = 1, \cdots, m$, the ϱ_ith row in $\mathscr{C}_o^{-1}$ that corresponds to the controllability index ρ_i, associated to b_i, $i = 1, \cdots, m$. The matrix of the transformation, P, is given by:

$$P = \begin{bmatrix} q_1 \\ q_1 A \\ \vdots \\ q_1 A^{\rho_1 - 1} \\ q_2 \\ q_2 A \\ \vdots \\ q_2 A^{\rho_2 - 1} \\ \vdots \\ q_m \\ q_m A \\ \vdots \\ q_m A^{\rho_m - 1} \end{bmatrix}$$

Using this matrix, we get the following controllability canonical form:

$$\begin{aligned} \eta(k+1) &= \bar{A} x(k) + \bar{B} u(k) \\ y(k) &= \bar{C} \eta(n) \end{aligned}$$

where $\bar{A} = P A P^{-1}$, $\bar{B} = P B$, and $\bar{C} = C P^{-1}$.

Using this transformation, the matrices $\bar{A}$ and $\bar{B}$ will have the following forms:

$$\bar{A} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} & \cdots & \bar{A}_{1\rho} \\ \bar{A}_{21} & \bar{A}_{22} & \cdots & \bar{A}_{2\rho} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{A}_{\rho 1} & \bar{A}_{\rho 2} & \cdots & \bar{A}_{\rho\rho} \end{bmatrix}$$

$$\bar{B} = \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \\ \vdots \\ \bar{B}_\rho \end{bmatrix}$$

where for any i and any $j \neq i$:

$$\bar{A}_{ii} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ \times & \times & \times & \cdots & \times \end{bmatrix}$$

$$\bar{A}_{ij} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ \times & \times & \times & \cdots & \times \end{bmatrix}$$

$$\bar{B}_1 = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 1 & \times & \times & \cdots & \times \end{bmatrix}$$

$$\bar{B}_2 = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ \times & 1 & \times & \cdots & \times \end{bmatrix}$$

$$\vdots$$

$$\bar{B}_\rho = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ \times & \times & \times & \cdots & 1 \end{bmatrix}$$

Example 6.5.7 *To show how to determine the controllable canonical form, let us consider the system of the previous example. It can be shown that $\mathscr{C}_o$ and $\mathscr{C}_o^{-1}$ are given by:*

$$\mathscr{C}_o = \begin{bmatrix} 1 & 1 & 1 & 2 \\ 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\mathscr{C}_o^{-1} = \begin{bmatrix} 1 & 0 & 0 & -1 \\ -1 & 1 & 0 & 1 \\ 3 & -3 & -2 & -2 \\ -1 & 1 & 1 & 1 \end{bmatrix}$$

From the matrix $\mathscr{C}_o^{-1}$, we get:

$$q_1 = \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix}$$
$$q_2 = \begin{bmatrix} 0 & 1 & 1 & 2 \end{bmatrix}$$

The matrix P of the transformation is given by:

$$P = \begin{bmatrix} q_1 \\ q_1 A \\ q_2 \\ q_2 A \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ -1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Its inverse is given by:

$$P^{-1} = \begin{bmatrix} 1 & 1 & -2 & 1 \\ 1 & 1 & -1 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 1 \end{bmatrix}$$

The controllable canonical form is given by:

$$\eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) = \bar{C}\eta(k)$$

with

$$\bar{A} = PAP^{-1}$$
$$= \left[\begin{array}{cc|cc} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 \\ 2 & 0 & -3 & 3 \end{array}\right]$$

$$\bar{B} = PB$$
$$= \left[\begin{array}{cc} 0 & 0 \\ 1 & 0 \\ \hline 0 & 0 \\ 0 & 1 \end{array}\right]$$

$$\bar{C} = CP^{-1}$$
$$= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 1 \end{bmatrix}$$

For the observable canonical form, the duality principe can be used. In fact, for a dynamical system with the following dynamics:

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

its dual system is given by:

$$\eta(k + 1) = A^{\top}\eta(k) + C^{\top}u(k)$$
$$y(k) = B^{\top}\eta(k)$$

The system

$$x(k + 1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

is observable if the rank of the observability matrix $\mathcal{O}$

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

is equal to n.

Since the transpose of the observability marix $\mathcal{O}$ will not change the rank, we get:

$$\mathcal{O}^{\top} = \begin{bmatrix} C^{\top} & A^{\top}C^{\top} & \cdots & (A^{\top})^{n-1}\, C^{\top} \end{bmatrix}$$

which represents the controllability matrix of the system:

$$\eta(k + 1) = A^{\top}\eta(k) + C^{\top}u(k)$$
$$y(k) = B^{\top}\eta(k)$$

Therefore, the observability canonical form can be obtained using the results on controllable canonical form. We will present two example to show how we obtain the observable canonical form for the single output and multiple outputs.

Remark 6.5.4 *We just present a procedure to determine the observable canonical form with respect to the first column. This form is obtained using the last row of the inverse of the observability matrix. Here we will give another procedure that is based on the controllability matrix $\mathcal{C}$. For this purpose, let us assume that the dynamics of the system is described by: The system*

$$x(k + 1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

where $x(k) \in \mathbb{R}^n$, $y(k) \in \mathbb{R}$ and $u(k) \in \mathbb{R}$.

Now if we let P, $(x(k) = P\eta(k))$ be given by: The system

$$P = \mathcal{C} = \begin{bmatrix} B & AB & \vdots & A^{n-2}B & A^{n-1}B \end{bmatrix}$$

and following the same idea as we did previously we can show that

$$\bar{A} = P^{-1}A = \begin{bmatrix} 0 & 0 & \cdots & 0 & \times \\ 1 & 0 & \cdots & 0 & \times \\ 0 & 1 & \cdots & 0 & \times \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \times \end{bmatrix}$$

$$\bar{B} = P^{-1}B = \begin{bmatrix} 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\bar{C} = CP = \begin{bmatrix} \times & \times & \times & \times & \times \end{bmatrix}$$

Using the transformation, the new one is given by:

$$\eta(k+1) = P^{-1}[Ax(k) + Bu(k)] = P^{-1}APP^{-1}x(k) + P^{-1}Bu(k) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) = CPP^{-1}x(k) = CP\eta(k)$$

with $\bar{A} = P^{-1}AP$, $\bar{B} = P^{-1}B$ and $\bar{C} = CP$.

Example 6.5.8 *To show how to compute the controllability canonical form for the single output case, let us consider the following system:*

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 & 1 & 1 \end{bmatrix}$$

The dual system of this one is given by:

$$v(k+1) = A^{\top}v(k) + C^{\top}u(k)$$
$$y(k) = B^{\top}v(k)$$

where

$$A^{\mathsf{T}} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$C^{\mathsf{T}} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$B^{\mathsf{T}} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

The controllability matrix of the dual system is given by:

$$\mathscr{C} = \begin{bmatrix} C^{\mathsf{T}} & A^{\mathsf{T}} C^{\mathsf{T}} & (A^{\mathsf{T}})^2 C^{\mathsf{T}} & (A^{\mathsf{T}})^3 C^{\mathsf{T}} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 3 & 7 \\ 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 \\ 1 & 2 & 4 & 7 \end{bmatrix}$$

This matrix is has a rank equal to 4. Its inverse is given by:

$$\mathscr{C}^{-1} = \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1.6 & -0.8 & 2.2 & -1.4 \\ -1.6 & -0.2 & -1.2 & 1.4 \\ 0.6 & 0.2 & 0.2 & -0.4 \end{bmatrix}$$

From this we get:

$$q = \begin{bmatrix} 0.6 & 0.2 & 0.2 & -0.4 \end{bmatrix}$$

The matrix P of the transformation is given by:

$$P = \begin{bmatrix} q; q(A^{\mathsf{T}}); q(A^{\mathsf{T}})^2; q(A^{\mathsf{T}})^3 \end{bmatrix}$$

$$= \begin{bmatrix} 0.6 & 0.2 & 0.2 & -0.4 \\ 0.2 & 0.4 & -0.6 & 0.2 \\ 0.4 & -0.2 & -0.2 & 0.4 \\ 0.8 & -0.4 & 0.6 & 0.8 \end{bmatrix}$$

Its inverse is given by:

$$P^{-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & -3 & 1 \\ 0 & 0 & -2 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix}$$

Based on this we get the following observable canonical form that we obtain by taking the dual:

$$\eta(k + 1) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) = \bar{C}\eta(k)$$

with

$$\bar{A} = \left(PA^\top P^{-1}\right)^\top$$

$$= \begin{bmatrix} 0.6 & 0.2 & 0.2 & -0.4 \\ 0.2 & 0.4 & -0.6 & 0.2 \\ 0.4 & -0.2 & -0.2 & 0.4 \\ 0.8 & -0.4 & 0.6 & 0.8 \end{bmatrix}$$

$$\bar{B} = \left(B^\top P^{-1}\right)^\top$$

$$= \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ -2 & 0 \\ 1 & 1 \end{bmatrix}$$

$$\bar{C} = \left(PC^\top\right)^\top$$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

Example 6.5.9 *To show how to compute the controllability canonical form for the multiple outputs case, let us consider the following system:*

$$x(k + 1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

It can be verified that this system is observable.
The dual system of this one is given by:

$$v(k + 1) = A^\top v(k) + C^\top u(k)$$
$$y(k) = B^\top v(k)$$

where

$$A^\top = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$C^\top = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$B^\top = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

The controllability matrix of the dual system using the last theorem is given by:

$$\mathscr{C} = \begin{bmatrix} C^\top & A^\top C^\top & (A^\top)^2 C^\top \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 1 & 1 & 3 & 3 \\ 1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & 0 & -1 \\ 1 & 0 & 2 & 2 & 4 & 3 \end{bmatrix}$$

From which we get:

$$\mathscr{C} = \begin{bmatrix} c_1^\top & A^\top c_1^\top & c_2^\top & A^\top c_2^\top \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 0 & -1 \\ 1 & 1 & 1 & 0 \\ 1 & 2 & 1 & 3 \end{bmatrix}$$

where $c_i^\top$ is the row i of the matrix $C^\top$.

The controllability indices of the dual system are respectively 2 and 2 and the controllability index of the dual system is also equal to 2.

This matrix is has a rank equal to 4. Its inverse is given by:

$$\mathscr{C}^{-1} = \begin{bmatrix} 1 & 2 & -1 & 0 \\ -3 & -3 & 2 & 1 \\ 2 & 1 & 0 & -1 \\ 1 & 1 & -1 & 0 \end{bmatrix}$$

From this we get:

$$q_1 = \begin{bmatrix} -3 & -3 & 2 & 1 \end{bmatrix}$$

$$q_2 = \begin{bmatrix} 1 & 1 & -1 & 0 \end{bmatrix}$$

The matrix P of the transformation is given by:

$$P = \begin{bmatrix} q_1 \\ q_1 A^\top \\ q_2 \\ q A^\top \end{bmatrix}$$

$$= \begin{bmatrix} -3 & -3 & 2 & 1 \\ -2 & -1 & 4 & -2 \\ 1 & 1 & -1 & 0 \\ 1 & 0 & -1 & 1 \end{bmatrix}$$

Its inverse is given by:

$$P^{-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & -3 & 1 \\ 0 & 0 & -2 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix}$$

Based on this we get the following observable canonical form that we obtain by taking the dual:

$$\eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) = \bar{C}\eta(k)$$

with

$$\bar{A} = \left(PA^\top P^{-1}\right)^\top$$

$$= \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & -2 & 0 & 2 \\ 0 & 4 & 0 & 4 \\ 0 & -9 & 1 & 5 \end{bmatrix}$$

$$\bar{B} = \left(B^\top P^{-1}\right)^\top$$

$$= \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 5 & 4 \\ 1 & 2 \end{bmatrix}$$

$$\bar{C} = \left(PC^\top\right)^\top$$

$$= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The controllable and observable canonical forms are of importance since they will make easier the controller and the observer design respectively. This matter will be covered in the next chapter.

It is important to notice that when dealing with the state space realization, i.e.: (A, B, C) and when the transfer matrix or the transfer function in case of SISO,

the dimension changes. The minimal realization is one that has the smallest-size A matrix for all triples (A, B, C) satisfying

$$H(s) = C\,(z\mathbb{I} - A)^{-1}\,B.$$

A realization (A, B, C) is minimal if and only if it is controllable and observable (see [1]).

6.6   Case Study

Let us consider the position control of a mechanical part driven by a dc motor. Previously the dynamics of the complete system was shown to be described by the following state space representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0 \\ y(t) = Cx(t) \end{cases} \tag{6.14}$$

where $x(t)$ and $u(t)$ are respectively the state vector and the control input and the matrices A, B and C are given by:

$$A \;=\; \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau_m} \end{bmatrix},$$

$$B \;=\; \begin{bmatrix} \frac{K_m}{\tau_m} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

where $K_m = 48.5$ and $\tau_m = 0.06\ s$.

The control input $u(t)$ is the voltage that we send to the dc motor and the state vector $x(t)$ is composed of:

- the speed of the mechanical part, $w(t)$

- the position of the part, $\theta(t)$

The modeling part of this system has been covered while the other concepts such stability, controllability and observability has not been covered and it will be done here. For the stability, we can either compute the transfer function of this system and then use any technique that we employ when the model is in transfer function. The other solution utilizes the Lyapunov. In fact if we consider a symmetric and positive-definite matrix $Q = \mathbb{I} \in \mathbb{R}^2$, the system will be stable if there a symmetric and positive-definite matrix, $P \in \mathbb{R}^2$ solution of the following Lyapunov equation:

$$A^\top P + PA = -Q$$

Let P be given by:

$$P = \begin{bmatrix} p_1 & p_2 \\ p_2 & p_3 \end{bmatrix}$$

Using this and the Lyapunov equation we can show that we can find a symmetric and positive-definite matrix P that satisfies the Lyapunov equation and therefore the

system is unstable. If we compute the transfer function, we can see directly that we have a pole at the origin which confirm the instability of the system.

For the controllability, this system will be controllable if the the rank of controllability matrix is equal to 2. In fact,

$$\mathscr{C} = \begin{bmatrix} B & AB \end{bmatrix}$$
$$= \begin{bmatrix} 0 & \frac{K_m}{\tau_m} \\ \frac{K_m}{\tau_m} & -\frac{K_m}{\tau_m^2} \end{bmatrix}$$

It is evident that the controllability matrix, $\mathscr{C}$ is of rank 2 which implies that our system is controllable.

For the observability, this system will be observable if the the rank of observability matrix is equal to 2. In fact,

$$\mathscr{O} = \begin{bmatrix} C \\ CA \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

It is evident that the observability matrix, $\mathscr{O}$ is of rank 2 which implies that our system is observable.

Since our system is unstable, the computation of the step response will give an unbounded output and therefore we will not compute it it here.

6.7   Conclusion

This chapter covers the state space representation. It is shown how to transform a continuous-time state space representation to an equivalent discrete-time one. The time concept and its computation is developed. The concepts of stability, controllability and observability are presented and techniques how to check these concepts are presented. Numerous examples are presented to show how each concept can be checked.

6.8   Problems

1. For the dynamical systems with the input $u(t)$ and the output $y(t)$ with the following dynamics develop the canonical forms:

- $\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} = u(t)$
- $\frac{d^2y(t)}{dt^2} + 4\frac{dy(t)}{dt} + 4y(t) = 4u(t)$
- $\frac{d^2y(t)}{dt^2} + 6\frac{dy(t)}{dt} + 8y(t) = 8u(t)$
- $\frac{d^3y(t)}{dt^3} + 3\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} = u(t)$

2. For the dynamical systems of the Problem 1,

    (a) determine the sampling period T
    (b) establish the corresponding discrete time description

3. Study the stability of the description of the Problems 1 & 2

4. For the dynamical systems of the Problems 1 & 2,

    (a) study the controllability of each description
    (b) study the observability of each description

5. For the dynamical systems of theProblems 1 & 2,

    (a) determine the solution of each description
    (b) determine the unit step response of each description for a chosen initial conditions that you impose.
    (c) plot the behavior of the states with respect to time t
    (d) using the final theorem show that the obtained results are correct

6. Consider a dynamical system with the following dynamics:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

    (a) develop the different canonical forms
    (b) establish their equivalent discrete time forms when the sampling period T is fixed to 0.1
    (c) study the stability, the controllability and the observability of each form
    (d) establish the solution when the inputs are fixed to unit steps
    (e) plot the phase diagram the behavior of the states
    (f) compute the transfer matrix of the system

7. Let the dynamics of a dynamical system be described by the following difference equations:

$$y(k + n) + a_{n-1}y(k + n - 1) + \cdots + a_1y(k + 1) + a_0y(k) = u(k)$$

where $y(k)$ and $u(k)$ represent respectively the output and the input of the system and $a_0, \cdots, a_{n-1}$ are know scalars.

(a) define

$$x_1(k) = y(k)$$
$$x_2(k) = y(k + 1)$$
$$\vdots$$
$$x_n(k) = y(k + n)$$

Based on this establish the corresponding state space description and determine which form we have.

(b) define

$$x_1(k) = y(k + n)$$
$$x_2(k) = y(k + n - 1)$$
$$\vdots$$
$$x_n(k) = y(k)$$

Based on this establish the corresponding state space description and determine which form we have

8. Consider a dynamical system with the following dynamics:

$$x(t + 1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -4 \end{bmatrix}$$

$$B = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 2 & 0 \end{bmatrix}$$

(a) develop the different canonical forms
(b) study the stability, the controllability and the observability of each form
(c) using Matlab, compute the step response

9. Consider a dynamical system with the following dynamics:

$$x(t + 1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ -2 & -2 & -4 & 0 \\ 1 & -1 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 2 & 0 \\ 0 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 2 & 0 & -1 \\ 0 & 1 & -1 & 1 \end{bmatrix}$$

(a) develop the different canonical forms
(b) study the stability, the controllability and the observability of each form
(c) using Matlab, compute the step response

7
Design Based on State Space

After reading this chapter the reader will:

1. be able to formulate a control design problem for mechatronic systems

2. be able to solve the design control problem

3. be able to compute the state feedback controller using either the pole placement technique or the optimal control approach

4. be able to use control tools to solve design control problems

7.1   Introduction

In the last chapters, we showed how to analysis and design linear time-invariant systems. The analysis has been done either using the state space or the transfer function descriptions. We showed also how to compute the performances of the given system. At the design phase we showed that using the transfer function description we were able to design some controllers like, the PID, the phase-lag, phase-lead or the phase lead-lag controllers. To compute the parameters of such controllers some procedures have been developed. We have also seen that the design can be done into

two steps. The first one uses the specifications of the system to choose the appropriate structure of the controller. Once this is fixed, the parameters of this controller are determined using the proposed procedures.

The aim of this chapter is to develop others techniques that can be used to design controllers based on the state space description. This approach requires more assumption that we don't have when using the transfer function approach. In this case, we need that the system is controllable and the state vector is accessible. The procedures that we developed for the transfer function description use mainly the system's output, meanwhile the ones of this chapter use the state vector.

Notice that if some of the states are not available for feedback, an estimator can be built to compute an estimate of the whole state vector or part of the states and therefore use this estimate for feedback instead of the state vector. This works fine and it is known in the literature as the separation principle. It consists of designing the controller and the estimator separately and when put together the results work fine.

7.2   Formulation of the Control Design Problem

Most of the built systems are either unstable or don't have the desired performances like for instance the settling time is not acceptable or the steady-state error is larger than a certain acceptable value. To overcome such situations and guarantee that the behavior of the closed-loop dynamics will be acceptable a controller should be added or improve the existing one.

In Chapter 5, we showed how to design classical controllers that are in general put in the direct loop and are mainly PID controllers or their equivalent phase-lag, phase-lead or phase lag-lead controllers. Meanwhile the one we will use here is put in the feedback loop and it is referred to as the state feedback controller. It requires extra assumptions that are summarized in:

- the complete access to the state vector,

- and the controllability (or stabilizable).

The condition of the accessibility to the state vector may be relaxed while the second one can not be. If we have only partial access to the state vector an estimator can be developed to estimate the state vector and therefore still continue to use the state feedback control. Another alternate consists of using the output feedback control.

To have an idea on the state feedback control, let us consider a dynamical linear discrete-time system with the following dynamics:

$$x(k+1) = Ax(k) + Bu(k), \, x(0) = x_0 \tag{7.1}$$

and suppose that we want that the closed-loop with a state feedback controller to have a certain desired behavior.

The structure of the state feedback controller is given by:

$$u(k) = -Kx(k) \tag{7.2}$$

where K is the gain matrix that we need to compute.

The problem we will face in this chapter is how we can design the matrix gain K in the single input single output and multi inputs multi outputs cases in order to guarantee the desired performances.

In the rest of this chapter we will focus on two approaches. The first one is known in the literature as pole assignment technique and the second one is the linear quadratic regulator. This approach uses a cost function to choose the optimal state feedback control.

7.3 State Feedback Controller Design

One of the methods that we can use to design the appropriate controller that will guarantee the desired performances is the pole assignment technique. The idea of this approach consists of designing the state feedback controller either for single input single output or multi inputs multi outputs systems that makes the closed-loop dynamics behave like the desired one with the acceptable performances. This technique requires the complete accessibility to the state vector or the system is observable that we can use an estimator to estimate the state vector otherwise the control law can not be computed. The approach can be stated as follows: Given a dynamical system with the following dynamics

$$x(k + 1) = Ax(k) + Bu(k), x(0) = x_0$$

find a controller of the form:

$$u(k) = -Kx(k)$$

such that the closed-loop dynamics will give the desired performances.

The gain K in the controller expression needs to be determined. Its dimension will depend on the nature of system we are dealing with, either single input single output or multi inputs multi outputs.

The main idea behind this technique consists of transforming the desired performances to a desired characteristic polynomial that will provide the desired eigenvalues it is why we refer to it as pole assignment technique.

For single input single output case, if the system is of dimension n, then the gain K has n scalar gains to be determined, i.e.:

$$K = \begin{bmatrix} k_1 & \cdots & k_n \end{bmatrix} \tag{7.3}$$

Remark 7.3.1 *The pole placement technique consists first of all of obtaining the poles of the closed-loop dynamics that give the desired performances, then using these poles the controller gain K is computed.*
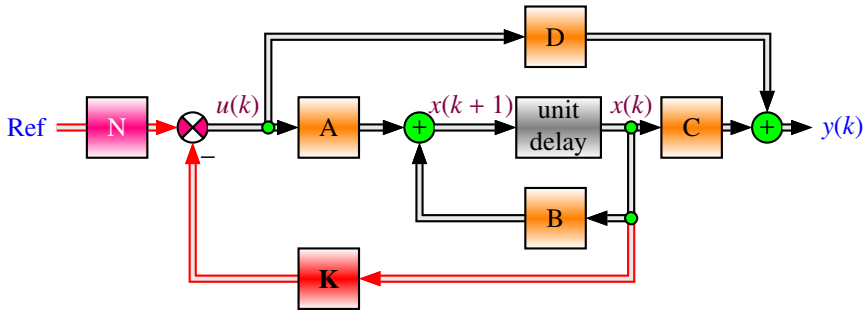
Fig. 7.1 Block diagram of discrete-time linear system

Let us assume that the desired poles that give the performances are:

$$z_1, z_2, \cdots, z_n$$

and that we can get from the desired specifications.

The corresponding desired characteristic polynomial is given by:

$$\Delta_d(z) = (z - z_1)(z - z_2) \cdots (z - z_n) \tag{7.4}$$

Based on Chapter 6, the closed-loop characteristic polynomial is given by:

$$\Delta(z) = |z\mathbb{I} - A + BK| \tag{7.5}$$

The design approach of pole assignment consists of equating the two characteristic polynomials. Performing this we get:

$$|z\mathbb{I} - A + BK| = (z - z_1)(z - z_2) \cdots (z - z_n) \tag{7.6}$$

which represents an algebraic equation with n unknown variables, $k_i, i = 1, 2, \cdots, n$ that have to be determined. The solution of this equation will give the appropriate gains.

Remark 7.3.2 *More often the specifications of the system are given in continuous-time and can combine the stability with the overshoot, the settling time, the steady error, etc. To get the desired poles in this case, the transformation is made in the continuous-time to get the desired poles in the s-domain and with the transformation, $z = e^{sT}$, where T is the sampling period of the system, we can compute the corresponding poles in the z-domain that should be inside the unit circle.*

Example 7.3.1 *In order to show how the pole placement method works, let us assume that we have a dynamical system with two states, $x_1(k)$ and $x_2(k)$ and suppose that the dynamics of the system has been transformed to the following discrete-time form:*

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} u(k)$$

This system has its poles outside the unit circle and therefore, it is unstable.

Firstly, we need to check the controllability of the system. This can be done by computing

$$\mathscr{C} = \begin{bmatrix} B & AB \end{bmatrix}$$
$$= \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & -0.1 \end{bmatrix}$$

which is of rank 2 and this means that the system is completely controllable and therefore a state feedback controller exists.

Let us also assume that the poles that give the desired performances are by:

$$z_1 = 0.2 + j0.2$$
$$z_2 = 0.2 - j0.2$$

The corresponding desired characteristic equation is given by:

$$\Delta_d(z) = (z - 0.2 - j0.2)(z - 0.2 + j0.2) = z^2 - 0.4z + 0.08$$

Let the controller gain, K, i.e.:

$$K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}$$

The characteristic equation of the closed-loop dynamics is given by:

$$\begin{aligned}
\Delta(z) &= |z\mathbb{I} - A + BK| \\
&= \left\| \begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ -2 & 1 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} \right\| \\
&= \left\| \begin{bmatrix} z + 0.1k_1 & -1 + 0.1k_2 \\ 2 + 0.1k_1 & z - 1 + 0.1k_2 \end{bmatrix} \right\| \\
&= z^2 + (-1 + 0.1k_1 + 0.1k_2)z + 2 - 0.2k_2
\end{aligned}$$

Equating the two characteristic equations gives:

$$-1 + 0.1k_1 + 0.1k_2 = -0.4$$
$$2 - 0.2k_2 = 0.08$$

Solving these equations gives:

$$k_1 = -3.6$$
$$k_2 = 9.6$$

We can easily compute the eigenvalues of the closed-loop dynamics and find out that they are equals to the desired poles.

*Matlab can be used to compute controller gain using the function **place**. The following instructions are used for this purpose:*

```
% Data
A = [0 1; -2 1]
B = [0.1;  0.1]
C = [1 0]
```

```
% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[0.2+0.2*i, 0.2-0.2*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)
```

If the desired poles are located at $0.1 \pm 0.1 j$ the controller gain is given by:

$$k_1 = -1.9$$
$$k_2 = 9.9$$

To understand the relationship between the pole location and the system response let us consider the following cases obtained from the poles $0.1 \pm 0.1 j$ by acting on the real and/or imaginary parts :

- *desired poles located at $0.4 \pm 0.4 j$*

- *desired poles located at $0.025 \pm 0.025 j$*

- *desired poles located at $0.4 \pm 0.1 j$*

- *desired poles located at $0.025 \pm 0.1 j$*

- *desired poles located at $0.1 \pm 0.4 j$*

- *desired poles located at $0.1 \pm 0.0.25 j$*

The corresponding gains are given by:

- *poles at $0.4 \pm 0.4 j$*

$$k_1 = -6.4$$
$$k_2 = 8.4$$

- *poles at $0.025 \pm 0.025 j$*

$$k_1 = -0.4937$$
$$k_2 = 9.937$$

- *poles at $0.4 \pm 0.1 j$*

$$k_1 = -7.15$$
$$k_2 = 9.15$$

- *poles at $0.025 \pm 0.1 j$*

$$k_1 = -0.4469$$
$$k_2 = 9.9469$$

- *poles at* $0.1 \pm 0.4j$

$$k_1 = -1.15$$
$$k_2 = 9.15$$

- *poles at* $0.1 \pm 0.025j$

$$k_1 = -1.9469$$
$$k_2 = 9.9469$$

Using Matlab, the following program has been written to simulate the time response for a step input of amplitude equal to one:

```
% Data
A = [0 1; -2 1]
B = [0.1;  0.1]
C = [1 0]
d=0

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% chose the time span
t=0:0.001:0.1;

% fix the input as a step
u=ones(size(t));

% Controller gain computation when the poles are located at
[0.1+ 0.1j  0.1-0.1j] K=place(A,B,[0.1+0.1*i, 0.1-0.1*i])

% Compute the step response of the closed-loop with this gain
[y0,x0]=dlsim(A-B*K,B,C,d,u)

% Controller gain computation when the poles are located at
[0.4+ 0.4j  0.4-0.4j] K=place(A,B,[0.4+0.4*i, 0.4-0.4*i])

% Compute the step response of the closed-loop with this gain
[y1,x1]=dlsim(A-B*K,B,C,d,u)

% Controller gain computation when the poles are located at
[0.025+ 0.025j  0.025-0.025j]
K=place(A,B,[0.025+0.025*i, 0.025-0.025*i])


% Compute the step response of the closed-loop with this gain
[y2,x2]=dlsim(A-B*K,B,C,d,u)

% Controller gain computation when the poles are located at
```

```
 [0.4+ 0.1j  0.4-0.1j] K=place(A,B,[0.4+0.1*i, 0.4-0.1*i])

% Compute the step response of the closed-loop with this gain
[y3,x3]=dlsim(A-B*K,B,C,d,u)

% Controller gain computation when the poles are located at
 [0.025+ 0.1j  0.025-0.1j] K=place(A,B,[0.025+0.1*i, 0.025-0.1*i])

% Compute the step response of the closed-loop with this gain
[y4,x4]=dlsim(A-B*K,B,C,d,u)

% Controller gain computation when the poles are located at
[0.1+ 0.4j  0.1-0.4j] K=place(A,B,[0.1+0.4*i, 0.1-0.4*i])


% Compute the step response of the closed-loop with this gain
[y5,x5]=dlsim(A-B*K,B,C,d,u)


% Controller gain computation when the poles are located at
[0.1+ 0.025j  0.1-0.025j] K=place(A,B,[0.1+0.025*i, 0.1-0.025*i])

% Compute the step response of the closed-loop with this gain
[y6,x6]=dlsim(A-B*K,B,C,d,u)

stairs(t,x0(:,1),'b')
xlabel('time in sec')
ylabel('x(1) and x(2)')
title('Behavior of the states versus time')

hold on
stairs(t,x0(:,2),'b')

stairs(t,x1(:,1),'r')
stairs(t,x1(:,2),'r')

stairs(t,x2(:,1),'m')
stairs(t,x2(:,2),'m')

stairs(t,x3(:,1),'c')
stairs(t,x3(:,2),'c')

stairs(t,x4(:,1),'g')
stairs(t,x4(:,2),'g')

stairs(t,x5(:,1))
stairs(t,x5(:,2))

stairs(t,x6(:,1))
```

```
stairs(t,x6(:,2))

legend('1','2','3','4','5','6','7')

print -deps chap5.fig.1
```

The behavior of the states versus time is illustrated by Fig. (7.2).
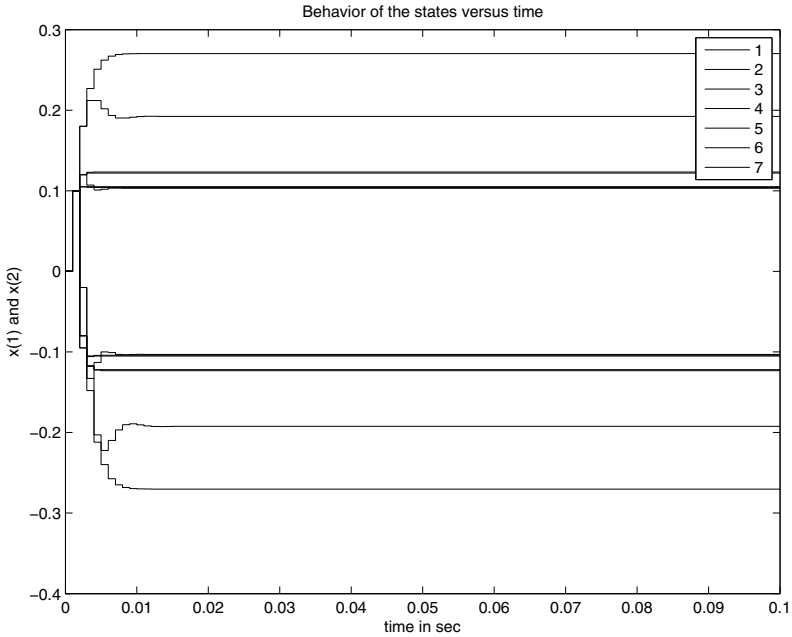
Fig. 7.2 Behavior of the output versus time with state feedback controller

When the state space description is put in the controllable form, the computation of the controller's gains becomes easier. In fact, referring to the previous chapter the controllable form for the open dynamics is given by:

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} b_{n-1} & \cdots & b_1 \end{bmatrix}.$$

Notice also that

$$-BK = -\begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} k_1 & \cdots & k_n \end{bmatrix} = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ -k_1 & \cdots & -k_n \end{bmatrix}$$

The corresponding characteristic polynomial of the closed-loop dynamics is:

$$|z\mathbb{I} - A + BK| = z^n + (a_{n-1} + k_n) z^{n-1} + \cdots + (a_1 + k_2) z + (a_0 + k_1) z^0 \quad (7.7)$$

The desired characteristic polynomial can also be put in the following form:

$$\Delta_d(z) = z^n + d_{n-1} z^{n-1} + \cdots + d_1 z + d_0 z^0 \quad (7.8)$$

By equating the characteristic polynomial with the desired characteristic polynomial we get:

$$k_{i+1} = d_i - a_i, i = 0, 1, 2, \cdots, n-1$$

Remark 7.3.3 *In the previous chapter, we presented a transformation that put the system description in the controllable canonical form, a question that comes is what relationship exists between the controller gains of the original description, K and the one of the controllable canonical form, $\bar{K}$. To answer this this question, notice that the characteristic equation for the closed-loop of the system in the controllable canonical form is given by:*

$$\det\left(z\mathbb{I} - \bar{A} + \bar{B}\bar{K}\right) = 0$$

where $\bar{A}$ and $\bar{B}$ are the matrices of the controllable canonical from obtained after the transformation $\eta(k) = Px(k)$.

Using the fact that the matrix P is nonsingular and $PP^{-1} = \mathbb{I}$, we get

$$\det\left(zPP^{-1} - PAP^{-1} + PB\bar{K}PP^{-1}\right) = 0$$

that we can write as follows:

$$\det\left(P\left(z\mathbb{I} - A + B\bar{K}P\right)P^{-1}\right) = 0$$

This gives in turn:

$$\det(P) \det\left(z\mathbb{I} - A + B\bar{K}P\right) \det\left(P^{-1}\right) = 0$$

Since $\det(P) \neq 0$, $\det\left(P^{-1}\right) \neq 0$ *and* $\det(P) \det\left(P^{-1}\right) = 1$, *we obtain:*

$$\det\left(z\mathbb{I} - A + B\bar{K}P\right) = 0$$

This characteristic equation will have the same poles as the characteristic equation of original description:

$$\det\left(z\mathbb{I} - A + BK\right) = 0$$

if we have the following relation between the controller gains:

$$K = \bar{K}P$$

Example 7.3.2 *In order to show how the pole placement method works when the system dynamics is in canonical controllable form, let us assume that we have a dynamical system with two states, $x_1(k)$ and $x_2(k)$ and suppose that the dynamics of the system has been transformed to the following discrete-time form:*

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$

Firstly, we need to check the controllability of the system. This can be done by computing

$$\begin{aligned} \mathscr{C} &= \begin{bmatrix} B & AB \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \end{aligned}$$

which is of rank 2 and this means that the system is completely controllable and therefore a state feedback controller exists.

It is important to notice that system in open-loop is unstable since its poles are outside the unit circle. Let us also assume that the poles that give the desired performances are given by:

$$z_1 = 0.2 + j0.2$$
$$z_2 = 0.2 - j0.2$$

The corresponding desired characteristic equation is given by:

$$\Delta_d(z) = (z - 0.2 - j0.2)(z - 0.2 + j0.2) = z^2 - 0.4z + 0.08$$

Let the controller gain, K, i.e.:

$$K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}$$

Using the relationship $k_{i+1} = d_i - a_1$, when $i = 0, 1$, we get:

$$k_1 = -1.92$$
$$k_2 = 0.6$$

We can easily compute the eigenvalues of the closed-loop dynamics and find out that they are equals to the desired poles.

More often the computation for general state space description using the pole assignment technique is tedious and one of the used method to overcome this problem is to use the Ackerman's method. This method is based on the following relations:

$$\Delta_d(z) = z^n + d_{n-1}z^{n-1} + \cdots + d_1 z + d_0 z^0$$

where $\Delta(z)$ is the desired characteristic polynomial.

For the closed-loop dynamics we have also:

$$\Delta_d(A - BK) = (A - BK)^n + d_{n-1}(A - BK)^{n-1} + \cdots + d_1(A - BK) + d_0\mathbb{I} = 0 \quad (7.9)$$

To use this relation we need firstly to expand the terms $(A - BK)^n$, $(A - BK)^{n-1}$, ..., $(A - BK)$. For this purpose, notice that:

$$\mathbb{I} = \mathbb{I}$$
$$(A - BK) = A - BK$$
$$(A - BK)^2 = A^2 - ABK - BK(A - BK)$$
$$(A - BK)^3 = A^3 - A^2 BK - ABK(A - BK) - BK(A - BK)^2$$
$$(A - BK)^4 = A^4 - A^3 BK - A^2 BK(A - BK) - ABK(A - BK)^2 - BK(A - BK)^3$$
$$\vdots$$
$$(A - BK)^{n-1} = A^{n-1} - A^{n-2}BK - \ldots - BK(A - BK)^{n-2}$$
$$(A - BK)^n = A^n - A^{n-1}BK - \ldots - BK(A - BK)^{n-1}$$

In order to use (7.9) multiply these relations respectively by d_0, d_1, ..., d_{n-1} and 1 and sum them, we get:

$$\Delta_d(A - BK) = \Delta_d(A) - \mathscr{C} \begin{bmatrix} d_1 K + d_2 K(A - BK) + \ldots + K(A - BK)^{n-1} \\ d_2 K + d_3 K(A - BK) + \ldots + K(A - BK)^{n-2} \\ \vdots \\ K \end{bmatrix}$$

where $\mathscr{C}$ is the controllability matrix and $\Delta_d(A)$ is given by:

$$\Delta_d(A) = A^n + d_{n-1}A^{n-1} + \cdots + d_1 A + d_0\mathbb{I}.$$

Now if the system is controllable, which means that the inverse of the controllability exists, and the fact that $\Delta_d(A - BK) = 0$, this relation can be rewritten as follows:

$$\begin{bmatrix} d_1 K + d_2 K(A - BK) + \ldots + K(A - BK)^{n-1} \\ d_2 K + d_3 K(A - BK) + \ldots + K(A - BK)^{n-2} \\ \vdots \\ K \end{bmatrix} = \mathscr{C}^{-1}\Delta_d(A)$$

To extract the controller gain, K, from this relation we multiply both sides by $\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$, which gives in turn:

$$K = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \mathscr{C}^{-1} \Delta_d(A)$$

In summary the Ackerman's method consists of the following steps:

1. compute the desired characteristic polynomial $\Delta_d(z)$ as before, i.e.:

$$\Delta_d(z) = z^n + d_{n-1} z^{n-1} + \cdots + d_1 z + d_0 z^0$$

2. use the Cayly-Hamilton theorem to compute $\Delta_d(A)$, i.e.:

$$\Delta_d(A) = A^n + d_{n-1} A^{n-1} + \cdots + d_1 A + d_0 \mathbb{I}$$

3. use the following formula to compute the gain K:

$$K = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \mathscr{C}^{-1} \Delta_d(A)$$

where $\mathscr{C}$ is the controllability matrix.

Remark 7.3.4 *Notice that the presence of the inverse of the controllability matrix $\mathscr{C}$ in the computation of the controller gain, justifies why we need the controllability assumption we made earlier for our system.*

Remark 7.3.5 *It is important to notice that the presence of the inverse of the controllability matrix, $\mathscr{C}$, may render the computation of the controller gain harder. To avoid this, the following can be used:*

- *Compute $v^\top$ such that:*

$$v^\top \mathscr{C} = \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}$$
$$v^\top = (v_1, \cdots, v_n)$$

- *Compute the controller gain K using the following:*

$$K = v^\top \Delta_d(A)$$

Example 7.3.3 *In this example, we will show that the design of the state feedback controller is affected by the canonical forms i.e. the controller gains are different. For this purpose, let us consider a dynamical system with output $y(k)$ and input $u(k)$ has the following dynamics:*

$$G(z) = \frac{z + 1}{z^2 - 1.4z + 0.48}$$

Let us firstly establish the canonical forms:

- *controllable canonical form: following the steps for establishing the controllable form as in Chapter 6, we have:*

$$x(k+1) = \begin{bmatrix} 0 & 1 \\ -0.48 & 1.4 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$
$$y(k) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(k)$$

- *observable canonical form: following the same steps for establishing the observable form as in Chapter 6, we get:*

$$x(k+1) = \begin{bmatrix} 1.4 & 1 \\ -0.48 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k)$$

- *Jordan canonical form: following the same steps for establishing the observable form as in Chapter 6, we get:*

$$x(k+1) = \begin{bmatrix} 0.6 & 0 \\ 0 & 0.8 \end{bmatrix} x(k) + \begin{bmatrix} -8 \\ 9 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(k)$$

It can be shown easily that the system is controllable. Using the function **place** (or **acker**) of Matlab when the desired poles are located at $0.2 \pm 0.2j$, we get the controller gains for each canonical form as follows:

- *controllable canonical form:*

$$K = \begin{bmatrix} -0.4 & 1 \end{bmatrix}$$

- *observable canonical form:*

$$K = \begin{bmatrix} 0.5139 & 0.4861 \end{bmatrix}$$

- *Jordan canonical form:*

$$K = \begin{bmatrix} 0.1250 & 0.2222 \end{bmatrix}$$

Simulation results for each canonical form with the corresponding controller gain are illustrated by Fig. 7.3 when the input is fixed to a step function. The Matlab progran that gives us such simulation results is gven:

```
% Data for controllable form
A = [0 1; -0.48 1.4]
B = [0;  1]
C = [1 1]
d=0

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[0.2+0.2*i, 0.2-0.2*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)

t=0:0.01:0.2
u=ones(size(t))
```

```
[y,x]=dlsim(A-B*K,B,C,d,u)
stairs(t,x(:,1),'r')
hold on
stairs(t,x(:,2),'r')
xlabel('Time in seconds')
ylabel('States x1(k) and x2(k)')
title('states versus time for a step input')


% Data for observable form
A = [1.4 1; -0.48 0]
B = [1;  1]
C = [1 0]
d=0

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[0.2+0.2*i, 0.2-0.2*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)

t=0:0.01:0.2
u=ones(size(t))
[y,x]=dlsim(A-B*K,B,C,d,u)
stairs(t,x(:,1),'b')
stairs(t,x(:,2),'b')
xlabel('Time in seconds')
ylabel('Output y(k)')
title('States versus time for a step input')


% Data for Jordan form
A = [0.6 0; 0 0.8]
B = [-8;  9]
C = [1 1]
d=0

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[0.2+0.2*i, 0.2-0.2*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)
```

```
t=0:0.01:0.2
u=ones(size(t))
[y,x]=dlsim(A-B*K,B,C,d,u)
stairs(t,x(:,1),'g')
stairs(t,x(:,2),'g')
xlabel('Time in seconds')
ylabel('States x1(k) and x2(k)')
title('States versus time for a step input')
legend('1','2','3','4','5','6')
```
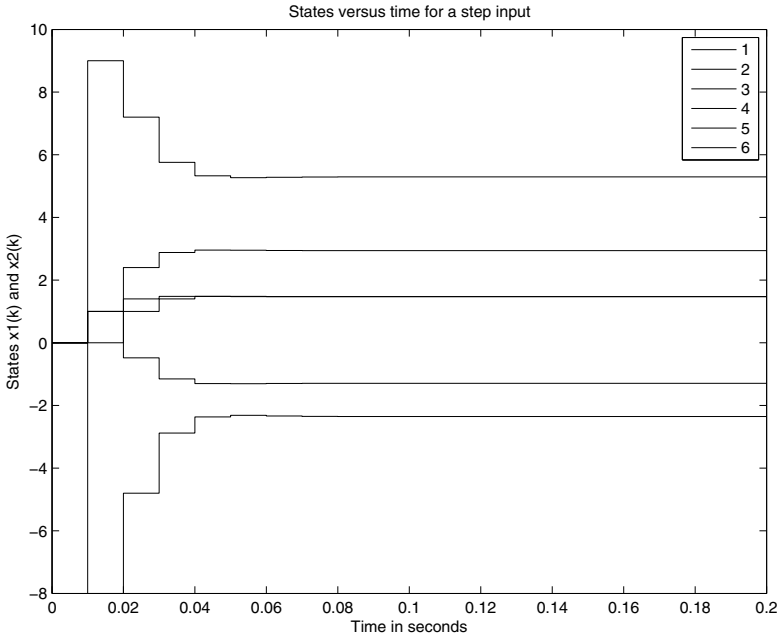

Fig. 7.3 Behavior of states vs time with state feedback controller

Example 7.3.4 *To show how the state feedback controller design procedure works for a practical system (a dc motor driving a mechanical load), let us consider the single-input single-output system with the following dynamics:*

$$G(s) \;=\; \frac{k}{s(\tau s + 1)}$$

with $k = 1$ and $\tau = 50ms$ represent respectively the gain and the time constant of the system.

This system represents one wheel of the balancing robot that we have already presented. The system is unstable and has two poles located respectively at

0 and $-\frac{1}{\tau}$. Our aim in this example is to stabilize the closed-loop dynamics and improve the settling time at 5% equal to 50ms while guaranteeing that the overshoot is less than or equal to 5%.

Firstly, we need to choose the sampling period T. Since the time constant is equal to 50 ms, a proper choice for T is 5 ms. This value will be used to get the different canonical forms.

To solve this design problem, we will use all the canonical forms. Therefore we have (see Chap. 6 for more details):

- the controllable form of this system is given by:

$$x(k + 1) = Fx(k) + Gu(k), x(0) = x_0$$
$$y(k) = Cx(k)$$

  with

$$F = \phi(T) = \begin{bmatrix} 1 & \tau\left[1 - e^{-\frac{T}{\tau}}\right] \\ 0 & e^{-\frac{T}{\tau}} \end{bmatrix} = \begin{bmatrix} 1 & 0.0048 \\ 0 & 0.9048 \end{bmatrix}$$

$$G = \Psi(T) = \begin{bmatrix} k\left[T - \tau\left[1 - e^{-\frac{T}{\tau}}\right]\right] \\ k\left[1 - e^{-\frac{T}{\tau}}\right] \end{bmatrix} = \begin{bmatrix} 0.0002 \\ 0.0952 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

- the observable form of this system is given by:

$$x(k + 1) = Fx(k) + Gu(k), x(0) = x_0$$
$$y(k) = Cx(k)$$

  with

$$F = \phi(T) = \begin{bmatrix} e^{-\frac{T}{\tau}} & \tau\left[1 - e^{--\frac{T}{\tau}}\right] \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.9048 & 0.0048 \\ 0 & 1 \end{bmatrix}$$

$$G = \Psi(T) = \begin{bmatrix} k\left[T - \tau\left[1 - e^{-\frac{T}{\tau}}\right]\right] \\ \frac{k}{\tau}T \end{bmatrix} = \begin{bmatrix} 0.0002 \\ 0.1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

- the Jordan form of this system is given by:

$$x(k + 1) = Fx(k) + Gu(k), x(0) = x_0$$
$$y(k) = Cx(k)$$

  with

$$F = \phi(T) = \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{T}{\tau}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0.9048 \end{bmatrix}$$

$$G = \Psi(T) = \begin{bmatrix} kT \\ k\tau\left[1 - e^{-\frac{T}{\tau}}\right] \end{bmatrix} = \begin{bmatrix} -0.005 & 0 \\ 0 & -0.0048 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & -1 \end{bmatrix}$$

From the specifications, we get:

$$d = 100e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}}$$

$$t_s = \frac{3}{\xi\omega_n} = 0.05$$

From these relations we obtain:

$$\xi = 0.707$$

$$\omega_n = 84.8656 rd/s$$

which gives the following poles:

$$s_{1,2} = \xi\omega_n \pm j\omega_n \sqrt{1-\xi^2} = -60.0000 \pm 60.0181j$$

Their corresponding poles in discrete-time domain when the sampling period T is chosen equal to 0.1 are given by:

$$z_{1,2} = e^{s_{1,2}T} = 0.7077 \pm 0.2190j$$

The corresponding characteristic polynomial is given by

$$\Delta_d(z) = (z - z_1)(z - z_2) = z^2 - 1.4154z + 0.5488$$

The closed-loop characteristic polynomial is given by:

$$\Delta(z) = |z\mathbb{I} - F + GK|$$

By equating the two characteristic polynomials we get the controller gain depending on the considered representation as follows:

- *controllable form*

$$K = \begin{bmatrix} 280.3412 & 4.4304 \end{bmatrix}$$

- *observable form*

$$K = \begin{bmatrix} 191.7330 & 4.4304 \end{bmatrix}$$

- *Jordan form*

$$K = \begin{bmatrix} 280.3412 & 191.7330 \end{bmatrix}$$

For the state feedback design for multi-input multi-output dynamical systems, the control law is still given by the following expression:

$$u(t) = -Kx(t)$$

where $u(t) \in \mathbb{R}^m$ and $K \in \mathbb{R}^{n \times m}$.

When the system in open loop is asymptotically stable an optimal state feedback can be designed using the Lyapunov approach. In fact, if our system is asymptotically stable in open-loop, this implies that there exists a symmetric and positive-definite matrix P such that the following holds:

$$A^{\top}PA - P = -Q$$

for a given symmetric and positive-definite matrix Q.

Now if we consider the following Lyapunov function candidate:

$$V(x_k) = x_k^\top P x_k$$

Based on Chap. 6, the discrete-time rate change of $V(x_k)$ is given by:

$$\Delta V = V(x_{k+1}) - V(x_k)$$

For the closed-loop dynamics, we have:

$$
\begin{aligned}
\Delta V &= V(x_{k+1}) - V(x_k) \\
&= [Ax_k + Bu_k]^\top P [Ax_k + Bu_k] - x_k^\top P x_k \\
&= x_k^\top A^\top P A x_k + 2u_k^\top B^\top P A x_k + u_k^\top B^\top P B u_k - x_k^\top P x_k
\end{aligned}
$$

To design the optimal state feedback controller, we will consider the following performance index:

$$J = \Delta V$$

Using now the optimality condition (see [3], we get:

$$\frac{\partial J}{\partial u_k} = \frac{\partial \Delta V}{\partial u_k} = 0$$

which implies:

$$2B^\top P A x_k + 2B^\top P B u_k = 0$$

that gives in turn:

$$u_k = -\left[B^\top P B\right]^{-1} B^\top P A x_k$$

The controller gain is then given by:

$$K = \left[B^\top P B\right]^{-1} B^\top P A$$

where P is the solution of the following Lyapunov function:

$$A^\top P A - P = -Q$$

for a given Q.

Example 7.3.5 *To show how this technique applies for asymptotically stable system, let us consider the following one with the transfer function given by:*

$$G(s) = \frac{2}{(s+1)(s+2)}$$

This system is asymptotically stable since its poles are respectively -1 and -2.

Let us firstly establish its corresponding discrete-time state space representation. This can be done by the following procedure developed earlier.

In fact the continuous-time state space description of this system is given:

$$
\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t), x(0) = x_0 \\
y(t) &= Cx(t)
\end{aligned}
$$

with

$$A = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix},$$

$$B = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 2 & -2 \end{bmatrix}$$

Let the sampling period T be equal to 0.05. The transition matrix $\phi(s)$ is given by:

$$\phi(s) = [s\mathbb{I} - A]^{-1}$$

$$= \begin{bmatrix} \frac{1}{s+1} & 0 \\ 0 & \frac{1}{s+2} \end{bmatrix}$$

which gives:

$$\phi(t) = \begin{bmatrix} e^{-t} & 0 \\ 0 & e^{-2t} \end{bmatrix}$$

From this expression we get:

$$\phi(T) = \begin{bmatrix} e^{-T} & 0 \\ 0 & e^{-2T} \end{bmatrix} = \begin{bmatrix} 0.9512 & 0 \\ 0 & 0.9048 \end{bmatrix}$$

$$\Psi(T) = \int_0^T \phi(T - \tau)Bd\tau$$

$$= \int_0^T \begin{bmatrix} e^{-(T-\tau)} & 0 \\ 0 & e^{-2(T-\tau)} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} d\tau$$

$$= \begin{bmatrix} 1 - e^{-T} \\ 0.5(1 - e^{-2T}) \end{bmatrix} = \begin{bmatrix} 0.0488 \\ 0.0476 \end{bmatrix}$$

Finally we get the discrete-time description:

$$x_{k+1} = \phi(T)x_k + \Psi(T)u_k$$

$$y_k = Cx_k$$

We can check easily that the system is controllable and observable and therefore there is a state feedback control of the form $u_k = -[B^{\mathsf{T}}PB]^{-1} B^{\mathsf{T}}PAx_k$ where the matrix P is the solution of the following Lyapunov equation:

$$A^{\mathsf{T}}PA - P = -Q$$

for a given symmetric and positive-definite matrix Q.

Using $Q = \mathbb{I}$, we get

$$P = \begin{bmatrix} 10.5022 & 0 \\ 0 & 5.5146 \end{bmatrix}$$

The corresponding controller gain is given by:

$$K = \begin{bmatrix} 12.9981 & 6.3326 \end{bmatrix}$$

Remark 7.3.6 *It is important to notice that the observability is not necessarily invariant under a state feedback control. We can show this by the following example:*

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -0.6 & -0.7 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

It can be firstly checked that the system is controllable and observable. Since the system is controllable, the poles of the system can be placed any where inside the unit circle.

The controllability matrix is given by:

$$\mathscr{C} = \begin{bmatrix} 0 & 1 \\ 1 & -0.7 \end{bmatrix}$$

$$\mathscr{O} = \begin{bmatrix} 0 & 1 \\ -0.6 & -0.7 \end{bmatrix}$$

If we select the controller gain K given by:

$$K = \begin{bmatrix} -0.6 & -1 \end{bmatrix}$$

The closed-loop poles are placed respectively at 0 *and* 0.3. *For this gain the observability matrix is given by:*

$$\mathscr{O} = \begin{bmatrix} 0 & 1 \\ 0 & 0.3 \end{bmatrix}$$

which is of rank one and therefore the observability is lost by this state feedback control law.

For the multi-input multi-output, we will cover two approaches that we can use to design the state feedback controller. The first approach is simple is consist of writing the matrix gain in way that the design problem can be solved using the single input single output approach. In fact if the system has m inputs and n states, the gain K has $m \times n$ scalar gains to be determined. Using the single input single output approach this can not be done unless we make some transformation. In Fact, if we write the gain K as follows:

$$K = q\tilde{K}$$

$$= \begin{bmatrix} q_1 \\ \vdots \\ q_m \end{bmatrix} \begin{bmatrix} k_1 & \cdots & k_n \end{bmatrix}$$

where $q_i, i = 1, \cdots, m$ are fixed scalars

Now if we replace this in the characteristic equation we get:

$$\det\left[z\mathbb{I} - A + Bq\tilde{K}\right] = 0$$

Compare this with the single input single output characteristic equation, we conclude that the gain $\tilde{K}$ is determined by:

$$\tilde{K} = \begin{bmatrix} 0 \cdots & 0 & 1 \end{bmatrix}\mathscr{C}^{-1}\Delta(A)$$

with $\mathscr{C} = \begin{bmatrix} Bq & ABq & \cdots & A^{n-1}Bq \end{bmatrix}$.

For the second approach, notice that in the previous chapter we have seen how to compute the transformation, $\eta(k) = Px(k)$ that put the system in the controllable canonical form for multi-input multi output systems. This canonical form is obtained using the controllability matrix. Once this form is obtained the computation of the matrix gain becomes easy. An example showing how this method works is presented later in this chapter.

7.4   Output Feedback Controller Design

It may happen in some circumstances that we don't have complete access to the state vector and therefore, the approach we used for the state feedback control can not be used and an alternate is required for this case. In the rest of this section, we will develop an approach that estimates the state vector and use this estimate as the state vector for the actual control.

In the rest of this section we will firstly focus on the design of the observer that can be used to estimate the state vector which can be used for feedback. Then, we will see how to combine the controller and the observer designs.

Let us consider the following system:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases} \tag{7.10}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$ and $y(k) \in \mathbb{R}^p$ represent respectively the state, the input and the output of the system, A, B and C are known real matrices with appropriate dimensions.

One easy way to build the estimate of the state $x(k)$ is to use the following structure for the estimator:

$$\begin{cases} \hat{x}(k+1) = A\hat{x}(k) + Bu(k) \\ \hat{y}(k) = C\hat{x}(k) \end{cases} \tag{7.11}$$

where $\hat{x}(k) \in \mathbb{R}^n$ is the state estimate of the state vector $x(k)$.

Using (7.10) and (7.11), we get:

$$\begin{aligned} e(k+1) &= x(k+1) - \hat{x}(k+1) = A\left[x(k) - \hat{x}(k)\right] \\ &= Ae(k) \end{aligned}$$

with $e(k) = x(k) - \hat{x}(k)$ is the estimation error.

Notice that the error dynamics doesn't depend on the control $u(k)$ and therefore the behavior of the error will depend on the stability of the matrix A and we have no way to change the behavior to make it faster if it is necessary to guarantee the convergence of the estimator by placing the pole of the matrix A at some appropriate locations. To overcome this we should change the structure of the estimator and a natural one is given by the following dynamics:

$$\begin{cases} \hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L\left[y(k) - \hat{y}(k)\right] \\ \hat{y}(k) = C\hat{x}(k) \end{cases} \tag{7.12}$$

where $\hat{x}(k) \in \mathbb{R}^n$ is the state estimate of the state vector $x(k)$ and L is a constant gain matrix to be designed and that will be referred as the observer gain.

Using again (7.10) and (7.12), we get:

$$\begin{aligned} e(k+1) &= x(k+1) - \hat{x}(k+1) = A\left[x(k) - \hat{x}(k)\right] - LC\left[x(k) - \hat{x}(k)\right] \\ &= \left[A - LC\right]e(k) \end{aligned}$$

with $e(k) = x(k) - \hat{x}(k)$ is the estimation error.

The new dynamics for the estimation error depends on the choice of the gain matrix, L, and therefore the behavior can be controlled by the choice of this observer gain L.

It is important to notice that the eigenvalues of the matrix $A^\top - C^\top L^\top$ are the same as those of the matrix $A - LC$. Therefore, if we denote by $z_1, \cdots, z_n$, the poles that permit the design of the matrix L, the characteristic equation is given by:

$$\det\left[z\mathbb{I} - A^\top + C^\top L^\top\right] = \prod_{l=1}^{n}(z + z_l) \tag{7.13}$$

Now if we compare this characteristic equation with the one of the design of the state feedback controller, we can design the gain matrix L using the Ackerman formula for the following dynamics

$$x(k+1) = A^\top x(k) + C^\top u(k)$$

with the control $u(k) = -L^\top x(k)$.

Based on this and using the Ackerman formula, the observer gain for the single output case is then given by:

$$L^\top = \left[l_1, \cdots, l_n\right] = \left[0, \cdots, 0, 1\right]\left(\mathscr{O}^\top\right)^{-1}\Delta(A^\top)$$

with $\mathscr{O} = \begin{bmatrix} CA^0 \\ \vdots \\ CA^{n-1} \end{bmatrix}$.
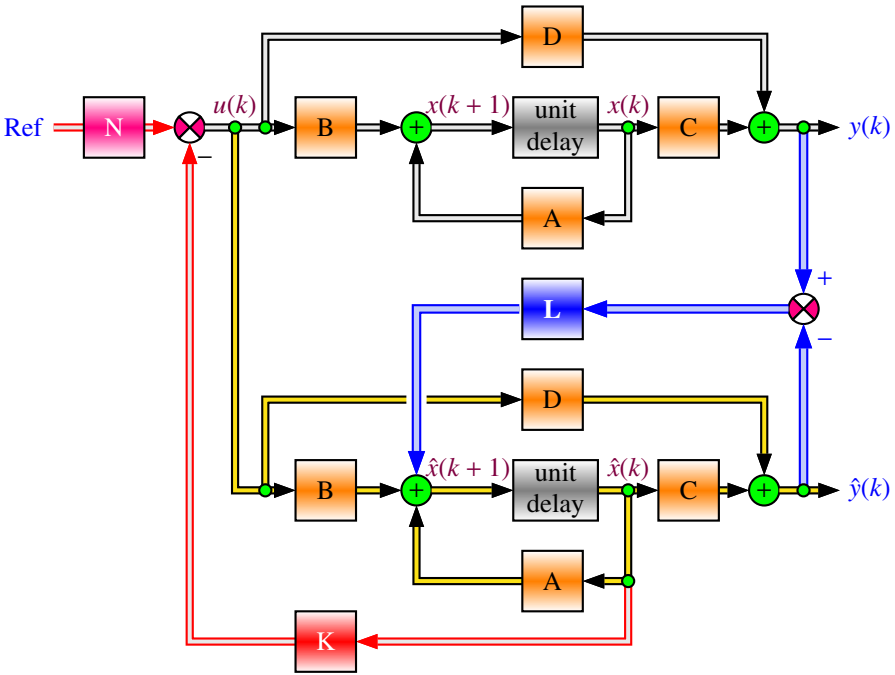
Fig. 7.4 Block diagram of discrete-time linear system

Example 7.4.1 *To show how to design the observer gain, let us consider the position control of a dc motor that drives a mechanical load as discussed before. Let the dynamics between the output θ(t) and the input u(t) be described by the following dynamics:*

$$G(s) = \frac{k}{s(\tau s + 1)}$$

with $k = 1$ and $\tau = 20ms$ represent respectively the gain and the time constant of the system. We also assume that we have only one sensor that measures the output system θ(t).

The system is unstable and has two poles located respectively at 0 and $-\frac{1}{\tau}$. Our aim in this example is to stabilize the closed-loop dynamics and improve the settling time at 5% and to make it equal to 20ms while guaranteeing that the overshoot is less than or equal to 5%.

To solve this design problem, we use the controllable canonical form. This form is given by:

$$x(k + 1) = Fx(k) + Gu(k), x(0) = x_0$$
$$y(k) = Cx(k)$$

with

$$F = \phi(T) = \begin{bmatrix} 1 & \tau\left[1 - e^{-\frac{T}{\tau}}\right] \\ 0 & e^{-\frac{T}{\tau}} \end{bmatrix} = \begin{bmatrix} 1 & 0.0021 \\ 0 & 0.9049 \end{bmatrix}$$

$$G = \Psi(T) = \begin{bmatrix} k\left[T - \tau\left[1 - e^{-\frac{T}{\tau}}\right]\right] \\ k\left[1 - e^{-\frac{T}{\tau}}\right] \end{bmatrix} = \begin{bmatrix} 0.0019 \\ 0.0952 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

From the specifications, we get:

$$d = 100e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}}$$
$$t_s = \frac{3}{\xi\omega_n} = 0.02s$$

which gives in turn:

$$\xi = 0.707$$
$$\omega_n = 212.1641 rad/s$$

this gives the following poles:

$$s_{1,2} = -\xi\omega_n \pm j\omega\sqrt{1 - \xi^2} = -150 \pm j114.8434$$

Their corresponding poles in discrete-time domain when the sampling period T is chosen equal to 2ms are given by:

$$z_{1,2} = e^{s_{1,2}T} = -0.0003 \pm j0.0025$$

The corresponding characteristic polynomial is given by

$$\Delta_d(z) = (z - z_1)(z - z_2) = z^2 - (z_1 + z_2)z + z_1z_2 = z^2 - 0.0006z + 6.3400e - 06$$

Using the Ackerman formula, the observer gain L is given by:

$$L^\top = \begin{bmatrix} l_1, l_2 \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} \mathscr{O}^{-1}\Delta(F^\top)$$

with

$$\mathscr{O}^\top = \begin{bmatrix} CF^0 \\ CF \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1.0000 & -0.0021 \end{bmatrix}$$

$$\Delta(F^\top) = F^2 - 0.0006F + 6.3400e - 06 = \begin{bmatrix} 0.9994 & -0.0040 \\ 0.0000 & 0.8182 \end{bmatrix}$$

The observer gain is:

$$L = \begin{bmatrix} -0.0030 \\ -388.9831 \end{bmatrix}$$

Sometimes, the computation of the observer gain becomes more easier when the dynamics of the system is in observable canonical form. Referring to Chapter 6, it can be shown easily that we have for single input single output system:

$$det\,[zI - A_o + LC_o] = 0, L \in \mathbb{R}^{n\times1}, C_o \in \mathbb{R}^{1\times n}$$

$$A_o - LC_o = \begin{bmatrix} -(a_{n-1}+l_1) & 1 & \dots & 0 & 0 \\ -(a_{n-2}+l_2) & 0 & 1 & \dots & 0 \\ \vdots & & \vdots & \dots & \vdots & \vdots \\ -(a_1+l_{n-1}) & 0 & \dots & 0 & 1 \\ -(a_0+l_n) & 0 & \dots & 0 & 0 \end{bmatrix}$$

The characteristic equation of the system can be written as follows:

$$\Delta_o(z) = z^n + (a_{n-1}+l_1)\,z^{n-1} + \dots + (a_1+l_{n-1})\,z + (a_0+l_n)$$

Also, the desired characteristic equation can be written as follows:

$$\Delta_d(z) = z^n + d_{n-1}z^{n-1} + \dots + d_1 z + d_0$$

Equating these two characteristic equations, we get:

$$l_i = d_{n-i} - a_{n-i}, i =, 1, \dots, n$$

which gives directly the observer gains.

For the multiple outputs (i.e. $C \in \mathbb{R}^{n\times p}$), the design of the observer gain requires the determination of $n \times p$ gains using n equations which is not possible without using smart approaches that fix some of the gains. One possible way to overcome this is to use the following expression for the gain L:

$$L = \begin{bmatrix} l_1 \\ \vdots \\ l_n \end{bmatrix} \begin{bmatrix} q_1, \cdots, q_p \end{bmatrix} = \widetilde{L}q^\top$$

with q_i is an arbitrary real number that has to be chosen by the designer to compute the observer gain, $\widetilde{L}$.

It is important to notice that this approach allows us to determine n-gains by fixing the p-gains. Other approaches are also available and the reader is invited to consult appropriate references for this purpose.

Using the same remark as for the single output case, we can design the gain matrix $\widetilde{L}$ using the Ackerman formula for the following dynamics

$$x(k+1) = A^\top x(k) + C^\top q u(k)$$

with the control $u(k) = -\widetilde{L}^\top x(k)$.

The observer gain for this case is then given by:

$$\widetilde{L}^\top = \begin{bmatrix} l_1, \cdots, l_n \end{bmatrix} = \begin{bmatrix} 0, \cdots, 0, 1 \end{bmatrix} \left(\mathscr{O}^\top \right)^{-1} \Delta(A^\top)$$

with $\mathcal{O} = \begin{bmatrix} q^\top C A^0 \\ \vdots \\ q^\top C A^{n-1} \end{bmatrix}$.

Most often, the state vector is divided into two parts, the first one has the same size of the measured output, i.e. m, while the second has the dimension of $n - m$. If we denote respectively these variables respectively by $x_a(k) \in \mathbb{R}^m$ and $x_b(k) \in \mathbb{R}^{n-m}$, i.e:

$$x(k) = \begin{bmatrix} x_a(k) \\ x_b(k) \end{bmatrix}$$

which implies:

$$\begin{aligned} y(k) &= \begin{bmatrix} \mathbb{I}_{n-m} & 0 \end{bmatrix} x(k) \\ &= x_a(k) \end{aligned}$$

Using this the system dynamics can be rewritten as follows:

$$\begin{aligned} \begin{bmatrix} x_a(k+1) \\ x_b(k+1) \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_a(k) \\ x_b(k) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} \mathbb{I}_{n-m} & 0 \end{bmatrix} x(k) \\ &= x_a(k) \end{aligned}$$

which gives respectively:

$$\begin{aligned} x_a(k+1) &= A_{11} x_a(k) + A_{12} x_b(k) + B_1 u(k) \\ x_b(k+1) &= A_{21} x_a(k) + A_{22} x_b(k) + B_2 u(k) \\ y(k) &= x_a(k) \end{aligned}$$

Notice that the first relation can be rewritten as follows:

$$x_a(k+1) - A_{11} x_a(k) - B_1 u(k) = A_{12} x_b(k)$$

Using now the relation $y(k) = x_a(k)$ for all k, we get:

$$y(k+1) - A_{11} y(k) - B_1 u(k) = A_{12} x_b(k)$$

which implies that the left hand term can be measured, while the right hand term contains the states $x_b(k)$ that have to be estimated.

For the unmeasured part of the state vector, $x_b(k)$, we have

$$\begin{aligned} x_b(k+1) &= A_{21} x_a(k) + A_{22} x_b(k) + B_2 u(k) \\ &= A_{22} x_b(k) + A_{21} x_a(k) + B_2 u(k) \end{aligned}$$

Let us now denote by $\hat{x}_b(k)$ the estimate of the state vector part, $x_b(k)$ and by L_b the observer gain to be designed. Referring to the design of the observer design of full order, and comparing the state equations

$$x(k+1) = Ax(k) + Bu(k)$$

with the dynamics of the unmeasured state parts, we remark that A_{22} and $A_{21}x_a(k) + B_2u(k)$ correspond respectively to A and $Bu(k)$

Doing the same for the output equation:

$$y(k) = Cx(k)$$

with the measurement equation:

$$y(k+1) - A_{11}y(k) - B_1u(k) = A_{12}x_b(k)$$

we remark also that C corresponds to A_{12}.

Using now the results of the full order estimator, we get for the following for the state part, $\hat{x}_b(k)$:

$$\hat{x}_b(k+1) = [A_{22} - L_bA_{12}]\hat{x}_b(k) + A_{21}x_a(k) + B_2u(k)$$
$$+L_b[y(k+1) - A_{21}x_a(k) - B_1u(k)]$$

that we can rewritten using the fact $y(k) = x_a(k)$:

$$\hat{x}_b(k+1) = [A_{22} - L_bA_{12}]\hat{x}_b(k) + L_by(k+1) + [A_{21} - L_bA_{21}]x_a(k)$$
$$+ [B_2 - L_bB_1]u(k)$$

It is important to notice that the presence of the term $y(k+1)$ in the expression of $\hat{x}_b(k+1)$ is not convenient and some changes are needed to overcome this. For this purpose, let us rewrite the previous equation as follows:

$$\hat{x}_b(k+1) - L_by(k+1) = [A_{22} - L_bA_{12}]\hat{x}_b(k) + [A_{21} - L_bA_{21}]x_a(k)$$
$$+ [B_2 - L_bB_1]u(k)$$

Adding and subtract the term $[A_{22} - L_bA_{12}]L_bx_a(k)$ to the right hand side of this relation we get:

$$\hat{x}_b(k+1) - L_by(k+1) = [A_{22} - L_bA_{12}][\hat{x}_b(k) - L_bx_a(k)]$$
$$+ [[A_{22} - L_bA_{12}]L_b + A_{21} - L_bA_{11}]x_a(k)$$
$$+ [B_2 - L_bB_1]u(k)$$

Let us now define the new variable $\rho(k)$ and its estimate $\hat{\rho}(k)$ by:

$$\rho(k) = x_b(k) - L_bx_a(k)$$
$$\hat{\rho}(k) = \hat{x}_b(k) - L_bx_a(k)$$

Using this we get:

$$\hat{\rho}(k+1) = [A_{22} - L_bA_{12}]\hat{\rho}(k)$$
$$+ [[A_{22} - L_bA_{12}]L_b + A_{21} - L_bA_{11}]x_a(k)$$
$$+ [B_2 - L_bB_1]u(k)$$

Let us now define, the estimation error, $e(k)$:

$$e(k) = \rho(k) - \hat{\rho}(k) = x_b(k) - \hat{x}_b(k)$$

The dynamics of the reduced order estimator is given by:

$$
\begin{aligned}
e(k+1) \ &= x_b(k+1) - \hat{x}_b(k+1) \\
&= A_{22}x_b(k) + A_{21}x_a(k) + B_2u(k) - [A_{22} - L_bA_{12}][\hat{x}_b(k) - L_bx_a(k)] \\
&\quad + [[A_{22} - L_bA_{12}]L_b + A_{21} - L_bA_{11}]x_a(k) \\
&\quad + [B_2 - L_bB_1]u(k) \\
&= A_{22}[x_b(k) - \hat{x}_b(k)] + L_bA_{12}\hat{x}_b(k) - L_bA_{12}x_b(k) \\
&= [A_{22} - L_bA_{12}][x_b(k) - \hat{x}_b(k)] \\
&= [A_{22} - L_bA_{12}]e(k)
\end{aligned}
$$

Remark 7.4.1 *It is important to notice that the error expression of the reduced order estimator is similar to the one of the full order estimator except that the size of the matrices is less.*

Example 7.4.2 *In this example we consider a dynamical system with three states with the following dynamics:*

$$
\begin{aligned}
x(k+1) \ &= Ax(k) + Bu(k) \\
y(k) \ &= Cx(k)
\end{aligned}
$$

with

$$
A \ = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0.504 & -1.91 & 2.4 \end{bmatrix},
$$

$$
B \ = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},
$$

$$
C \ = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}
$$

It can be shown that this system is completely controllable and obersvable.

Let us design an observer for this system to estimate the state vector. Firstly let us design a full order estimator. Let us assume that the poles we use for the design of the observer gain are given:

$$
\begin{aligned}
z_1 \ &= 0.1 \\
z_2 \ &= 0.1 + 0.6j \\
z_3 \ &= 0.1 - 0.6j
\end{aligned}
$$

The corresponding characteristic equation is given by:

$$
z^3 - 0.3z^2 + 0.39z - 0.037 = 0
$$

Using pole placement as we did before, it can be shown that the following observer gain is the solution:

$$
L \ = \begin{bmatrix} 2.1 \\ 3.52 \\ 4.904 \end{bmatrix}.
$$

For the reduced order observer, let us assume that we have access to the first state. Based on the theory, we have:

$$A_{11} = \begin{bmatrix} 0 \end{bmatrix}, A_{12} = \begin{bmatrix} 1 & 0 \end{bmatrix}, A_{21} = \begin{bmatrix} 0 \\ 0.504 \end{bmatrix}, A_{22} = \begin{bmatrix} 0 & 1 \\ -1.91 & 2.4 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 0 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Let the poles used for the design of the reduced order observer are:

$$z_2 = 0.1 + 0.6j$$
$$z_3 = 0.1 - 0.6j$$

The corresponding characteristic equation is given by:

$$z^2 - 0.2z + 0.37 = 0$$

Using pole placement as we did before, it can be shown that the following observer gain is the solution:

$$L = \begin{bmatrix} 2.2 \\ 3.75 \end{bmatrix}.$$

Now if we consider simultaneously, the design of the gain observer and the gain controller in order to control the states of the system to have some desire behavior. The control law in this case will use the estimate delivered by the estimator at period k and its expression is given by:

$$u(k) = -K\hat{x}(k) + Nr(k)$$

where $r(k)$ is the reference signal and N a matrix to be designed also.

More frequently the design specifications are given for the controller design only. From these design specifications we extract some poles that are used in the design of the controller gain. For the observer gain, we usually choose some fast poles compared the one used for the design of the controller gain. In general the set of poles used either for the controller gain or the observer gain designs are divided into two parts, the dominant ones and the dominated ones. The dominant poles are in general a pair of complex poles that are extracted from the data related to the overshoot and the settling time.

Now if we combine the system dynamics and the observer one, we get the following augmented dynamics:

$$\begin{cases} \begin{bmatrix} x(k+1) \\ \hat{x}(k+1) \end{bmatrix} = \begin{bmatrix} A & -BK \\ LC & A - LC + BK \end{bmatrix} \begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix} + \begin{bmatrix} BN \\ BN \end{bmatrix} r(k) \\ y(k) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix} \end{cases}$$

Notice that:

$$
\begin{bmatrix} x(k) \\ e(k) \end{bmatrix} = \begin{bmatrix} x(k) \\ x(k) - \hat{x}(k) \end{bmatrix} = \begin{bmatrix} \mathbb{I} & 0 \\ \mathbb{I} & -\mathbb{I} \end{bmatrix} \begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix} = \widetilde{\mathbb{I}} \begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix}
$$

It is easy to verify that $\widetilde{\mathbb{I}}^{-1}$ is equal to $\widetilde{\mathbb{I}}$, and we have:

$$
\begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix} = \begin{bmatrix} \mathbb{I} & 0 \\ \mathbb{I} & -\mathbb{I} \end{bmatrix} \begin{bmatrix} x(k) \\ e(k) \end{bmatrix}
$$

Combining this with the augmented dynamics, we get:

$$
\begin{cases} \begin{bmatrix} x(k+1) \\ e(k+1) \end{bmatrix} = \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix} \begin{bmatrix} x(k) \\ e(k) \end{bmatrix} + \begin{bmatrix} BN \\ 0 \end{bmatrix} r(k) \\ y(k) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ e(k) \end{bmatrix} \end{cases}
$$

From this, we see that the matrix

$$
\begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix}
$$

is triangular and based on algebra theory, the eigenvalues of this matrix correspond to the union of the matrix $A - BK$ and those of the matrix $A - LC$. Consequently it can be concluded that the eigenvalues of the closed-loop dynamics with the state feedback are not affected by the fact that we add the estimator. As a direct conclusion of this, we can design the controller and the estimator separately and this is known in the literature as the *separation principle*.

Example 7.4.3 *To show how we design simultaneously the controller and the observer gains, let us consider the following dynamical system:*

$$
x(k+1) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u(k)
$$

$$
y(k) = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} x(k)
$$

The objective is to design a state feedback that assures the following performances:

- *stable*

- *an overshoot less than 5 %*

- *a settling time at 5 % of 3 s*

First of it is important to notice that the system is unstable since all the poles are at the unit circle.

The controllability and observability matrices are given by:

$$\mathscr{C} = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}$$

$$\mathscr{O} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \\ 1 & 3 & 3 \end{bmatrix}$$

which are both full rank. The system is then controllable and observable. Therefore it is possible to place the poles where we want to guarantee the desired performances.

Let us now convert the performances to desired poles. As we did previously, we have:

$$d = 100e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}}$$

$$t_s = \frac{3}{\xi\omega_n} = 3s$$

From these relations we obtain:

$$\xi = 0.707$$

$$\omega_n = 1.4144$$

which gives the following poles:

$$s_{1,2} = -\xi\omega_n \pm j\omega\sqrt{1-\xi^2} = -1 \pm j$$

Since the order of the system is equal to 3, a third pole can be chosen equal to $s_3 = -5$.

Their corresponding poles in discrete-time domain when the sampling period T is chosen equal to 0.1 s are given by:

$$z_{1,2} = e^{s_{1,2}T} = 0.9003 \pm j0.0903$$

$$z_3 = 0.6065$$

The corresponding characteristic polynomial is given by

$$\Delta_d(z) = (z - z_1)(z - z_2) = z^2 - (z_1 + z_2)z + z_1 z_2 = z^3 - 2.4072z^2 + 1.9109z - 0.4966$$

The controller gain can be computed either using the Ackerman formula or the function **place** of Matlab. The controller gain is given:

$$K = \begin{bmatrix} 0.0071 & 0.4963 & 0.0894 \end{bmatrix}$$

Since we don't have access to all the states, we need to apply the state feedback control to estimate the state. In the rest of this example, we will design an observer for this purpose. The poles we will consider for the observer design are derived from those of the specifications. As we said earlier, these poles can chosen faster

compared the ones used in the controller design. We will consider them four times faster. Based on this, the poles can be chosen as:

$$s_{1,2} = -4 \pm j$$
$$s_3 = -20$$

Their corresponding poles using the same sampling period T are given by:

$$z_{1,2} = e^{s_{1,2}T} = 0.6670 \pm j0.0669$$
$$z_3 = 0.1353$$

The corresponding characteristic polynomial is given by

$$\Delta_d(z) = (z - z_1)(z - z_2) = z^2 - (z_1 + z_2)z + z_1 z_2 = z^3 - 1.4693z^2 + 0.6299z - 0.0608$$

*The controller gain can be computed either using the Ackerman formula or the function **place** of Matlab. The observer gain is given:*

$$L = \begin{bmatrix} 0.9392 \\ 0.0998 \\ 0.5915 \end{bmatrix}$$

```
% Data
A=[1 0 1; 0 1 0; 0 1 1]
B=[1; 1; 1]
C = [1 0 1]
d=0

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[ 0.6065, 0.9003+0.0903*i, 0.9003-0.0903*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)

t=0:0.01:1
u=ones(size(t))
[y,x]=dlsim(A-B*K,B,C,d,u)
stairs(t,x(:,1),'r')
hold on
stairs(t,x(:,2),'b')
stairs(t,x(:,3),'g')
xlabel('Time in seconds')
ylabel('States x1(k), x2(k) and x3(k)')
title('states versus time for a step input')
legend('1','2','3')
```
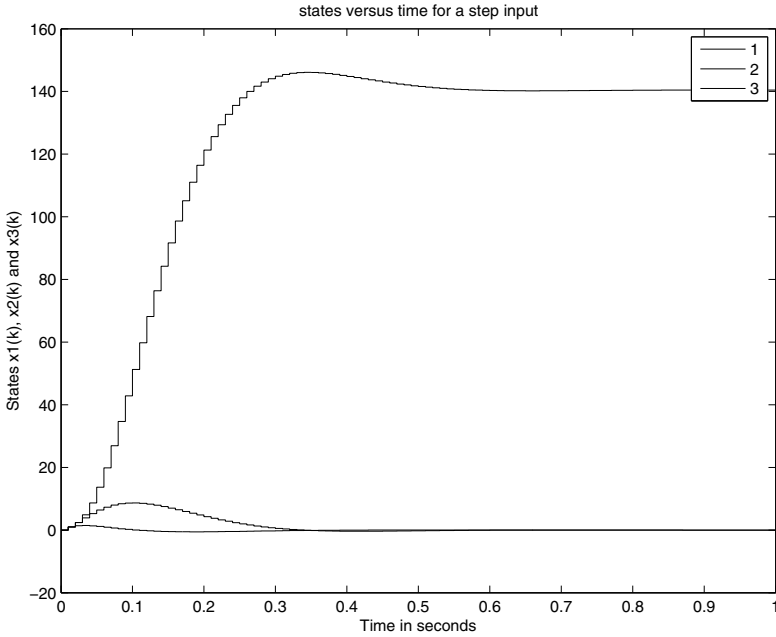
Fig. 7.5 Behavior of the output vs time with state fdk controller

The behavior of the states versus time is illustrated by Fig. (7.5).

Example 7.4.4 *Let us now consider the case on multi input multi output case. For this purpose, we consider the following dynamical system:*

$$x(k+1) = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 1 & 0 \\ 1 & -1 \\ 1 & 1 \\ 1 & -1 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} x(k)$$

First of all it is important to notice that this system is unstable. Our goal in this example is to design a state feedback controller that stabilizes the system and place the poles of the closed-loop of this system at 0.2, 0.1 and 0.2 ± 0.2 j.

To design the state feedback controller, we will search for the transformation η(k) = Px(k) that gives the controllable canonical form and then use the procedure we presented earlier to design the controller gain.

It is important to notice that we don't have access to all the states and therefore an observer is required to estimate the state for feedback. To design the controller and the observer gains, the system must be controllable and observable. The

system is of order 4 and has two inputs and two outputs. The controllability and the observability matrices are given by:

$$\mathscr{C} = \left[B \ AB \ A^{4-2}B \right]$$

$$= \begin{bmatrix} 1 & 0 & 0 & 1 & 2 & -1 \\ 1 & -1 & 2 & -2 & 4 & -3 \\ 1 & 1 & 2 & 0 & 4 & -2 \\ 1 & -1 & 2 & -1 & 4 & 0 \end{bmatrix}$$

$$\mathscr{O} = \begin{bmatrix} C \\ CA \\ CA^{4-2} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ -1 & 1 & 2 & 0 \\ 1 & 2 & 1 & 2 \\ 1 & 3 & 3 & 1 \\ 3 & 3 & 2 & 4 \end{bmatrix}$$

The ranks of these matrices are equal to four and therefore the system is controllable and observable.

Let us firstly focus on the design of the state feedback controller. For this purpose since the system in multi-input multi-output, we need to transform the actual description to a controllable canonical form. To determine the controllability indices, notice that by inspection of the fourth columns are linearly independent and therefore the matrix

$$\mathscr{C}_o = \left[b_1 \ Ab_1 \ b_2 \ Ab_2 \right]$$

$$= \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 2 & -1 & -2 \\ 1 & 2 & 1 & 0 \\ 1 & 2 & -1 & -1 \end{bmatrix}$$

From this we conclude that the controllability indices associated to the first and the second columns of the matrix B are equal to 2 and therefore controllablity index of the system is equal to 2.

The inverse of this matrix is given by:

$$\mathscr{C}_o^{-1} = \begin{bmatrix} 1 & 1 & 0 & -1 \\ -0.5 & -0.75 & 0.25 & 1 \\ 0 & 0.5 & 0.5 & -1 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

From this we get:

$$q_1 = \left[-0.5 \ -0.75 \ 0.25 \ 1 \right]$$

$$q_2 = \left[0 \ -1 \ 0 \ 1 \right]$$

The matrix P of the transformation, $\eta(k) = Px(k)$ is given by:

$$P = \begin{bmatrix} q_1 \\ q_1 A \\ q_2 \\ q_2 A \end{bmatrix}$$

$$= \begin{bmatrix} -0.5 & -0.75 & 0.25 & 1 \\ 1.5 & -0.5 & -0.25 & 0.25 \\ 0 & -1 & 0 & 1 \\ 1 & -1 & 0 & 0 \end{bmatrix}$$

Its inverse is given by:

$$P^{-1} = \begin{bmatrix} 1 & 1 & -1.25 & 0 \\ 1 & 1 & -1.25 & -1 \\ 5 & 1 & -5.25 & 1 \\ 1 & 1 & -0.25 & -1 \end{bmatrix}$$

The new description is given by:

$$\eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k)$$
$$y(k) \quad \bar{C}\eta(k)$$

with

$$\bar{A} = PAP^{-1}$$
$$= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 4 & -1 & -4 & 2.25 \\ 0 & 0 & 0 & 1 \\ 2 & -2 & -2.5 & 3 \end{bmatrix}$$

$$\bar{B} = PB$$
$$= \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\bar{C} = CP^{-1}$$
$$= \begin{bmatrix} 6 & 2 & -6.5 & 1 \\ 7 & 3 & -6.75 & -1 \end{bmatrix}$$

We are now ready to design the controller gain. First of all notice that the system has four states and two inputs. Therefore, the controller gain has 2×4 components:

$$\bar{K} = \begin{bmatrix} \bar{k}_{11} & \bar{k}_{12} & \bar{k}_{13} & \bar{k}_{14} \\ \bar{k}_{21} & -\bar{k}_{22} & \bar{k}_{23} & \bar{k}_{24} \end{bmatrix}$$

Notice that $\bar{B}\bar{K}$ *is given by:*

$$\bar{B}\bar{K} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \bar{k}_{11} & \bar{k}_{12} & \bar{k}_{13} & \bar{k}_{14} \\ 0 & 0 & 0 & 0 \\ \bar{k}_{21} & \bar{k}_{22} & \bar{k}_{23} & \bar{k}_{24} \end{bmatrix}$$

Using this we get:

$$\bar{A} - \bar{B}\bar{K} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 4 - \bar{k}_{11} & -1 - \bar{k}_{12} & -4 - \bar{k}_{13} & 2.25 - \bar{k}_{14} \\ 0 & 0 & 0 & 1 \\ 2 - \bar{k}_{21} & -2 - \bar{k}_{22} & -2.5 - \bar{k}_{23} & 3 - \bar{k}_{24} \end{bmatrix}$$

From the other side notice that we can have the two separate following characteristic polynomials:

$$(z - 0.1)(z - 0.2) = z^2 - 0.3z + 0.02$$
$$(z - 0.2 - 0.2j)(z - 0.2 - 0.2j) = z^2 - 0.4z + 0.08$$

From which we can construct the following matrix that will have the same desired poles:

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ -0.02 & 0.3 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -0.08 & 0.4 \end{bmatrix}$$

Equating these two matrices gives the following gain:

$$\bar{K} = \begin{bmatrix} 4.02 & -1.3 & -4 & 2.25 \\ 2.0 & -2.0 & -2.42 & 2.6 \end{bmatrix}$$

It can be checked that the closed-loop dynamics $\bar{A} - \bar{B}\bar{K}$ *has the desired poles.*

To get now the gain, K, that goes with the original description, we use the following relation:

$$K = \bar{K}P$$

which gives:

$$\bar{K} = \begin{bmatrix} -1.7100 & -0.6150 & 1.3300 & -0.3050 \\ -1.4000 & -0.6800 & 1.0000 & -0.9200 \end{bmatrix}$$

It can be checked that the closed-loop dynamics $A - BK$ *has the desired poles.*

```
% Data
A=[1 0 1; 0 1 0; 0 1 1]
B=[1; 1; 1]
C = [1 0 1]
d=0
```

```
% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K=place(A,B,[ 0.6065, 0.9003+0.0903*i, 0.9003-0.0903*i])

% Check that the eigenvalues are equals to the desired poles
eig(A-B*K)

t=0:0.01:1
u=[ones(size(t)); ones(size(t))]
[y,x]=dlsim(A-B*K,B,C,d,u)
stairs(t,x(:,1),'r')
hold on
stairs(t,x(:,2),'b')
stairs(t,x(:,3),'g')
stairs(t,x(:,4),'g')
xlabel('Time in seconds')
ylabel('States x1(k), x2(k), x3(k) and x4(k)')
title('states versus time for a step input')
legend('1','2','3','4')
```

If we assume that we complete access to the states, the simulation results under controller gain are illustrated in Fig. (7.6).

Let us now focus on the design of the observer. For this purpose we consider poles for the design of the observer gain faster than those used for the design of the controller gains. Le us select the followins ones:

$$
\begin{aligned}
z_1 &= 0.01 \\
z_2 &= 0.02 \\
z_{3,4} &= 0.1 \pm 0.1 j
\end{aligned}
$$

To design the observer gain, we need to search for the transformation that gives the observable canonical form. For this purpose, we use the dual principle. The dual system is

$$
\begin{aligned}
x(k+1) &= A^\top x(k) + C^\top u(k) \\
y(k) &= B^\top x(k)
\end{aligned}
$$
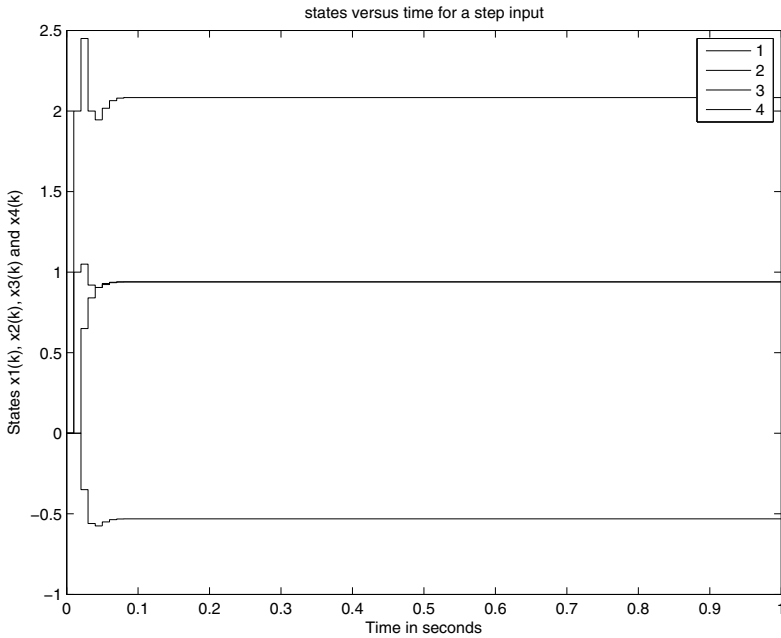
Fig. 7.6 Behavior of the output vs time with state fdk controller

If we let c_1 and c_2 denote respectively the two columns of the matrix $C^\top$, the controllability matrix is given by:

$$\mathscr{C} = \begin{bmatrix} c_1 \ A^\top c_1 \ c_2 \ A^\top c_2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1 & 0 & 1 \\ 0 & 1 & 1 & 2 \\ 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

Its inverse is given by:

$$\mathscr{C} = \begin{bmatrix} 0.5 & -0.5 & 0.5 & 0 \\ 0 & 1 & 0 & -1 \\ -1 & -3 & 1 & 3 \\ 0.5 & 1.5 & -0.5 & 1 \end{bmatrix}$$

From this we get:

$$q_1 = \begin{bmatrix} 0 & 1 & 0 & -1 \end{bmatrix}$$
$$q_2 = \begin{bmatrix} 0.5 & 1.5 & -0.5 & -1 \end{bmatrix}$$

The matrix P of the transformation, $\eta(k) = Px(k)$ is given by:

$$
P = \begin{bmatrix} q_1 \\ q_1 A^\top \\ q_2 \\ q_2 A^\top \end{bmatrix}
$$

$$
= \begin{bmatrix} 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0.5 & 1.5 & -0.5 & -1 \\ -1 & 0.5 & 1 & -0.5 \end{bmatrix}
$$

Its inverse is given by:

$$
P^{-1} = \begin{bmatrix} -3 & 1 & 2 & 0 \\ -2.5 & 0 & 2 & 1 \\ -3.5 & 1 & 2 & 1 \\ -3.5 & 0 & 2 & 1 \end{bmatrix}
$$

The new description is given by:

$$
\eta(k+1) = \bar{A}^\top \eta(k) + \bar{C}^\top u(k)
$$
$$
y(k) \quad \bar{B}^\top \eta(k)
$$

with

$$
\bar{A}^\top = PA^\top P^{-1}
$$
$$
= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -0.5 & 2 & 0 & -1 \\ 0 & 0 & 0 & 1 \\ -6 & 3.5 & 4 & 0 \end{bmatrix}
$$
$$
\bar{C}^\top = PC^\top
$$
$$
= \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}
$$
$$
\bar{B} = BP^{-1}
$$
$$
= \begin{bmatrix} -12.5 & 2 & 8 & 3 \\ 2.5 & 1 & -2 & -1 \end{bmatrix}
$$

Using the dual principle we get the observable canonical form:

$$
\eta(k+1) = \bar{A}\eta(k) + \bar{B}u(k)
$$
$$
y(k) \quad \bar{C}\eta(k)
$$

with

$$\bar{A} \;=\; \begin{bmatrix} 0 & -0.5 & 0 & -6 \\ 1 & 2 & 0 & 3.5 \\ 0 & 0 & 0 & 4 \\ 0 & -1 & 1 & 0 \end{bmatrix}$$

$$\bar{B} \;=\; \begin{bmatrix} -12.5 & 2.5 \\ 2.0 & 1.0 \\ 8.0 & -20 \\ 3.0 & -1.0 \end{bmatrix}$$

$$\bar{C} \;=\; \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We are now ready to design the observer gain, L. First of all notice that the system has four states and two inputs. Therefore, the controller gain has 4×2 components:

$$\bar{L} \;=\; \begin{bmatrix} \bar{l}_{11} & \bar{l}_{12} \\ \bar{l}_{21} & \bar{l}_{22} \\ \bar{l}_{31} & -\bar{l}_{32} \\ \bar{l}_{41} & \bar{l}_{42} \end{bmatrix}$$

Notice that $\bar{L}\bar{C}$ is given by:

$$\bar{L}\bar{C} \;=\; \begin{bmatrix} 0 & \bar{l}_{11} & 0 & \bar{l}_{12} \\ 0 & \bar{l}_{21} & 0 & \bar{l}_{22} \\ 0 & \bar{l}_{31} & 0 & \bar{l}_{32} \\ 0 & \bar{l}_{41} & 0 & \bar{l}_{42} \end{bmatrix}$$

Using this we get:

$$\bar{A} - \bar{L}\bar{C} \;=\; \begin{bmatrix} 0 & -0.5 - \bar{l}_{11} & 0 & -6 - \bar{l}_{12} \\ 1 & 2 - \bar{l}_{21} & 0 & 3.5 - \bar{l}_{22} \\ 0 & -\bar{l}_{31} & 0 & 4 - \bar{l}_{32} \\ 0 & -1 - \bar{l}_{41} & 1 & -\bar{l}_{42} \end{bmatrix}$$

From the other side notice that we can have the two separate following characteristic polynomials:

$$(z - 0.01)(z - 0.02) = z^2 - 0.03z + 0.0002$$
$$(z - 0.1 - 0.1j)(z - 0.1 - 0.1j) = z^2 - 0.2z + 0.02$$

From which we can construct the following matrix that will have the same desired poles:

$$\begin{bmatrix} 0 & -0.0002 & 0 & 0 \\ 1 & 0.03 & 0 & 0 \\ 0 & 0 & 0 & -0.02 \\ 0 & 0 & 1 & 0.2 \end{bmatrix}$$

Equating these two matrices gives the following gain:

$$\bar{L} \;=\; \begin{bmatrix} -0.4998 & -6 \\ 1.97 & 3.5 \\ 0.0 & 4.02 \\ -1.0 & -0.2 \end{bmatrix}$$

It can be checked that the closed-loop dynamics $\bar{A} - \bar{L}\bar{C}$ has the desired poles.

Notice that the poles are solution of $\det\left(P^{-\top}AP^{\top} - \bar{L}CP^{\top}\right) = 0$. This relation can be transformed as:

$$\det\left(P^{-\top}AP^{\top} - \bar{L}CP^{\top}\right) = \det\left(P^{-\top}\left(A - P^{\top}\bar{L}C\right)P^{\top}\right) = \det\left(A - LC\right)$$

To get now the gain, L, that goes with the original description, we use the following relation:

$$L = P^{\top}\bar{L}$$

which gives:

$$\bar{L} \;=\; \begin{bmatrix} 1.0000 & 2.2100 \\ -0.9998 & -0.0700 \\ 0.9700 & 1.2900 \\ -0.9702 & -1.4200 \end{bmatrix}$$

It can be checked that the closed-loop dynamics $A - LC$ has the desired poles.

7.5   Linear Quadratic Regulator

In the previous section we presented techniques that allowed us to design state feedback controllers to guarantee some desired specifications without fixing a criteria for this choice. In this section we will try to design the optimal state feedback that respects a certain fixed criteria.

Let us focus first of all on the finite horizon optimal control problem. The cost function is given by:

$$J \;=\; \sum_{k=0}^{N-1}\left[x^{\top}(k)Qx(k) + u^{\top}(k)Ru(k)\right] + x^{\top}(N)S\,x(N), \tag{7.14}$$

where the matrices Q, S and R are respectively symmetric and semi-positive-definite, and symmetric and positive definite.

Remark 7.5.1 *The matrix R is supposed to be symmetric and positive-definite because as we will see at the design phase of the optimal controller, we need to compute the inverse of this matrix. The first term in the cost is used to penalize the state and the second one for the control. The last term is used to penalize the final state. For general matrices Q and R it is difficult to give an explanation, but for diagonal matrices the highest coefficient in the diagonal of the appropriate matrix will give the smallest either state or control.*

The linear regulator problem can be stated: given a linear time-invariant system with the following dynamics:

$$x(k + 1) = Ax(k) + Bu(k), x(0) = x_0$$

find a control law:

$$u^\star(k) = Kx^\star(k)$$

that minimizes the cost function (7.14).

To solve this optimization problem three approaches can be used among them we quote the dynamic programming approach known in the literature also as the Bellman principle [3].

To establish the optimality conditions that will give us the optimal solution we proceed recursively. For this purpose, if the initial state is $x(N - 1)$ and we want to drive it to the final state $x(N)$. In this case, the cost becomes:

$$J_{N-1,N} = x^\top(N - 1)Qx(N - 1) + u^\top(N - 1)Ru(N - 1) + x^\top(N)S x(N)$$

Using the system dynamics, we can rewritten this as follows:

$$\begin{aligned} J_{N-1,N} &= x^\top(N - 1)Qx(N - 1) + u^\top(N - 1)Ru(N - 1) \\ &+ [Ax(N - 1) + Bu(N - 1)]^\top S [Ax(N - 1) + Bu(N - 1)] \end{aligned} \quad (7.15)$$

where the variable decision is $u(N - 1)$ and that we would like to determine and that makes the criteria smaller.

Using the fact that the cost is continuous in the decision variable and the necessary condition for optimality (see [3]), i.e:

$$\frac{\partial J_{N-1,N}}{\partial u(N - 1)} = 0$$

we get:

$$Ru(N - 1) + B^\top S [Ax(N - 1) + Bu(N - 1)] = 0$$

which gives in turn the optimal control, $u^\star(N - 1)$, to drive the state from $x(N - 1)$ to $x(N)$:

$$u^\star(N - 1) = - \left[R + B^\top S B\right]^{-1} B^\top S Ax(N - 1) \quad (7.16)$$

It is well known from optimization theory that this solution will be the minimizing one if the following holds:

$$\frac{\partial^2 J_{N-1,N}}{\partial u^2(N - 1)} = R + B^\top S B > 0$$

From Eq. (7.16), we can see that the control is well a state feedback one that we can rewrite as:

$$u^\star(N - 1) = K(N - 1)x(N - 1)$$

with $K(N - 1) = - \left[R + B^\top S B\right] B^\top S A$.

The corresponding cost, $J^\star_{N-1,N}$ is given by:

$$
\begin{aligned}
J^\star_{N-1,N} &= x^\top(N-1)Qx(N-1) + x^\top(N-1)K^\top(N-1)RK(N-1)x(N-1) \\
&\quad + [Ax(N-1) + BK(N-1)x(N-1)]^\top S [Ax(N-1) + BK(N-1)x(N-1)] \\
&= x^\top(N-1)\Big[[A+BK(N-1)]^\top S [A+BK(N-1)] + Q \\
&\quad +K^\top(N-1)RK(N-1)\Big]x(N-1)
\end{aligned}
$$

that we can rewrite as follows:

$$
J^\star_{N-1,N} = x^\top(N-1)S_{N-1}x(N-1)
$$

with $S_{N-1} = [A + BK(N-1)]^\top S [A + BK(N-1)] + Q + K^\top(N-1)RK(N-1)$

Notice that by this choice we have:

$$
J_{N,N} = J^\star_{N,N} = x^\top(N)S x(N)
$$

and therefore we have $S_N = S$.

If now we consider another step backward and using the principle of optimality, we have:

$$
J^\star_{N-2,N} = J_{N-2,N-1} + J^\star_{N-1,N}
$$

The expression of $J_{N-2,N}$ is given by:

$$
\begin{aligned}
J^\star_{N-2,N} &= \Big[[Ax(N-2) + Bu(N-2)]^\top S [Ax(N-2) + Bu(N-2)] \\
&\quad +Bu(N-2) + x^\top(N-2)Qx(N-2) + u^\top(N-2)Ru(N-2)\Big]
\end{aligned}
$$

Proceeding similarly as before we get:

$$
u^\star(N-2) = K(N-2)x(N-2)
$$

with $K(N-2) = - [R + B^\top S_{N-1}B]^{-1} B^\top S_{N-1}A$.

In a similar manner if we would like to drive the state from $x(k)$ to $x(N)$, we get:

$$
u^\star(k) = K(k)x(k)
$$

with $K(k) = - [R + B^\top S_{k+1}B]^{-1} B^\top S_{k+1}A$, and

$$
S_k = [A + BK(k)]^\top S_{k+1} [A + BK(k)] + Q + K^\top(k)RK(k)
$$

To get the solution of this optimization problem we should solve backward the following equation:

$$
S_k = [A + BK(k)]^\top S_{k+1} [A + BK(k)] + Q + K^\top(k)RK(k) \tag{7.17}
$$

with the following initial condition

$$
S_N = S
$$

The corresponding optimal control at each step is given by:

$$
u^\star(k) = K(k)x(k) \tag{7.18}
$$

with $K(k) = - [R + B^\top S_{k+1}B]^{-1} B^\top S_{k+1}A$

Example 7.5.1 *To show how to solve the optimal control problem for a finite horizon, let us consider the following dynamical system:*

$$x(t + 1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Let the weighting matrices Q, R and S be given by:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

$$R = \begin{bmatrix} 10 \end{bmatrix}$$

$$S = \begin{bmatrix} 5 & 0 \\ 0 & 4 \end{bmatrix}$$

It is important to notice that the system has two states and therefore the gain K has 1×2 components. First of all notice that it is difficult to solve this optimization by hand. We will use Matlab to solve the optimality condition. For this purpose, we write a Matlab program.

First of all, it is important to notice that the system in unstable since one of its poles is outside the unit circle. Our objective is then to search for a stabilizing controller. For this purpose, we will use the optimal control approach. The optimization problem is a finite horizon with $N = 10$. To get the gain of the controller we use the Matlab program below. This gain is obtained after a certain number of iterations. As it is shown by Fig. (7.7), the two components of the gain converge to finite values. These values are then used for simulation. It can be verified that the closed-loop dynamics has all its poles inside the unit circle and therefore, we stabilize the system using the optimal control. The steady state gain is given by:

$$K = \begin{bmatrix} -1.8465 & -3.2720 \end{bmatrix}.$$

```
% Data
A=[0 1;  2 3 ]
B=[0; 1]
C=[1 0]
d=[0]

n=size(A,1);
m=size(B,2);
```

```
Q=[1 0; 0 2]
R=[10]
S=[5 0; 0 4]

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation

S = S
K = -inv(R+B'*S*B)*B'*S*A

M=[S]
N=[K]

for i=9:-1:1,
     Sa = (A+B*K)'*S*(A+B*K)+Q+K'*R*K
     K=-inv(R+B'*Sa*B)*B'*Sa*A

     S=Sa
     M=[M; S]
     N=[N; K]
end

% Plot
k=10:-1:1
stairs(k,N(:,1),'r')
hold on
stairs(k,N(:,2),'b')
xlabel('Iterations')
ylabel('Gains k(1) and k(2)')
title('Gains versus iterations (Backward)')
legend('1','2')

print -deps chap5-fig.8.eps

pause

t=0:0.01:0.5
u=ones(size(t))
[y,x]=dlsim(A+B*K,B,C,d,u)
stairs(t,x(:,1),'r')
hold on
stairs(t,x(:,2),'b')
xlabel('Time in seconds')
ylabel('States x1(k) and x2(k)')
title('states versus time for a step input')
```

```
legend('1','2')

print -deps chap5-fig.9.eps
```

The behavior of the controller gains versus iteration is illustrated by Fig. 7.7.
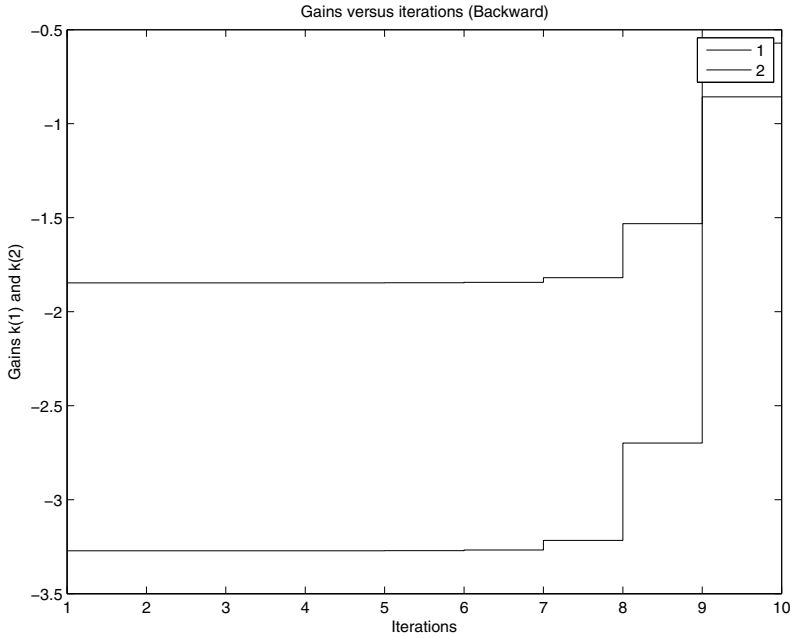
Fig. 7.7 Behavior of the controller gains versus iteration

The behavior of the states versus time is illustrated by Fig. 7.8.

For the infinite horizon case, the cost function becomes:

$$J = \sum_{k=0}^{\infty} \left[x^{\top}(k)Qx(k) + u^{\top}(k)Ru(k) \right],$$ (7.19)

Now if we assume the optimal cost is given by:

$$J^{\star}(x_k) = x_k^{\top} P x_k$$

where P is an unknown matrix.

Notice that the cost can be rewritten as follows:

$$x_k^{\top} P x_k = x^{\top}(k)Qx(k) + u^{\top}(k)Ru(k) + \sum_{l=k+1}^{\infty} \left[x^{\top}(l)Qx(l) + u^{\top}(l)Ru(l) \right]$$

$$= x^{\top}(k)Qx(k) + u^{\top}(k)Ru(k) + x_{k+1}^{\top} P x_{k+1}$$
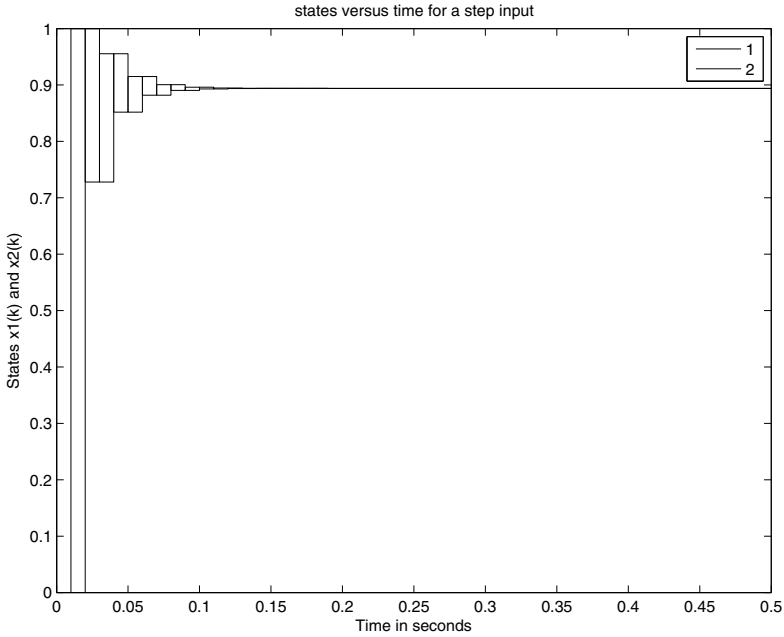
Fig. 7.8 Behavior of the output vs time with state fdk controller

Using now the expression of the control and the system dynamics, we get:

$$J(x_k) = x^\top(k)Qx(k) + u^\top(k)Ru(k) + [Ax_k + Bu_k]^\top P[Ax_k + Bu_k]$$

Based on optimality conditions we get:

$$0 = \frac{\partial J}{\partial u_k}(x_k) = Ru(k) + B^\top P[Ax_k + Bu_k]$$

that gives the optimal control law:

$$u^\star(k) = -\left[R + B^\top PB\right]^{-1} B^\top PAx(k)$$
$$= -Kx(k)$$

with $K = [R + B^\top PB]^{-1} B^\top PA$.

Using this expression for the control law and the previous one for the cost function $J(x(k))$, we get the following that must holds for all $x(k)$:

$$x^\top(k)\left[(A - BK)^\top P(A - BK) - P + Q + KRK\right]x(k) = 0$$

This implies in turn the following:

$$(A - BK)^\top P(A - BK) - P + Q + KRK = 0$$

Replacing K by its expression we obtain the following Ricatti equation:

$$A^\top PA - P + Q - A^\top PB\left(R + B^\top PB\right)^{-1} B^\top PA = 0$$

Example 7.5.2 *To show how to solve the optimal control problem for a finite horizon, let us consider the following dynamical system:*

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with:

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ -5 & -1 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 1 & 2 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Let the weighting matrices Q, R and S be given by:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

$$R = \begin{bmatrix} 5 & 0 \\ 0 & 8 \end{bmatrix}$$

It is important to notice that the system has two inputs and therefore the gain K has 2×3 components. First of all notice that it is difficult to solve this optimization by hand. We will use Matlab to solve. For this purpose, we write a Matlab program.

```
% Data
A=[1 1 0; 0  0 1; -5 -1 -1]
B=[-1 0; 1 -1; 1 2]
C=[1 0 1; 1 1 1]

n=size(A,1);
m=size(B,2);

D=zeros(2)
```

```
Q=[1 0 0; 0 2 0; 0 0 3]
R=[5 0 ;  0 10]

% Check the controllablity
CO=ctrb(A,B)
rank(ctrb(A,B))

% Controller gain computation
K = dlqr(A,B,Q,R)
Ac = [(A-B*K)];
Bc = [B];
Cc = [C];
Dc = [D];

t=0:0.01:2;
u=[ones(size(t)); ones(size(t))];
[y,x]=dlsim(Ac,Bc,Cc,Dc,u);

% plot the outputs
stairs(t,y(:,1),'r')
hold on
stairs(t,y(:,2),'b')
xlabel('Time in seconds')
 ylabel('Outputs y1(k) and y2(k)')
 title('Outputs versus time for a step input')
 legend('1','2')

print -deps chap5-fig.10.eps

 pause
 % Plot the states

 stairs(t,x(:,1),'r')
 hold on
 stairs(t,x(:,2),'b')
 stairs(t,x(:,3),'g')
 xlabel('Time in seconds')
 ylabel('States x1(k), x2(k) and x3(k)')
 title('states versus time for a step input')
 legend('1','2','3')

 print -deps chap5-fig.11.eps
```

The behavior of the ouputs and the states versus time is illustrated by Figs. 7.9-7.10.
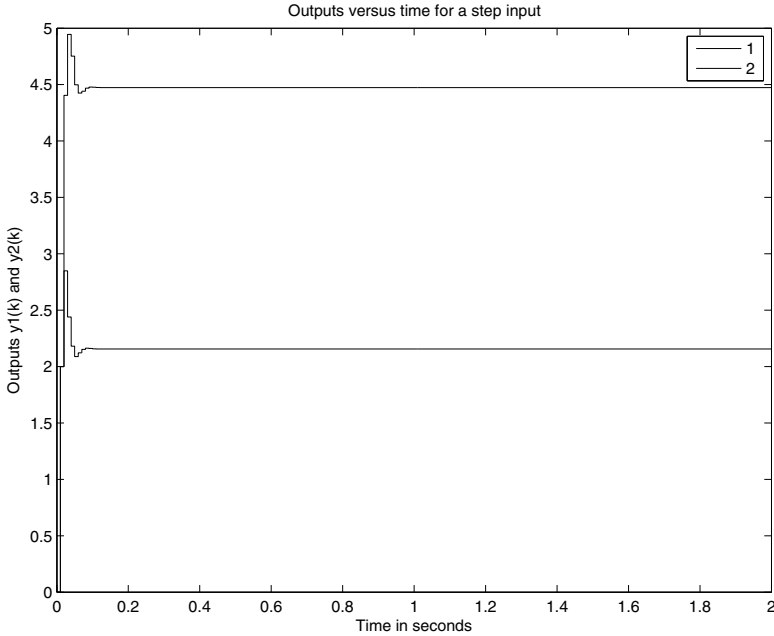
Fig. 7.9 Behavior of the output vs time with state fdk controller

7.6 Case Study

In this case study we present the design of controllers for our dc motor kit. The mathematical model for this system is described by the following transfer function:

$$G(s) = \frac{K_m}{s(\tau_m s + 1)}$$

where $K_m = 48.5$ and $\tau_m = 0.06$ s.

It can be shown that canonical controllable form is given by:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0 \\ y(t) = Cx(t) \end{cases} \tag{7.20}$$

where $x(t)$ and $u(t)$ are respectively the state vector and the control input and the matrices A, B and C are given by:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau_m} \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{K_m}{\tau_m} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Previously, it was shown that this system is controllable and observable. For the design of the state feedback controller, we will present the pole placement technique and the optimal control technique. The design will be done in the continuous-time.
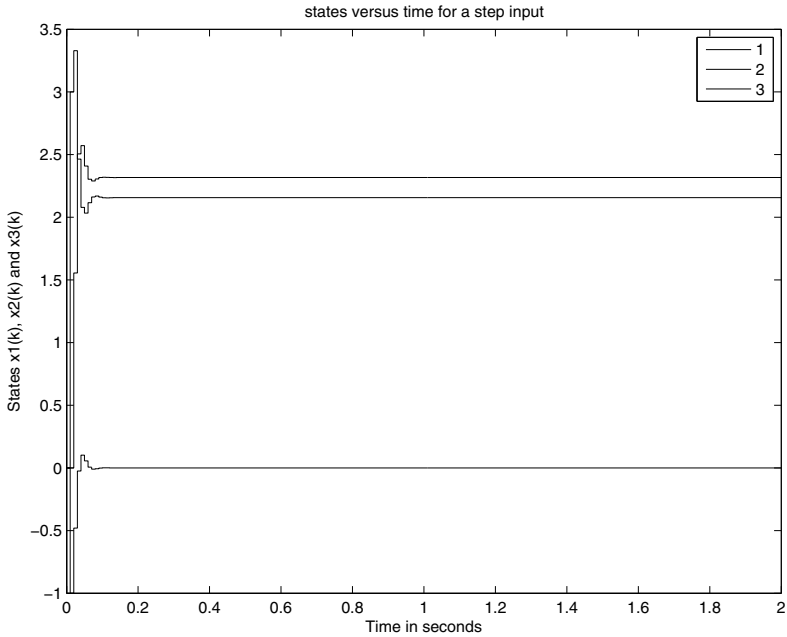
Fig. 7.10 Behavior of the states vs time with state fdk controller

Let us first of all focus on the pole placement technique and design the controller that assures the following performances:

1. system stable

2. overshoot less or equal to 5 %

3. settling time at 5 % less or equal to 0.05 s.

If we assume the complete access to the states, the controller is then given by:

$$u(t) = -Kx(t)$$

where the gain K is to be determined.

From these performances, it can be shown that the pair of complex dominant poles is given by:

$$s_{1,2} = -60 \pm 60j$$

If we denote by $\Delta_d(s)$ the desired characteristic equation, the gain K is the solution of the following equation:

$$\det[s\mathbb{I} - A + BK] = \Delta_d(s)$$

We can either solve analytically this equation or use the Matlab function **acker**. If the desired poles are denoted by p, the following syntax will give us the solution:

```
K = acker(A,B,p)
```

If we use the data for our system, we get:

$$K = \begin{bmatrix} 8.9813 \ 0.1289 \end{bmatrix}$$

If we simulate the system with this gain, we get the results illustrated by Fig. 7.11 shows that the controller gives the desired performances.
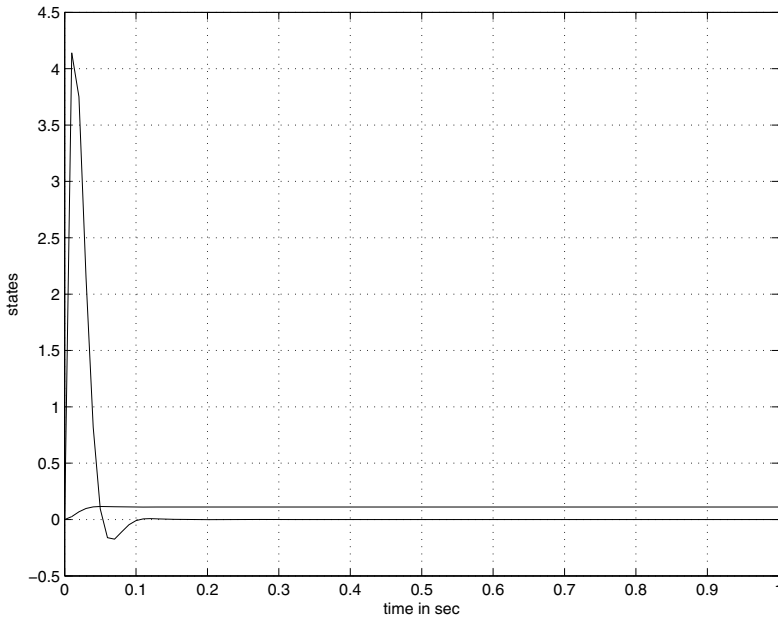
Fig. 7.11 Behavior of the states vs time with state fdk controller

The case of partially known states can be done similarly and we refer the reader to the case study part for this purpose. Let us now design the optimal controller that stabilizes our system. The matrices Q and R are chosen as:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix},$$
$$R = \begin{bmatrix} 1 \end{bmatrix}$$

Using the **lqr**, we get:

$$K = \begin{bmatrix} 1.0000 \ 2.2159 \end{bmatrix}$$
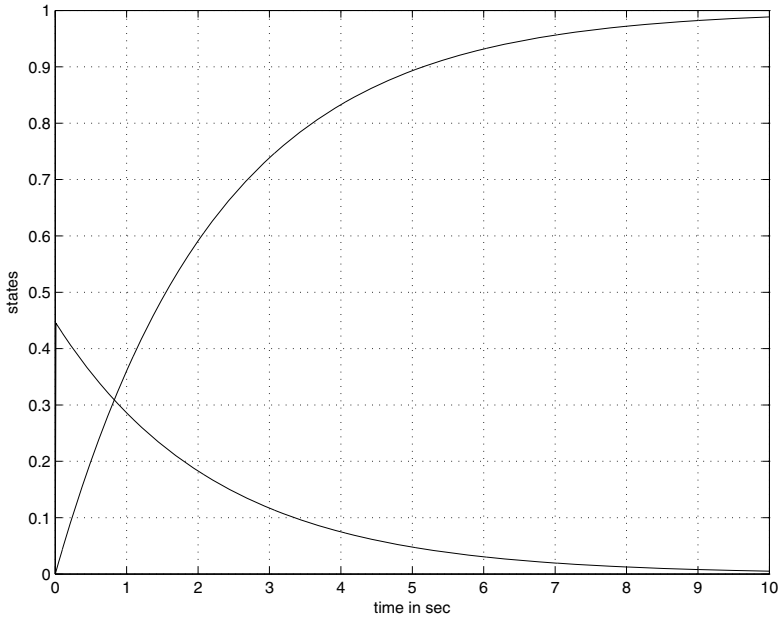
Fig. 7.12 Behavior of the states vs time with state fdk controller

The simulation results for the system with this gain are illustrated by Fig. 7.12:

It is important to remark that the choice of the matrix Q and R will affect the solution. We let this to the reader to investigate this matter.

7.7   Conclusions

This chapter covers the control design problem for mechatronic systems. Mainly we focused on the design of the state feedback controller. In case of complete access to the sate vector two approaches, pole placement and optimal control, were presented to deal with the design of the state feedback controller gain. Many numerical example were given to show how the different techniques work.

7.8   Problems

1. Consider a dynamical system with the following dynamics:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

(a) develop the different canonical forms
(b) study the stability, the controllability and the observability of each form
(c) establish their equivalent discrete time forms when the sampling period T is fixed to 0.1
(d) design a state feedback that gives an overshoot about 5 % and a settling time at 5 % about one second
(e) establish the solution of the closed-loop when the inputs are fixed to unit steps
(f) plot the phase diagram the behavior of the states
(g) compute the transfer matrix of the system

2. For the system of the Problem 1,

(a) establish the Riccati equation for the discrete-time and continuous-time cases
(b) using the function lqr of Matlab, determine the controller gain for the following weighting matrices:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix},$$

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix},$$

and plot the state behavior with respect of time t
(c) study the impact of the matrices Q and R on the controller gain and the states.

3. Let us consider the following dynamics system:

$$x(t + 1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$$

(a) check the controllability and the observability of this system
(b) establish the controllable and the observable canonical forms
(c) assume that we have complete access to the state vector, design a state feedback controller that place the poles of the closed-loop dynamics at 0.01 and $0.1 \pm 0.1j$
(d) design simultaneously the controller and the observer that give the same poles for the closed-loop dynamics.
(e) write a Matlab program that simulate the step response of this system and plot the states.

4. Let us consider the following dynamics system:

$$x(t+1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(a) check the controllability and the observability of this system
(b) establish the controllable and the observable canonical forms
(c) assume that we have complete access to the state vector, design a state feedback controller that place the poles of the closed-loop dynamics at 0.01 and $0.1 \pm 0.1j$

(d) design simultaneously the controller and the observer that give the same poles for the closed-loop dynamics.

(e) write a Matlab program that simulate the step response of this system and plot the states.

5. Let us consider the following dynamics system:

$$x(t + 1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

with:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -3 & -2 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 2 & 1 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0.1 & 1 \end{bmatrix}$$

(a) check the controllability and the observability of this system

(b) establish the controllable and the observable canonical forms

(c) assume that we have complete access to the state vector, design a state feedback controller that place the poles of the closed-loop dynamics at 0.01 and $0.1 \pm 0.1 j$

(d) design simultaneously the controller and the observer that give the same poles for the closed-loop dynamics.

(e) write a Matlab program that simulate the step response of this system and plot the states.

(f) choose the appropriate matrices Q and R that we can use to solve the LQR regulator problem and study the effect of each matrix on the performances.

(g) for a given choice of couple matrices Q and R, determine the time response of the corresponding gain when the initial conditions are not equal to zero.

Part V

Implementation

8

Design and Implementation of Mechatronic System

After reading this chapter the reader will:

1. master the design and real time implementation of mechatronic systems

2. be able to perform the different phases of the design of mechatronic system

3. be able to solve the control problem and establish the control law that we have to implement in real time

4. be able to write programs in C language using the interrupt concept for real time implementation

8.1 Introduction

In the last chapters we developed theoretical concepts and it is now time to pass to action and show how these concepts apply for real systems and how we can implement them in real time. This chapter deals with the design and the real-time implementation of algorithms for mechatronic systems. The design process consists

of creating the physical mechatronic system and once this system is designed the next step consists of implementing the intelligence we want to give the system. The two phases are not independent and care should be paid during the execution of these two phases. In either the design or the implementation, specifications must be stated first and the request task is done in order to satisfy these restrictions.

The design consists in some sense of creating the mechatronic system with all its components, mechanical part, sensors, actuators, electronic circuit, etc. The mechanical part can either be manufactured in house or assembled from existing parts in the marketplace, meanwhile the other components, actuators, sensors, electronic parts are selected with precaution and assembled in an electronic circuit that will be the brain of the mechatronic system once the control algorithm is implemented.

The implementation consists of building a real-time control system which requires itself two stages that are the controller design and its digital implementation. At controller design stage, some specifications are firstly formulated and a controller that can satisfy these performances is designed. The controller can be designed using one of the methods developed earlier. At the implementation phase, the recurrent equation of the controller is implemented in real-time. Care should be paid during the implementation to minimize computation errors and delays that may cause instability of the whole system. The implementation is mainly based on interrupts and uses information from the used sensors to generate the actions that the microcontroller should send to the different actuators.

In the rest of this chapter we will cover these concepts and give an idea to the reader on how these concepts work in practice. In Section 2, the design phase is developed and some design techniques are presented. Section 3 covers the electronic design. In Section 4, the software design and real-time implementation are tackled. Section 5 treats the design and implementation based on the transfer function while the Section 6 covers the one based on state space representation. Numerical examples are used in all the chapter to give an idea to the reader.

8.2 Design Phase

As we said earlier the design is the philosophy by which the mechatronic system is created. This phase starts in general from a vague idea that can be improved to produce the desired system. The design can start by a desire to build a system that can perform a task. In general, the design phase is done by a group of persons that own some experience in the field. The success of the project requires that the group of persons follows a certain number of steps to attain the goal. Most of the steps that are mostly used are:

- define the project and its planning
- identify customers and their needs

- evaluate existing similar products if there exist

- generate the engineering specifications and target values

- perform conceptual design

- perform concept evaluations

- develop product/prototype

- evaluate product for performance and cost

For more details on these steps we recommend the reader to consult specialized references on the subject.

More specifically the design phase itself is divided in the following steps:

- understanding of the design problem

- decomposition of the design problem

- generation of solutions

- analysis of the chosen solutions

- practicability of the solutions

The brainstorming technique is a crucial step in the success of the project and it is consists of an unstructured way of generating ideas by the chosen group of persons working in the group and those invited for this purpose. These persons are selected for their knowledge and creativity to seek solution to design problems. During the brainstorming session, a leader to direct the discussion and a secretary to collect the generated ideas are nominated. Before the brainstorming session starts, the leader may want to brief the members of the group on what they are brainstorming about. He has also to remind them of the goal of the project, factors that affect the project, resources available, and constraints they face, etc. During all the brainstorming session, the members of the group are encouraged to generate solutions and all the ideas are accepted. It is important that during the brainstorming step, the problem is identified and the goals and the objectives of the problem are characterized. Then, the leader has to encourage everyone in the group to participate actively in generating ideas, and to keep the ideas flowing as long as possible by restating the problem and pushing the members to be more active. Sometimes, by going through the collected ideas and reviewing them can help to generate new ones. After suspension of the session, the leader invites the members of the group to rank the generated ideas using some acceptability criteria. The members are also invited to select the top ideas for further analysis and to prepare a final report that should contain more details such as diagrams, tables, charts, etc.

In the design phase, we should also concentrate on the design of the electronic circuit of the system we would like to build. Once this is done, we pass to the design of the algorithms we would like to implement in the system. This phase as we said earlier is done in two steps. The first one fixes the performances we would like to give the system and then we try to design the controller that can assure

such requirements. Once this is done and tested by simulation for performances and efficiency, the second phase starts and it consists of implementing the designed algorithms to work in real-time. More often interrupts are used for this purpose and code optimization is done to satisfy the requirements of real time implementation.

As an example of idea for mechatronic systems, we mention the one that gave the Segway. This system beside the challenges that pose for researchers in mechatronic laboratories, can be used as a transportation system in places like airports or downtowns where cars can not be used. If we take this example and show how the design steps apply for this system. The main goal for this system is to transport human being in a given environment. As constraints, the weight of the system and its autonomy are of great importance. Also, the weight of the transported person is also of great importance and should be taken during the design phase. Vaguely, the design problem can be formulated as we desire to design a machine with two wheels that can transport human being safely indoor and outdoor. The idea of this mechatronic system is the same as the one of the two wheels robot developed earlier. The design of such machine should be ergonomic with an attractive shape and minimum cost to increase the competition. It is also important that during the design phase to think about the environment and solve the recycling problem of the different components of the Segway.

The whole vehicle can be divided into the following components:

- mechanical part

- choice of the sensors

- choice of the actuators

- electronic circuit

- batteries selection

The mechanical part comprises of:

- the platform on which the person to be transported should stand,

- the two wheels that are solidly attached to the platform

- the handlebar which is used to help the rider to turn either left or right

These parts are designed in the same manner as for the two wheels robot except in this case, we are transporting human beings and more care should paid for security and we should eliminate all the causes of injuries that may result from malfunctioning of any part of this system.

Once the mechanical part is designed, intelligence must be implemented to make the system works as it is desired. The principle of how the Segway works is in some sense based of the human being balancing. To have an idea on this, imagine that a person stands up and leans forward or backward far from the vertical position, as a consequence, he will be out of balance, and probably he will not fall on his face or his back since the brain of this person knows that he is out of balance based on a complicated mechanism that human beings have built since his young age,

therefore, his brain will order him to put his leg forward or backward to stop the fall. If the person keeps leaning forward for instance, his brain will keep putting his legs forward to keep him upright. Instead of falling, he walks forward, one step at a time. From this we can see that the Segway works following the same principle as human being, except it has wheels instead of two legs, a set of actuators instead of muscles, a collection of microcontrollers instead of a brain and a set of sophisticated tilt sensors (accelerometers, gyroscopes, etc.) instead of an inner-ear balancing system. If you are taking a ride on this machine, like the human brain, the Segway knows when you are leaning forward by measuring the angle you are making with respect to the vertical axis and to maintain you in balance, it turns the wheels at the just time with the right speed, so you move forward.

The brain of the Segway is the electronic circuit that is built around one or many microcontrollers that run the intelligence we implement for this system. Based on information from the different sensors, controls are computed and transmitted to the different actuators at the just moments to take the necessary actions. To work properly, the Segway must be equipped with batteries that give the required energy. These batteries must provide more autonomy and be included in the design to make the shape of the whole system attractive and safe. Fig. 8.1 gives an idea on the electronic circuit of the mobile robot.
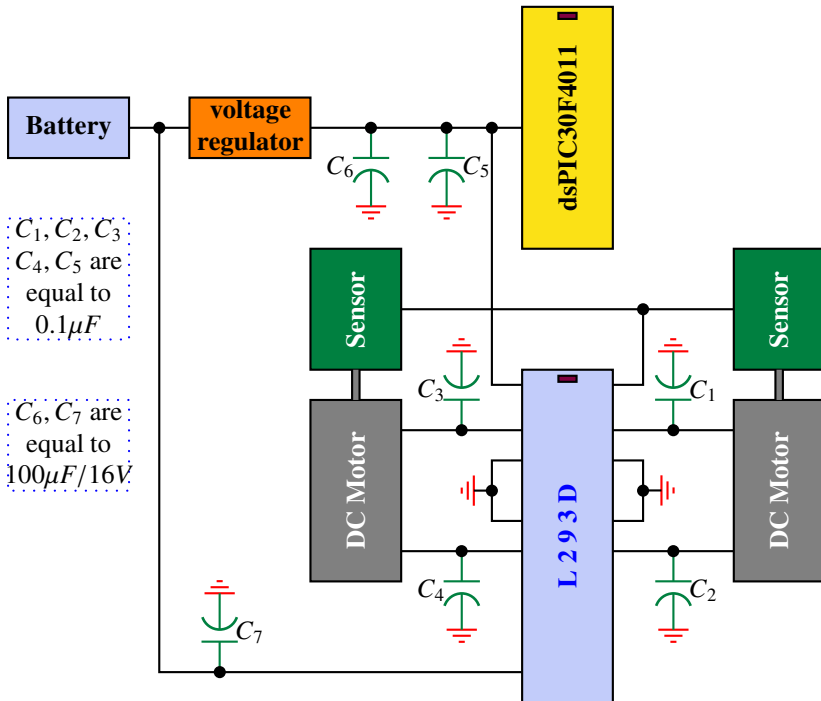

Fig. 8.1 Two wheels robot

8.3   Electronic Design

The electronic design consists in some sense to built the electronic circuit around the dsPIC30F4011. The electronic components like resistors, capacitors and integrated circuits must be appropriately chosen to guarantee a good precision in the real-time implementation. Protection should be implemented to protect either the electronic components and the users. When the voltage can harm the users, security rules have to be followed rigorously. Since most of the mechatronic systems combine analog and digital signals and the conversion from one type to another. In case of converting analog signal to digital one, a analog-digital converter (ADC) is used and for precision purpose, a 10 bits one can be used, but if the budget of the project allows that, a 16 bits can be used instead. From digital to analog, we will use in general PWM otherwise the digital-analog converter (DAC) can be used. To guarantee the functioning of the whole system energy is needed and most of the case since the mechatronic systems are designed to be autonomous, batteries are used for this purpose

The dc motor kit or the two wheels robot are examples that we will cover in more detail later in this book and where all these concepts are treated.

8.4   Software Design and Real-Time Implementation

The software design consists of building the intelligence we should give to our mechatronic system. It is this part that will constitute the brain of the system. Here we will develop the code that makes the dsPIC30F4011 interacts with all the electronic components and makes the system executes the task for which it was designed. This part in our case is written in C and uses mainly the interrupts that the dsPIC30F4011 offers. It also uses PWM to deliver the appropriate voltage to the different actuators used in the examples we are treating in this volume.

To make this volume self contained and before giving the structure we will use in this volume, let us present the dsPIC30F4011 and give some examples of how to use the PWM and the interrupts and how to program such functions.

8.4.1   dsPIC30F4011

The dsPIC30F4011 is a 16 bits microcontroller produced by Microchip. It is a dsPIC digital signal controller and it is capable of performing very fast mathematical operations. Referring to its datasheet we can notice the following features that our microcontroller has:

- a processing speed that can reach a maximum of 30 MIPS (which means that the internal clock runs at 120 MHz)

- a modified Harvard architecture that requires four clock ticks to increment the Instruction pointer (IP) like almost all the microcontrollers of Microchip

- 48 Kbytes Flash Memory

- 1 Kbytes EEPROM

- a 2048 bytes SRAM

- an internal oscillator of 7.37 MHz, 512 KHz

- a supply voltage that ranges from 2.5 V up to 5 V (need to be regulated voltage)

- nine pins are connected to the 10 bits analog to digital converter (ADC) module, working at 500 Ksps (Kilo samples per second)

- one quadrature encoder interface (QEI)

- an instructions register (IR) with 24 bits, where 16 bits are used for the data, and 8 bits for commands

- thirty interrupt sources

- thirty I/O (General Input/Output) pins capable of sourcing or sinking up to 25 mA each

- five 16 bits timers (5×16 bits) that may be paired into 32 bits timers (2×32 bits) if it is necessary

- two UART (Universal Asynchronous Receiver Transmitter) that can be used for for communications, one SPI (Serial to Peripheral Interface), one I2C (Inter Integrated Circuit), and one CAN (Controller Area Network).

- 4 standard PWM and six motor control PWM (Pulse Width Modulation) channels

The dsPIC30F4011 controller we will use in this volume is a 40 pins PDIP. It comes also in other forms but we will not use these forms. For more details on the other forms, we refer the reader to the datasheet of this dsPIC that can be found in the webpage of Microchip (www.mircichip.com). It is also important to mention that using the dsPIC30F4011 at high frequency may produce heat and can cause its damage. Fig. 8.2 gives an idea of the pins description.

In general the following structure is used when programming the dsPIC30F4011:

```
//
// Put here title and comments
//
#include "p30F4011.h"        // proc specific header
```

1	MCLR	AVDD	40
2	EMUD3/AN0/VREF+/CN2/RB0	AVSS	39
3	EMUC3/AN1/VREF-/CN3/RB1	PWM1L/RE0	38
4	AN2/SS1/CN4/RB2	PWM1H/RE1	37
5	AN3/INDX/CN5/RB3	PWM2L/RE2	36
6	AN4/QEA/IC7/CN6/RB4	PWM2H/RE3	35
7	AN5/QEB/IC8/CN7/RB5	PWM3L/RE4	34
8	AN6/OCFA/RB6	PWM3H/RE5	33
9	AN7/RB7	VDD	32
10	AN8/RB8	VSS	31
11	VDD	C1RX/RF0	30
12	VSS	C1TX/RF1	29
13	OSC1/CLKIN	U2RX/CN17/RF4	28
14	OSC2/CLKo/RC15	U2TX/CN18/RF5	27
15	EMUD1/SOSCI/T2CK/U1ATX/CN1/RC13	PGC/EMUC/U1RX/SDI1/SDA/RF2	26
16	EMUC1/SOSC0/T1CK/U1ARX/CN0/RC14	PGD/EMUD/U1TX/SD01/SCL/RF3	25
17	FLA/INT0/RF8	SCK1/RF6	24
18	EMUD2/OC2/IC2/INT2/RD1	EMUC2/OC1/IC1/INT1/RD0	23
19	OC4/RD3	OC3/RD2	22
20	VSS	VDD	21

Fig. 8.2 dsPIC30F4011 pins description

```
//
// Define gobal variables in RAM
//
float  Reference;   // simple variable
int variable0;      // (16 bits)
char myVariable; // (8 bits)

#define n1 10   /* sample constant definition */
#define n2 20;
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
            // array with dsPIC30F attributes
int array5[n2];    // simple array
int variable3 __attribute__((__space__(xmemory)));
            // variable with attributes
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
            // array with dsPIC30F attributes
int array5[n2];     // simple array
int variable4 __attribute__((__space__(xmemory)));
            // variable with attributes
```

```
 //
 // Define a struct
 //
  typedef struct {
       // PID Gains
       float KP;       // Propotional gain
       float KI;        // Integral gain
       float KD;       // Derivative gain

    //
    // PID Constants
    //
    float Const1_pid;  // KP + T KI + KD/T
    float Const2_pid;  // KP + 2KD/T
    float Const3_pid;  // Kd/T
    float Const4_pid;  // KP + KD/T
    float Const5_pid;  // T KI

    //
    // System variables
    //
    float y_c;      // y_c[k] -> controlled output
    float y_m;     // y_m[k] -> measured output
    float u_k;      // u[k]   -> control at time k
    float e_k;      // e[k]   -> error at time k

    //
    // System past variables
    //
    float u_km1;      // u[k-1] -> output at time k-1
    float e_km1;      // e[k-1] -> error at time k-1
    float e_km2;      // e[k-2] -> error at time k-2
    float y_mkm1;  // y_m[k-1] -> measured output at time k-1
    float y_mkm2;  // y_m[k-2] -> measured output at time k-2

  }PIDStruct;

  PIDStruct thePID;

//
// Constants in ROM
//
const char Variable_Rom[] = {1,2,3,4};
const int myConstant = 100;

//
// Non memorized constants
//
#define var1 0x1234;
```

```
#define var2 "ma chaine";

//
// Functions
//
float my_Function(float a, float b)
    {
        int local_var;

        local_var = a - b;
        return local_var;
    }



//
// Interrupt program here using Timer 1 (overflow of counter Timer 1)
//
void __ISR _T1Interrupt(void)      // interrupt routine code
  {
        // Interrupt Service Routine code goes here
        float Position_error;

        // get the actual position from the encoder
        // ThePID.y_m

        Position_error = my_Function(Reference, ThePID.y_m);
        .......

        IFS0bits.T1IF=0;    // Disable the interrupt
  }


int main ( void )               // start of main application code
{
// Application code goes here
int i;

// Initialize the variables Reference and ThePID.y_m
(it can be read from inputs) Reference =  0x8000; // Hexadecimal number
(0b...  Binary number) ThePID = 0x8000;

// Initialize the registers
TRISC=0x9fff; // RC13 and RC14 (pins 15 and 16) are configured as outputs
IEC0bits.T1IE=1;   // Enable the interrupt on Timer 1

// Infinite loop
while (1)
{
}
```

```
return 0
}
```

This structure is a starting point and we will improve it in the coming pages to make it a good program that can serve as example for future development. Since the PWM technique and interrupts are mostly used in the real-time implementation, let us now present these two concepts to make the reader familiar with them.

8.4.2 Pusle Width Modulation

The pulse width modulation (PWM) is a modulation technique that generates variable-width pulses to represent the amplitude of an analog input signal. It is also seen as a method that uses width to encode or modulate an analog signal. In PWM technique, by switching the delivered voltage to the actuator for instance with the appropriate duty cycle (the fraction between the time when the input is on and the period of the signal), the output will approximate a voltage at the desired level we would like to deliver to this actuator. It is important to keep in mind that maximum voltage of the delivered voltage is always the same and we only act on the duration when the voltage is at its maximum. Mathematically, if we consider an analog signal, $e(t)$ that varies between two bounds, $\bar{e}$ and $\underline{e}$, as shown in Fig. 8.3, the average value of this input is given by:
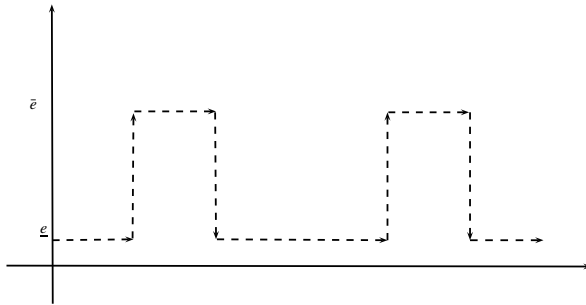
$$v = \frac{1}{T} \int_0^T e(t)dt$$

Fig. 8.3 Example of PWM signal

Since the input signal, $e(t)$, is a square signal that takes the following values:

$$e(t) = \begin{cases} \bar{e} & 0 \leq t \leq T_1 \\ \underline{e} & T_1 \leq t \leq T \end{cases}$$

we have:

$$v = \frac{1}{T}\left[\int_0^{T_1} e(t)dt + \int_{T_1}^{T} e(t)dt\right]$$
$$= \frac{1}{T}\left[\bar{e}T_1 + \underline{e}(T - T_1)\right]$$

Now if we define, the ratio between the time during which the signal is at its maximum value and the one when the signal is at its minimum as the duty cycle and we denote it by *dutyCycle* and noting that:

$$T = T_1 + T_2$$

with $T_2 - T_1$ is the interval of time when the signal is equal to $\underline{e}$, we have:

$$dutyCycle = \frac{T_1}{T}$$

The average value of the signal becomes then:

$$v = \frac{1}{T}\left[\bar{e}\ dutyCycleT + \underline{e}\ (T - dutyCycleT)\right]$$
$$= \bar{e}\ dutyCycle + \underline{e}\ (1 - dutyCycle)$$

As an example, let $\underline{e} = 0$, $\bar{e} = V$ and the analog signal is 75% of the period is on and the rest of time is off. If the amplitude of the analog signal is V, mathematically we have:

$$u = \int_0^T e(t)dt = \int_0^{T_1} e(t)dt + \int_{T_1}^{T_2} e(t)dt + (T - T_1)V$$

Since $T_2 - T_1 = 0.75T$ where T is the period of the analog signal, we have:

$$u = 0.75V$$

Based on this discussion, we conclude that wider pulses will give higher average voltage. Therefore, the maximum voltage is obtained when the duty cycle is 100% and the zero voltage is obtained when the duty cycle is 0 %. It is also important to notice that the choice of the period (frequency) of the square signal is very important. In fact, in case of dc motor control, a choice of a very low frequency will make the motor jerks, however the choice of a too high frequency can reduce the efficiency.

The PWM value that we can send to the chosen device is computed as follows:

$$PWM = dutyCycleV$$

where V is the maximum voltage of the square signal with a period T.

Controlling the voltage to send to the dc motor is brought to the control of the *dutyCycle* for a given period, T, of the square signal.

The PWM technique can be used to digitally create:

- analog output voltage level for control functions and power supplies;
- analog signals for arbitrary waveforms, music, speech and sounds

In this book we will mainly use this technique to deliver the required voltage to drive the desired actuator through the L293D.

To see how the PWM is implemented in real-time applications, let us consider the position control of a dc motor.

```c
#include "p30f4011.h"
#include "pwm.h"

#define ENABLETRIS          TRISEbits.TRISE2
#define ENABLE              LATEbits.LATE2

#define ENCODER_PRIORITY       7
#define CONTROLLER_PRIORITY    5
#define DISPLAY_PRIORITY       2

//Discrete-time PID:

 typedef struct {
       // PID Gains
       float KP;        // Propotional gain
       float KI;         // Integral gain
       float KD;       // Derivative gain

    //
    // PID Constants
    //
    float Const1_pid; // KP + T KI + KD/T
    float Const2_pid; // KP + 2KD/T
    float Const3_pid; // Kd/T
    float Const4_pid; // KP + KD/T
    float Const5_pid; // T KI

    //
    // System variables
    //
    float y_c;       // y_c[k] -> y commanded
    float y_m;       // y_m[k] -> y measured
    float u_k;       // u[k]   -> output at time k
    float e_k;       // e[k]   -> error at time k

    //
    // System past variables
    //
    float u_km1;      // u[k-1] -> output at time k-1
    float e_km1;      // e[k-1] -> error at time k-1
    float e_km2;      // e[k-2] -> error at time k-2
    float y_mkm1;  // y_m[k-1] -> y measured at time k-1
    float y_mkm2;  // y_m[k-2] -> y measured at time k-2
```

```c
   }PIDStruct;

   PIDStruct thePID;

#define   Ka    70
#define   Kb    -129
#define   Kc    59

#define   Ts    0.005; // 1.0/200;
#define   Fs    200.0;

//
// dsPIC4011configuration
//
_FOSC(CSW_FSCM_OFF & FRC_PLL16);// Primary oscillator = Fast RC @ PLLx16
_FWDT(WDT_OFF);                                      // Watch dog off
_FBORPOR(PBOR_OFF & MCLR_DIS);  // Enable MCLR reset pin and tun off
 the power-up timers
_FGS(CODE_PROT_OFF);  // Disable code protection
_FICD( ICS_NONE );      //


//
// Global variables
//
typedef enum _BOOL { FALSE = 0, TRUE } BOOL;

BOOL    A;
BOOL    B;
BOOL prevA;
BOOL prevB;

//
// PID variables
//
long Pos;
long e[3];
long ref;

double u[2];

unsigned int dutyCycle;


//
// Functions
//
void InitFunction(void);      // InitFunction
```

```
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void);
// Interrupt function
void __attribute__((__interrupt__)) _T1Interrupt(void);
// Interrupt function


//
// Main function
//
int main(void)
{
    init();

    ref = 600; // (90 deg) since we have 2400 pusles/revolution

    while(1)
    {
    }
    return 0
}



//
// CN interrupt to get the position of the dc motor
//
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)
{
    if(IFS0bits.CNIF)
    {
      CNLED = !CNLED;

      // Get current Encoder signals
      // Must read port before clearing flag!!
      A = PORTBbits.RB2;
      B = PORTBbits.RB3;

      // Compare current signals with previous signals to see which
         channel changed
      // Change on A
      if(A != prevA){
        if(A == B){
          Pos++;
        }else{
         Pos--;
        }
      // Change on B
      }else if(B != prevB){
        if(A == B){
          Pos--;
```

```
        }else{
          Pos++;
        }
      }

      // Save current signals for next time
      prevA = A;
      prevB = B;

      IFS0bits.CNIF=0;
    }

}//end of CN_interrupt function


//
// Interrupt function that goes with Timer 1 (to fix the sampling
   period)
//
void __attribute__((__interrupt__)) _T1Interrupt(void)
{
    if (IFS0bits.T1IF)
    {
        //
        // Compute the error
        //
        e[0] = ref - Pos;

        //
        // Compute the control action
        //
        u[0] = u[1] + Ka*e[0] + Kb*e[1] + Kc*e[2];

        //
        // Send the control action
        //
        SetDCMCPWM(1, 1024 + (int)(u[0]), 0);

        //
        // Keep the actual error and control action for next compuation
        //
        u[1] = u[0];
        e[2] = e[1];
        e[1] = e[0];

        IFS0bits.T1IF = 0;    // Clear timer interrupt flag
    }
}
```

```
//
// InitFunction
//
void InitFunction(void)
{
    //
    // Initilize variables
    //
    u[0] = 0.0;
        u[1] = 0.0;
    e[0] = 0;
    e[1] = 0;
    e[2] = 0;

    //
    // Activate the interrupt priorities
    //
    INTCON1bits.NSTDIS = 0;

    //
    // Initialize the AD
    //
    ADPCFG = 0b11111111;

    //
    // Configure the I/O
    //
    TRISEbits.TRISE0 = 0;   // PWM1H
    TRISEbits.TRISE1 = 0;   // PWM1L
        TRISBbits.TRISB2 = 1; // Channel A of the eoncoder: RB2 (pin 2)
        TRISBbits.TRISB3 = 1; // Channel B of the encoder: RB3 (pin 5)
     ENABLETRIS = 0;

    //
    // initialize the encoder variables
    //
    prevA = PORTBbits.RB2;
    prevB = PORTBbits.RB3;

    //
    // CN interrupt
    //
    CNEN1bits.CN0IE=0;          // CN0 interrupt disable
    CNEN1bits.CN1IE=0;          // CN1 interrupt disable
    CNEN1bits.CN2IE=0;          // CN2 interrupt ENABLE
    CNEN1bits.CN3IE=0;          // CN3 interrupt ENABLE
    CNEN1bits.CN4IE=1;          // CN4 interrupt disable
    CNEN1bits.CN5IE=1;          // CN5 interrupt disable
    CNEN1bits.CN6IE=0;          // CN6 interrupt disable
```

```
    CNEN1bits.CN7IE=0;          // CN7 interrupt disable
    CNEN2bits.CN17IE=0;         // CN17 interrupt disable
    CNEN2bits.CN18IE=0;         // CN18 interrupt disable

    IFS0bits.CNIF = 0;          // clear CN interrupt flag
    IPC3bits.CNIP = ENCODER_PRIORITY; // CN interrupt max priority (7)
    IEC0bits.CNIE = 1;          // CN interrupt enable

    //
    // Configure PWM
        //
    ConfigIntMCPWM(PWM_INT_DIS & PWM_FLTA_DIS_INT);

    SetDCMCPWM(1, 1024, 0);

    OpenMCPWM (0x3FF, 0x0, PWM_EN
    & PWM_IDLE_CON & PWM_OP_SCALE1 & PWM_IPCLK_SCALE1 & PWM_MOD_FREE,
    PWM_MOD1_COMP & PWM_PDIS3H & PWM_PDIS2H & PWM_PEN1H & PWM_PDIS3L
    & PWM_PDIS2L & PWM_PEN1L, PWM_SEVOPS1 & PWM_OSYNC_TCY & PWM_UEN);


    //
    // Timer 1 interrupt
    //
    T1CONbits.TON=1;            // turn timer 1 on
    T1CONbits.TGATE=0;
    T1CONbits.TSIDL=0;          // stop timer in idle mode (0=non)
    T1CONbits.TCKPS=1;          // prescaler (0=1:1, 1=1:8, 2=1:64
    T1CONbits.TCS=0;            // clock source (0=FOSC/4)
    PR1 = 18424;                // 200Hz
    IFS0bits.T1IF = 0;          // clear timer 1 interrupt flag
    IPC0bits.T1IP = CONTROLLER_PRIORITY;
    IEC0bits.T1IE=1;            // enable timer 1 interrupt
}
```

When the voltage to send to a device varies between $\bar{u}$ and $-\bar{u}$, which corresponds to:

$$u(t) = \begin{cases} \bar{u} & \text{DutyCycle} = 100\ \% \\ 0 & \text{DutyCycle} = 50\ \% \\ -\bar{u} & \text{DutyCycle} = 0\ \% \end{cases}$$

The evolution of the voltage is given by:

$$DutyCycle = 50 + 50\frac{u(t)}{\bar{u}}$$

The following function can be used to initiate the dsPIC PWM function as an example.

```
void init_PWM (void){
     Val_reg = 1023;              // PWM signal frequency : 30000 Hz
     PTCONbits.PTEN = 1;       // Activation of the time base
     PTCONbits.PTSIDL = 1;    // Configuration in Idle Mode
     PTCONbits.PTCKPS = 0;  // Prescaler equal to 1
     PTCONbits.PTMOD = 0;     // Slection of the free running mode
     PTMRbits.PTDIR = 0;                 // Increment of the time base
     PTMRbits.PTMR = Val_reg;    // Valeur du Registre du Time base
     PTPER = Val_reg;              // Valeur de la priode du signal
     PWMCON1bits.PMOD1 = 0;  // Slection du mode PWM complmentaire
     PWMCON1bits.PEN1H = 1;  // Activation des pins en mode PWM
     PWMCON1bits.PEN1L = 1;  // Activation des pins en mode PWM
     DTCON1bits.DTAPS = 0;     // Time base unit is 1TCY
     DTCON1bits.DTA = 0;            // Valeur du DT pour l'unit A
     PDC1 = 0;   // Mise  zro du registre pour la largeur dimpulsion
}
```

8.4.3 Interrupts

Interrupts play an important role in real-time control, it is why we reserve a whole section to this concept in order to make it easy for the reader to follow our implementations. If we look to the dictionary for the word "interrupt", we found that this means "stop the continuous progress of an activity. From our perspective, a interrupt can be defined as a signal emanating from a device that stops immediately the running program and forces the execution of another one, i.e: the one of the interrupt. This is done without waiting for the running program to finish. Once the interrupt service routine (ISR) finishes, the program execution returns to the halted program and continues exactly from where it was interrupted when the interrupt occurs.

Using hardware interrupts has some advantages among them we quote that:

- the processor is used properly

- the processor can use a wake-from-sleep interrupt which allows the processor when the interrupt is active to go into a lower mode to save energy

It is recommended to keep the interrupt function small compared to the processing time of the other functions in the program of the application you are working on.

The following listing gives a templates for an interrupt:

```
#include "p30F4011.h"         /* proc specific header */
#define CONSTANT1 10        /* sample constant definition */
int array1[CONSTANT1] __attribute__((__space__(xmemory), __aligned__(32)));
                           /* array with dsPIC30F attributes */
int array5[CONSTANT2];            /* simple array */
int variable1 __attribute__((__space__(xmemory)));
                             /* variable with attributes */
```

```
int variable3;        /* simple variable */
int main ( void )   /* start of main application code */
  {
  /* Application code goes here */
  }
void __attribute__((__interrupt__(__save__(variable1,variable2)))) _INT0Interrupt(void)
               /* interrupt routine code */
  {
  /* Interrupt Service Routine code goes here  */
  }
```

The structure of the code should be as follows:

```
#include "p30F4011.h"  // proc specific header

#include " myData.c" // here goes all the declaration of global variables
#include "InitFunction.c" // here goes all the initialization function
#include "timers.c"        //  here goes the code for the used timers
#include "usefulFunction.c"  // here goes other useful functions
#include "interrupts.c" // here goes the codes for all the interrupts of the system

int main ( void )          // start of main application code
  {
       //  call to the different functions
       myData();
       InitFunctions();
       while(1);
  }
```

The function *main* starts by initializing the global variables we us use in the program. It also initialize the timers for interrupts.

The structure of the implementation includes:

- the files headers. It always starts with the inclusion of "p30F4011.h"

- the file myData.c that contains all the declaration of the useful variables that we need in the program

- the file InitFunction.c that allows the initialization of the dsPIC30F4011 and its inputs/outputs. In this stage it is important to introduce a certain delay to allow that the voltage of the dsPIC30F4011 is stabilized to avoid surprises

- the file usefulFunction.c that gives the other function that we may use like functions for LCD etc.

- the file interrupts.c that give the body of the used interrupts and link to the timers. The body of the interrupts should simple and short.

- finally, we have the main.c function that calls all the previous functions in logical ways. After the declaration phase and the initialization process, we enter in an infinite loop that only the interrupts can halt to execute the desired tasks.

As a simple exmaple to this structure let us consider the the control of the speed of a mechanical part that is driven by a dc motor via gearbox with the characteristics are we presented earlier. For this purpose we connect the motor as follows:

- the pin 1 of the L293D is connected to Enable

- the pins 2 and 7 of the L293D are connected respectively to the pins A and B of the dsPIC30F4011

- the pin 3 of the L293D is connected to the positive connector of the dc motor

- the pin 6 of the L293D is connected to the negative connector of the dc motor

- the pins 4, 5, 12 and 13 of the L293D are connected to the ground

- the pins 8 and 16 of the L293D are connected respectively to 12V and 5V

The following convention is used:

Table 8.1 Convention for dc motor movement

A	B	Description
0	0	the dc motor is sptopped
0	1	the dc motor turns anticlockwise
1	0	the dc motor turns clockwise
1	1	the dc motor is stopped

```
#include "p30F4011.h"

// myData.c
#define IntMax 65535          // maximum unsigned integer limit
#define L293D_A P2_0        // Positive of motor
#define L293D_B P2_1        // Negative of motor
#define L293D_E P2_2        // Enable of L293D
#define nop2() {nop();nop();} // define a macro function

unsigned int i;             // local variable
#define myDelay() {for(i=0;i<IntMax;i++) nop2();} // first delay macro

// usefulFunctions.c
```

```
void rotateAntiClockwise(void);      // turn AntiClockwise
void rotateClockwise(void);             // turn Clockwise
void break(void);                                // Stop the dc motor

void main()                    // main function
     {
        while(1)
        {
          rotateAntiClockwise();    // turn the dc motor AntiClockwise
          delay();                                 // introduce a delay
          break();                                 // break the dc motor
          MyDelay();              //Some delay

          // change the direction of rotation of the dc motor
          rotateClockwise();          // turn the dc motor Clockwise
          delay();                               // introduc a delay
          break();                               // break the dc motor
          MyDelay();                               //Some delay
        }
}

void rotateAntiClockwise()
     {
        L293D_A = 1;            //Make positive of motor 1
        L293D_B = 0;            //Make negative of motor 0
        L293D_E = 1;            //Enable L293D
}

void rotateClockwise()
      {
        L293D_A = 0;            //Make positive of motor 0
        L293D_B = 1;            //Make negative of motor 1
        L293D_E = 1;            //Enable L293D
}

void break()
     {
        L293D_A = 0;            //Make positive of motor 0
        L293D_B = 0;            //Make negative of motor 0
        L293D_E = 0;            //Disable L293D
}
```

8.5 Design and Implementation Based of Transfer Function

This section covers the software design and its implementation in real-time using the transfer function model. For this purpose, we assume that the mechanical and the electronic designs are already done and the model of the whole mechatronic system is known. As it was said earlier, PID controller or equivalent controller can be used to reach the specifications of the control design. We have already presented techniques that can be used to compute the controller parameters. Depending on the method we use, the result is always the same and it consists of establishing the recurrent equation for the control law that has to be implemented in real-time. More specifically, if we denote the transfer function of the system by $G(z)$ and the one of the controller by $C(z)$. The block diagram in Fig. 8.4 gives the structure of the closed-loop of the system.
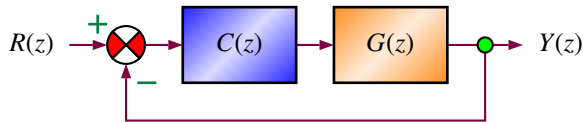

Fig. 8.4 Block diagram of the closed-loop

The transfer function of the controller, $C(z)$ is given by:

$$C(z) = \frac{U(z)}{E(z)}$$

where $E(z)$ is the error and $U(z)$ is the control action that we have to send to the system.

The determination of the the transfer function $C(z)$ is in general done into two steps. The first one determines the structure of the controller that can be chosen from the following list:

- proportional controller (P)
- proportional and integral controller (PI)
- proportional and derivative controller (PD)
- proportional, integral and derivative controller (PID)
- phase lead controller
- phase lag controller
- phase lead-lag controller

Once the structure is fixed in the first step, the second one consists of determining the controller parameters. For instance, if the specifications require a PID controller, we need to compute the three gains:

- K_P (proportional gain)

- K_I (integral gain)

- K_D (derivative gain)

Remark 8.5.1 *When the output signal, y(t), is noisy the derivative action has to be avoided unless an appropriate filter is implemented to make the feedback signal more smooth.*

The implementation in real-time of the PID controller requires a discretization of the following equation of the PID:

$$u(t) \ = \ K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \dot{e}(t)$$

where $e(t)$ is the error of the system at time t.

Based of what it was presented earlier many possibilities can be used to get the discrete-time version of the PID controller. The following ones are some possibilities:

- if we approximate the integral and the derivative by:

$$\int_0^t e(\tau)d\tau \ = \ T \sum_{i=0}^k e(i)$$

$$\dot{e}(t) \ = \ \frac{e(k) - e(k-1)}{T}$$

we get:

$$u(k) = K_P e(k) + T K_I \sum_{i=0}^k e(i) + \frac{K_D}{T}(e(k) - e(k-1))$$

Now if we replace k by $k-1$, we get:

$$u(k-1) = K_P e(k-1) + T K_I \sum_{i=0}^{k-1} e(i) + \frac{K_D}{T}(e(k-1) - e(k-2))$$

Subtracting this equation from the previous one gives:

$$u(k) = u(k-1) + \left(K_P + T K_I + \frac{K_D}{T}\right)e(k) - \left(K_P + 2\frac{K_D}{T}\right)e(k-1) + \frac{K_D}{T}e(k-2)$$

- using backward method

$$C(z) \ = \ \frac{U(z)}{E(z)} = K_P + K_I \frac{Tz}{z-1} + K_D \frac{z-1}{Tz}$$

$$= \ \frac{\left(K_P + T K_I + \frac{K_D}{T}\right)z^2 + \left(-K_P - 2\frac{K_D}{T}\right)z + \left(\frac{K_D}{T}\right)}{z^2 - z}$$

Dividing the numerator and the denominator by z^2 and going back to time, we get:

$$u(k) = u(k-1) + \left(K_P + TK_I + \frac{K_D}{T}\right)e(k) + \left(-K_P - 2\frac{K_D}{T}\right)e(k-1) + \left(\frac{K_D}{T}\right)e(k-2)$$

- using bilinear transformation (trapezoidal method):

$$C(z) = \frac{U(z)}{E(z)} = K_P + K_I\frac{T}{2}\frac{z+1}{z-1} + K_D\frac{2}{T}\frac{z-1}{z+1}$$

$$= \frac{\left(K_P + \frac{TK_I}{2} + \frac{2K_D}{T}\right)z^2 + \left(TK_I - \frac{4K_D}{T}\right)z + \left(-K_P + \frac{TK_I}{2} + \frac{2K_D}{T}\right)}{z^2 - 1}$$

Dividing the numerator and the denominator by z^2 and going back to time, we get:

$$u(k) = u(k-2) + \left(K_P + \frac{TK_I}{2} + \frac{2K_D}{T}\right)e(k) + \left(TK_I - \frac{4K_D}{T}\right)e(k-1)$$

$$+ \left(-K_P + \frac{TK_I}{2} + \frac{2K_D}{T}\right)e(k-2)$$

When the structure and the gains of the controller are fixed using one of the previous expression for the discrete-time PID to approximate the transfer function, $C(z)$, is adopted and the expression converted to a recurrent equation that we should implement in real-time using the interrupts as it was said before.

Remark 8.5.2 *It is important to mention that the PID is the controller that is used in most of the actual industrial control systems that are in service around the world. It is a famous controller that has attracted the interest of many control engineers. We would like also to draw the attention of the reader on the presence of the integral action that may cause windup and that we can prevent during the implementation by using some well known anti-windup techniques.*

Remark 8.5.3 *As we said regarding the schema for the discretization of the controller we can can use here also the trapezoidal schema for the integral action and the backward schema for the derivative one. In this case we get:*

$$u(k) = u(k-1) + ae(k) + be(k-1) + ce(k-2)$$

with $a = K_P + \frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $b = -K_P - 2\frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $c = \frac{K_D}{T_s}$;

To give an idea on the design and implementation based on transfer function, let us consider the position control of a mechanical load driven by a dc motor. As specifications, we would like to guarantee the following:

- an error to a step input equal to zero

- an overshoot less or equal to 5 %

- a settling time at 5 % less or equal to 3s

Following what we presented earlier a proportional controller is enough to respond to the performances. If we let the transfer function of the system and the controller be given by:

$$G(s) = \frac{K_m}{s(\tau_m s + 1)}$$
$$C(s) = K_P$$

where $K_m = 2$ and $\tau_m = 0.5\ s$ are known while K_P is to be determined to satisfy the performances.

The closed-loop transfer function is given by:

$$F(s) = \frac{\frac{K_P K_m}{\tau_m}}{s^2 + \frac{1}{\tau_m}s + \frac{K_P K_m}{\tau_m}}$$

From the specifications, we have:

$$\zeta = 0.707$$
$$t_s = \frac{3}{\zeta\omega_n} = 3\ s$$

which gives:

$$\omega_n = 1.4144\ rad/s$$

The corresponding poles of the desired closed-loop characteristic equation are given by:

$$s_{1,2} = -\zeta\omega_n \pm j\omega_n \sqrt{\zeta^2 - 1}$$
$$= -1 \pm 1j$$

The loci of the system is illustrated in Fig. 8.5. Using the fact we would like the closed-loop transfer to own a ζ equal to 0.707, we get:

$$K_P = 0.5$$

With this gain, if we simulate the closed-loop dynamics we get the behavior of Fig. 8.6 which shows that the behavior is the one we would like to have.

For the implementation in real-time, we obtain different results. The cause of this discrepancy comes from the neglected nonlinear dynamics in the systems. One of these nonlinearities is the backlash in the used gears. To improve the closed-loop behavior we use a proportional and derivative controller. If we let the transfer function of the controller be given by:

$$C(s) = K_P + K_D s$$

the closed-loop transfer function is given by:

$$F(s) = \frac{\frac{K_m K_D}{\tau_m}\left(s + \frac{K_P}{K_D}\right)}{s^2 + \frac{(1 + K_m K_D)}{\tau_m}s + \frac{K_m K_P}{\tau_m}}$$
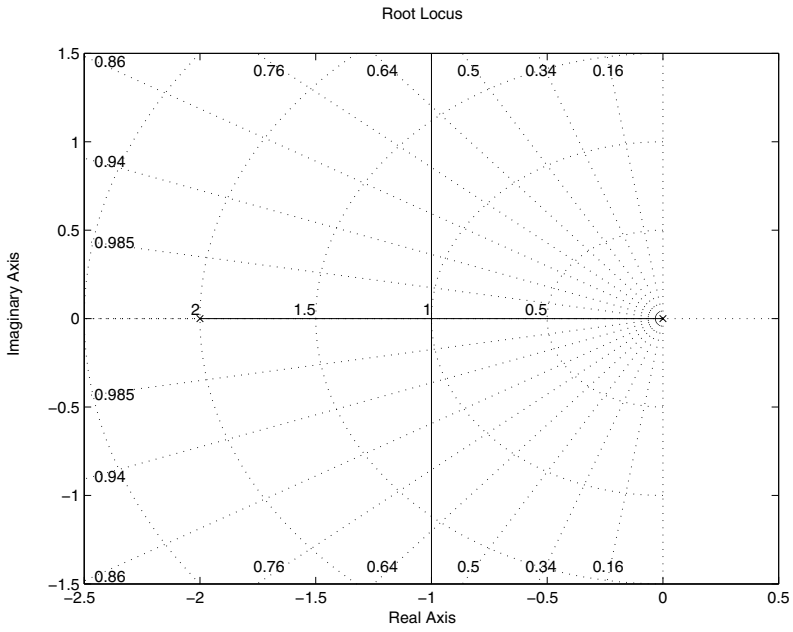
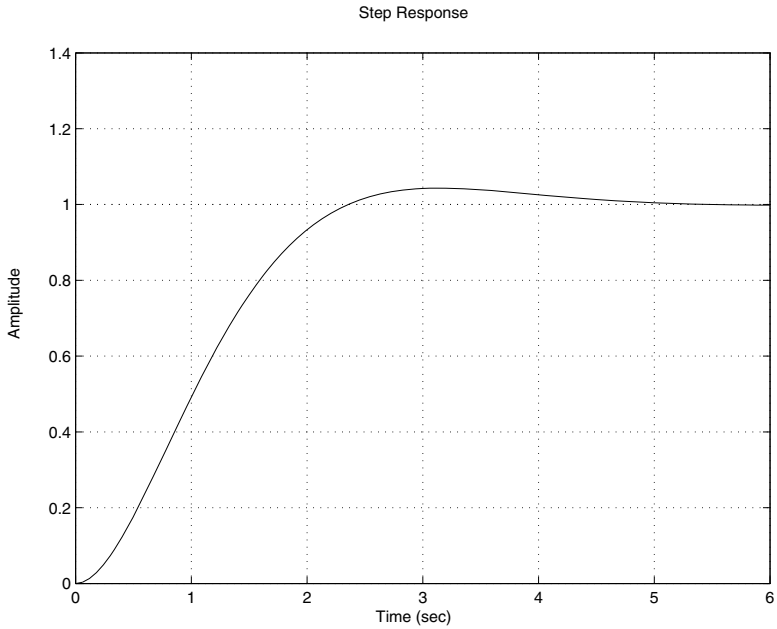**Fig. 8.5** Root locus of the dc motor with a proportional controller

Fig. 8.6 Output of the load driven by a dc motor vs time with 'p' controller

To determine the controller gains K_P and K_D we use the performances. By equating the desired characteristic equation and the one of the closed-loop dynamics some precautions have to be taken since this process doesn't take into consideration the presence of the zero introduced by the controller. This zero should be placed in the left of the dominant poles that will give the desired behavior.

As before two complex poles are used for the design of the controller. If we equate the two characteristic equations we get:

$$2\zeta w_n = \frac{1 + K_m K_D}{\tau_m}$$

$$w_n^2 = \frac{K_m K_P}{\tau_m}.$$

In this case we have two unknown variables K_P and K_D and two algebraic equations which determines uniquely the gains. Their expressions are given by:

$$K_P = \frac{\tau_m w_n^2}{K_m}$$

$$K_D = \frac{2\tau_m \zeta w_n}{K_m}$$

Using now the desired performances, we conclude similarly as before that the steady error to an input equal to step function of amplitude equal to 1 for instance is equal to zero and the damping ratio ζ corresponding to an overshoot equal to %5 is equal to 0.707. The settling time, t_s at % 5, that we may fix as a proportion of the time constant of the system, gives:

$$w_n = \frac{3}{\zeta t_s}.$$

Now if we fix the settling time at 3τ (which shorter than the one obtained by the proportional controller), we get:

$$w_n = 2.8289.$$

Using these values we get the following ones for the controller gains:

$$K_P = 2.0006$$
$$K_D = 1.0000.$$

which gives the following complex poles:

$$s_{1,2} = -2 \pm 2j.$$

and the zero at:

$$z = -20.8618.$$

Using this controller the time response for an input with an amplitude equal to 1 is represented in Fig. 8.7. As it can seen from this figure that the overshoot and the settling time are less those obtained using the proportional controller.
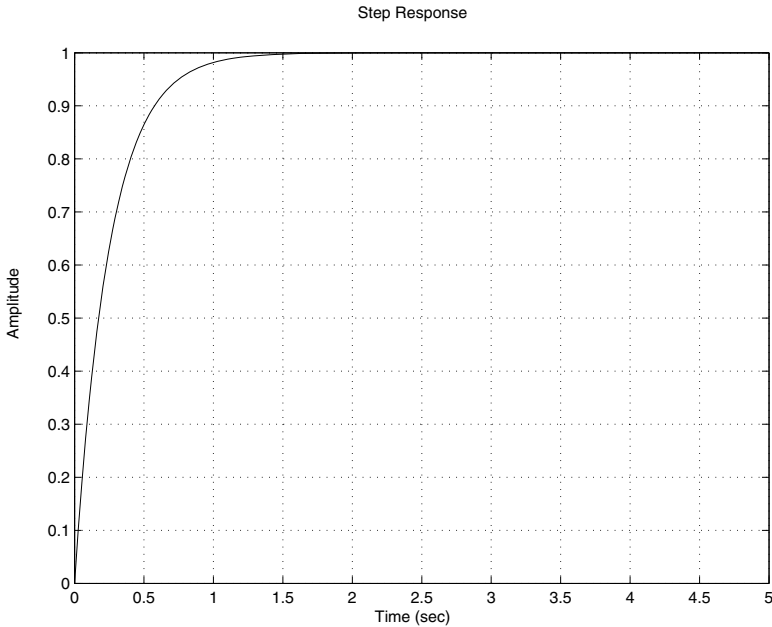
Fig. 8.7 Time response for a step function with 1 as amplitude

For the implementation of these two controllers, the recurrent equation is easy to obtain. For the proportional controller, it is given by:

$$u_k = K_P e_k$$

where K_P is the gain we computed and u_k and e_k represent respectively the control to send and the error between the reference and the output at instant kT, (T is the sampling period).

For the proportional and derivative controller, the trapezoidal method can be used and we get:

$$u_k = -u_{k-1} + ae_k + be_{k-1}$$

where $a = K_P + \frac{2K_D}{T}$ and $b = K_P - \frac{2K_D}{T}$.

8.6   Design and Implementation Based on State Space

This section covers the design and implementation using the state space model. In this section we will assume that the mechanical part and the electronic circuit have already been built and a mathematical model in the state space form is given. Our goal is to focus on the design of the state feedback controller that guarantees the specifications we were fixed for the control design. Referring to what it has been

developed earlier we can say that we have two approaches to design such controller. These two methods are:

- the pole placement (or pole location) technique
- the linear quadratic regulator

These two techniques are not equivalent since the design of the state feedback controller requires:

- for the pole placement techniques to convert the design control specifications to poles and then get the gain controller

- for the linear quadratic control, two matrices, Q and R are first fixed to form the cost function, then the controller gain is obtained by solving the Ricatti equation.

In both cases, we end up with the controller gain, K and the structure of the controller is the same for the two methods and it is given by:

$$u(k) = -Kx(k)$$

where $x(k)$ is the state of the system that we assume to be measurable and $u(k)$ is the control action at period kT, with T is the sampling period.

It is important to mention that when the state vector is not measurable due to the fact that we don't have enough sensors to measure them, an estimator either full or partial order can be built to compute an estimate for this state vector, $\hat{x}(k)$, and that we can use for feedback instead of $x(k)$. Based on the separation principle, the controller and estimator can be done separately, the only restriction that we should keep in mind is that the poles that will be used for the design of the estimator gain must be faster than those used for the computation controller gain.

To illustrate the design and the implementation using the state space model, let us again consider the position control a mechanical part driven by a dc motor and consider the same specifications as in the previous section. Let also assume that all the states are available for feedback.

The state space representation of this system is given by:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{k}{\tau} \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

where $x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$ and $u(t)$ is the voltage we should send to the dc motor, $k = 2$ and $\tau = 0.5$ s are the gain and the constant time of the dc motor.

As specifications, let us consider the following ones:

1. the system is stable in closed-loop;

2. the overshoot is less or equal to 5%;

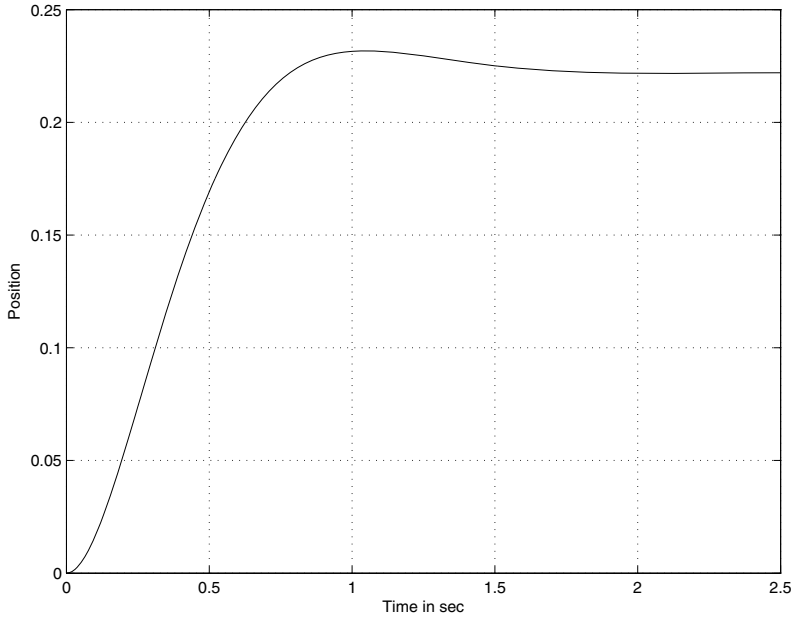3. the settling time at 5% is less or equal to 1 s

Fig. 8.8 Time response for a step function with 1 as amplitude

Let us assume that we have complete access to the states. From the data, it can verified that the system is controllable and therefore, there exists a state feedback controller. From the control specifications, we get the following poles:

$$s_1 = -3.0000 + 3.0009\,j$$
$$s_2 = -3.0000 - 3.0009\,j.$$

Using the pole placement technique, we get the following gain:

$$K = \begin{bmatrix} 4.5014 & 1.0000 \end{bmatrix}.$$

With this controller, the simulation results for a reference equal to 1 is illustrated in Fig. 8.8. From this figure, we can see that the steady error is not equal to zero and therefore an integral action is needed

Let us now assume that we don't have access to the states and try to design a full order observer. Following what it was developed in the previous chapter and the following Matlab program:

```
clear all

%data

tau=0.5
k=2
A = [0 1;0 -1/tau];
```
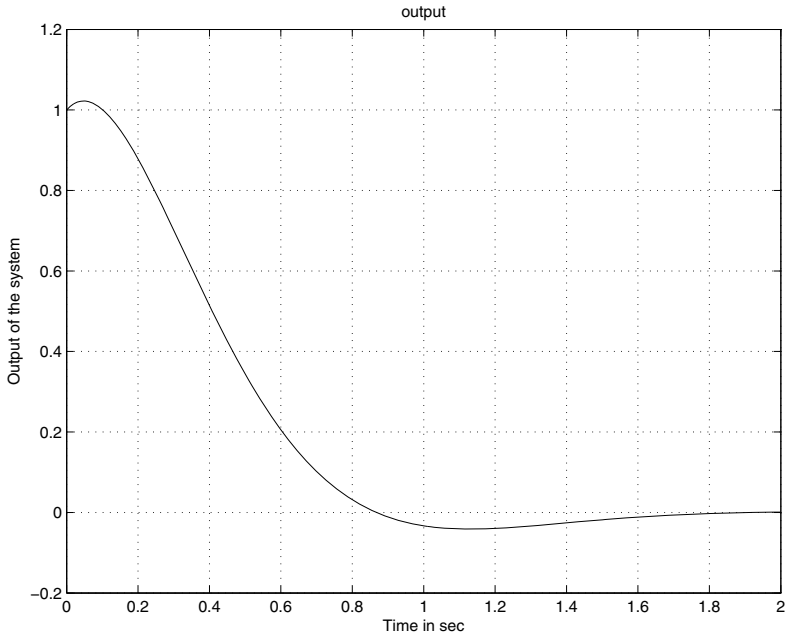
Fig. 8.9 Behavior of the output for a non null initial conditions

```
B = [0 ; k/tau];
C = [1 0];
D = 0;

% controller design
K = acker(A,B,[-3+3*j -3-3*j]);
L = acker(A',C',[-12+3*j -12-3*j])';

% Simualation data
Ts = 0.01;    % step time
Tf = 2;            % final time
x0 = [1 ; 1];
z0 = [1.1 ; 0.9];

%augmented system
Ah = [A              -B*K;
        L*C       A-B*K-L*C];
Bh = zeros(size(Ah,1),1);
Ch = [C D*K];
Dh = zeros(size(Ch,1),1);
xh0 = [x0 ; z0];
t=0:Ts:Tf;
u = zeros(size(t));
```
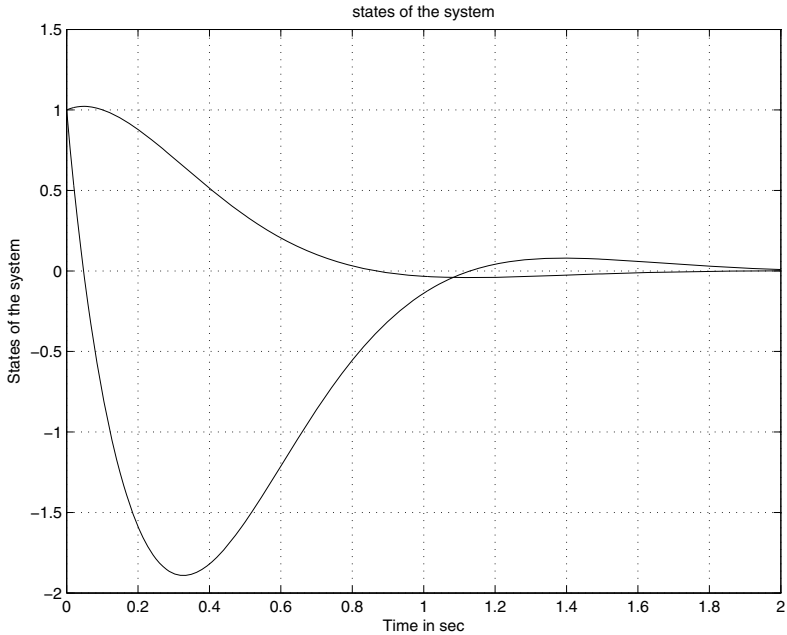
Fig. 8.10 Behavior of the system's states

```
m = ss(Ah,Bh,Ch,Dh);

%simulation
[y,t,x] = lsim(m,u,t,xh0)

%plotting
figure;
plot(t,y);
title('output');

figure;
plot(t,x(:,1:size(A,1)));
title('states of the system');

figure;
plot(t,x(:,size(A,1)+1:end));
title('states of the observer');
```

we get the results of the Figs. 8.9-8.11 that show that the states of the observer converge to those of the system and the output reaches zero.
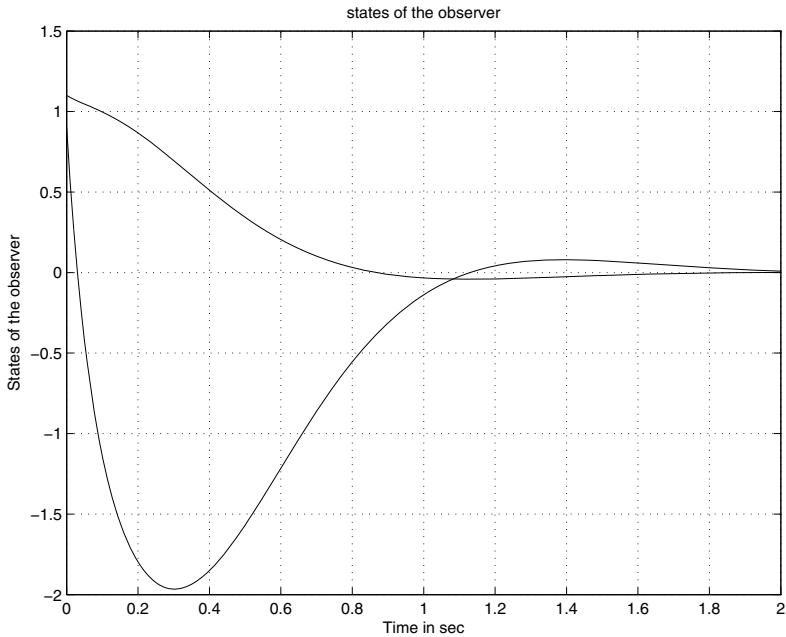
Fig. 8.11 Behavior of the observer's states

8.7   Conclusions

This chapter presents the design of mechatronic systems. Some techniques of how to built such systems are presented with a certain details that can be used by the reader to build his own mechatronic system in the future. The design consists of

- manufacturing the mechanical part of the system

- building the electronic circuit around the dsPIC30F4011 chip

- designing the controller that responds to the desired specifications of the control system design using the appropriate method

- writing the code that will make the different parts of the electronic circuit interact once the implementation in real-time is done.

Methods of how to deal with all these steps were developed. Also, the design and implementation in real time have been covered and illustrated by real system. This has been done for the two cases:

- using the transfer function model

- using the state space model

Some guidelines for the real-time implementation are provided to avoid surprises.

8.8   Problems

1. In this problem, we ask for the design of a vacuum cleaner. This device should be automatic and avoid obstacle in its environment. It is also important to design a cheap one that can communicate wireless via an emitter and receiver.

2. In this problem it is asked to fabricate a small plane that can be controlled wireless via a emitter and a receiver. We would like that this machine can transmit image of the space it "survol"

3. In this problem we ask to design a hoover that can be controlled to seal on water via a emitter and a receiver.

4. In this problem we ask for the design of an insect with four pattes that can walk in a smooth and irregular floor.

5. A dynamical system that we would like to control is described by the following transfer function:

$$G(s) \; = \; \frac{5}{(s+1)(s+5)}$$

   (a) choose the desired performances and design the controller that gives such performances;
   (b) establish the recurrent equation that we would implement in real time
   (c) give the structure of the program that we would write to assure such control in real-time

6. A dynamical system that we would like to control is described by the following transfer function:

$$G(s) \; = \; \frac{5}{(s+1)(s+2)(s+5)}$$

   (a) establish the canonical controllable form
   (b) choose the desired performances and design the controller that gives such performances in the case of full access to the states and partially access to the states;
   (c) establish the recurrent equation that we would implement in real time
   (d) give the structure of the program that we would write to assure such control in real-time

Part VI

Advanced Control

In the last chapters we covered many control concepts that belongs to classic control that were used to build most all the airplanes we are using nowadays.

The aim of this part is to introduce the reader to advanced control theory. It is also important to show the reader that these new concepts are also useful and can be implemented for practical systems.

9

Robust Control

> After reading this chapter the reader will:
>
> 1. understand some concepts on robust control
> 2. be able to resolve the robust stability problem
> 3. be able to solve the robust stabilization problem and know how to compute the controller gain
> 4. manipulate linear matrix inequalities
> 5. be able to use commercial tools to solve convex optimization problem

Previously we showed that any dynamical system can be described by the following state space description:

$$
\begin{cases}
\dot{x}(t) = A_c x(t) + B_c u(t) \\
y(t) = C_c x(t) + D_c u(t)
\end{cases}
\tag{9.1}
$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$ design respectively the state, the input and the output of the system, and A_c, B_c, C_c, and D_c, are known matrices with appropriate dimensions.

We also seen that using a ZOH and an appropriate choice of the sampling period T, we can transform this continuous-time state space description to a discrete-time one with the following structure:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) + Du(k) \end{cases} \tag{9.2}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $y(k) \in \mathbb{R}^p$ represent the values of the state, the input and the output at the instant kT and A, B, C, and D, are known matrices with appropriate dimensions that we obtain from the original matrices A_c, B_c, C_c, and D_c.

Most of the time, the matrices A_c, B_c, C_c, and D_c, and therefore those of the discrete-time description can not be known exactly for many reasons like for instance the dynamics we neglect when building the model or the nonlinearity effects. The model (9.1) or (9.2) can be corrected by adding uncertainties. The new state space description becomes for instance in the discrete-time case as follows:

$$\begin{cases} x(k+1) = [A + \Delta A(t)]\,x(k) + [B + \Delta B(t)]\,u(k) \\ y(k) = [C + \Delta C(t)]\,x(k) + [D + \Delta D(t)]\,u(k) \end{cases} \tag{9.3}$$

The uncertainties $\Delta A(k)$, $\Delta B(k)$, $\Delta C(k)$ and $\Delta D(k)$ are assumed to have the following forms:

$$\Delta A(k) = D_A \Delta F_A(k) E_A$$
$$\Delta B(k) = D_B \Delta F_A(k) E_B$$
$$\Delta C(k) = D_C \Delta F_A(k) E_C$$
$$\Delta D(k) = D_D \Delta F_D(k) E_D$$

$D_A, E_A, D_B, E_B, D_C, E_C, D_D$, and E_D are known real matrices, and the terms $\Delta F_A(k)$, $\Delta F_B(k)$, $\Delta F_C(k)$, and $\Delta F_D(k)$ satisfy the following conditions

$$\Delta A^\top(k)\Delta A(k) \leq \mathbb{I}$$
$$\Delta B^\top(k)\Delta B(k) \leq \mathbb{I}$$
$$\Delta C^\top(k)\Delta C(k) \leq \mathbb{I}$$
$$\Delta D^\top(k)\Delta D(k) \leq \mathbb{I}$$

with $\mathbb{I}$ is the identity matrix with appropriate dimension.

Remark 9.0.1 *The uncertainties that satisfy the previous conditions are referred in the literature to as norm bounded uncertainties. It is important to mention that there exist other types of uncertainties and for more details on this topic we refer the reader to [2].*

Remark 9.0.2 *The uncertainties are called admissible if the imposed conditions on these uncertainties are satisfied.*

Remark 9.0.3 *When the uncertainties are equal to zero we refer to the system as nominal otherwise the system is uncertain. When the term of the input is omitted from the dynamics, we refer to this respectively free nominal system (when the*

uncertainties are equal to zero) and free uncertain system (uncertainties are not equal to zero).

When the uncertainties are equal to zero we learned previously how to check the stability of the system of the form (9.2) by solving the Lyapunov equation given by the following result.

Theorem 9.0.1 *Let Q be a given symmetric and positive-definite matrix. The free nominal system (9.2) is stable iff there exists a symmetric and positive matrix P unique solution of the following Lyapunov equation:*

$$A^\top P A - P = -Q \qquad (9.4)$$

In this chapter we will extend this result and establish sufficient condition to check the stability of the system when the uncertainties are not equal to zero. We will refer to this as robust stability.

We will also consider the stabilization problem either for nominal or uncertain system using different type of controllers. Among these controllers, we quote:

- the state feedback controller

- the static output feedback controller

- and the output feedback control

The rest of this chapter is organized as follows. In Section 1, the stability problem is tackled, while in Section 2, the stabilization problem is covered and linear matrix inequality (LMI) conditions are developed in both cases. In case of uncertain systems, the LMIs we will develop are only sufficient conditions and if we are not able to find a solution to these LMIs, we can not conclude that the system is not stable or not stabilizable depending on the problem we are considering.

9.1 Stability Problem

Firstly, let us consider the free nominal system and see under which condition the system will be stable. If we refer to the previous chapters, the system (9.2) is stable, if we can find a Lyapunov function $V(x_k)$ such that

$$\Delta V = V(x_{k+1}) - V(x_k) < 0 \qquad (9.5)$$

In fact, we used previously the same reasoning to establish the Lyapunov stability condition. Now, if we consider a Lyapunov function, $V(x_k)$, with the following form:

$$V(x_k) = x_k^\top P x_k$$

where P is a symmetric and positive-definite matrix with appropriate dimension.

Let us compute $\Delta V = V(x_{k+1}) - V(x_k)$. This gives:

$$\Delta V = V(x_{k+1}) - V(x_k) = x_{k+1}^\top P x_{k+1} - x_k^\top P x_k$$

Using now the dynamics of the free nominal system, since the stability doesn't depend on the input for linear time invariant systems, we get:

$$\Delta V = [Ax_k]^\top P [Ax_k] - x_k^\top P x_k$$
$$= x_k^\top A^\top P A x_k - x_k^\top P x_k$$

Rearranging the terms we get:

$$\Delta V = x_k^\top \left[A^\top P A - P \right] x_k$$

Now if we impose that $[A^\top P A - P] < 0$, we get $\Delta V < 0$ and therefore the system is stable.

Remark 9.1.1 *Caution has to be made for the notation we made regarding the relation, $[A^\top P A - P] < 0$. A scalar can be negative but a matrix cannot be. By this notation we mean that the matrix has all its eigenvalues negative, or it is symmetric and negative-definite.*

This gives us the following results.

Theorem 9.1.1 *The free nominal linear system 9.2 is stable iff there exists a symmetric and positive-definite matrix, P, solution of the following LMI:*

$$A^\top P A - P < 0 \tag{9.6}$$

The following lemma will be used extensively in this chapter.

Lemma 9.1.1 *(Schur Complement) Let the symmetric matrix M be partitioned as*

$$M = \begin{bmatrix} X & Y \\ Y^\top & Z \end{bmatrix},$$

with X and Z being symmetric matrices. We have

(i) M is nonnegative-definite if and only if either

$$\begin{cases} Z \geq 0, \\ Y = L_1 Z \\ X - L_1 Z L_1^\top \geq 0 \end{cases} \tag{9.7}$$

or

$$\begin{cases} X \geq 0 \\ Y = X L_2 \\ Z - L_2^\top X L_2 \geq 0 \end{cases} \tag{9.8}$$

holds, where L_1, L_2 are some (nonunique) matrices of compatible dimensions;

(ii) M is positive-definite if and only if either

$$\begin{cases} Z > 0, \\ X - Y Z^{-1} Y^\top > 0, \end{cases} \tag{9.9}$$

or

$$\begin{cases} X > 0 \\ Z - Y^\top X^{-1} Y > 0 \end{cases} \qquad (9.10)$$

The matrix $X - YZ^{-1}Y^\top$ is called the Schur complement $X(Z)$ in M.

Remark 9.1.2 *It is important to notice that the LMI of this theorem is equivalent to the following one:*

$$\begin{bmatrix} -P & A^\top P \\ PA & -P \end{bmatrix} < 0$$

To get this LMI we used, the fact that $PP^{-1} = \mathbb{I}$ (i.e.: $A^\top PP^{-1}PA - P$) and the Schur complement.

Remark 9.1.3 *It is important to notice that the condition of the previous theorem are obtained directly from the Lyapunov equation. In fact we have the matrix Q which is symmetric and positive-definite, i.e: all its eigenvalues are positive. This implies that the matrix $-Q$ is symmetric and negative-definite, i.e: all its eigenvalues are negative. And using the notation, the result follows.*

It is well known that the poles of the system may move for many reasons among them we quote the changes of the system's parameters. This may cause instability and degradation of performances. One way to overcome this is to take precaution by using the concept of degree of stability. This may prevent instability for instance in case of change of parameters. The following result may be used to determine if the system under study has or not a degree of stability equal to $\alpha > 0$.

Corollary 9.1.1 *The free nominal linear system 9.2 has a degree of stability equal to $\alpha > 0$, iff there exists a symmetric and positive-definite matrix, P, solution of the following LMI:*

$$[A + \alpha\mathbb{I}]^\top P [A + \alpha\mathbb{I}] - P < 0 \qquad (9.11)$$

Remark 9.1.4 *Similarly as we did in the previous remark, we can transform the LMI of this Corollary to the equivalent following one:*

$$\begin{bmatrix} -P & [A + \alpha\mathbb{I}]^\top P \\ P[A + \alpha\mathbb{I}] & -P \end{bmatrix} < 0$$

Let us now focus on the robust stability problem of the free uncertain system 9.2. To establish the required sufficient condition for robust stability we need the following lemma.

Lemma 9.1.2 *Let J_A, D_A and E_A be real matrices of appropriate dimensions and J_A is a symmetric matrix, then:*

$$J_A + D_A F_A(k)E_A(t) + [D_A F_A(k)E_A(t)]^\top < 0$$

for any $F_A(k)$ that satisfies $F_A^\top(t)F_A(k) \leq \mathbb{I}$ if and only if there exists a scalar $\varepsilon_A > 0$ such that:

$$J_A + \varepsilon_A D_A D_A^\top(t) + \varepsilon_A^{-1} E_A^\top E_A(t) < 0$$

For the free uncertain system, the stability condition implies that the following must hold for all admissible uncertainties:

$$[A + D_A F_A(k)E_A]^\top P [A + D_A F_A(k)E_A] - P < 0$$

Using now the Schur complement and the fact that $PP^{-1} = \mathbb{I}$, we get:

$$\begin{bmatrix} -P & \star \\ PA + PD_A F_A(k)E_A & -P \end{bmatrix} < 0$$

which can be rewritten as follows:

$$\begin{bmatrix} -P & \star \\ PA & -P \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ PD_A F_A(k)E_A & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ PD_A F_A(k)E_A & 0 \end{bmatrix}^\top < 0$$

where $\star$ is the transpose of PA.

Notice that:

$$\begin{bmatrix} 0 & 0 \\ PD_A F_A(k)E_A & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ PD_A \end{bmatrix} F_A(k) \begin{bmatrix} E_A & 0 \end{bmatrix}$$

Using this we get:

$$\begin{bmatrix} -P & \star \\ PA & -P \end{bmatrix} + \begin{bmatrix} 0 \\ PD_A \end{bmatrix} F_A(k) \begin{bmatrix} E_A & 0 \end{bmatrix} + \left[\begin{bmatrix} 0 \\ PD_A \end{bmatrix} F_A(k) \begin{bmatrix} E_A & 0 \end{bmatrix} \right]^\top < 0$$

Based on Lemma 9.1.2, this condition will hold if the following one holds:

$$\begin{bmatrix} -P & \star \\ PA & -P \end{bmatrix} + \varepsilon_A^{-1} \begin{bmatrix} 0 \\ PD_A \end{bmatrix} \begin{bmatrix} 0 & D_A^\top P \end{bmatrix} + \varepsilon_A \begin{bmatrix} E_A^\top \\ 0 \end{bmatrix} \begin{bmatrix} E_A & 0 \end{bmatrix} < 0$$

for $\varepsilon_A > 0$.

This condition gives in turn after using the Schur complement:

$$\begin{bmatrix} -P + \varepsilon_A E_A^\top E_A & A^\top P & 0 \\ PA & -P & PD_A \\ 0 & D_A^\top P & -\varepsilon_A \mathbb{I} \end{bmatrix} < 0$$

Based on this development we get the following results:

Theorem 9.1.2 *The free unceratin linear system 9.2 is stable if there exist a symmetric and positive-definite matrix and a positive scalar, ε_A, solution of the following LMI:*

$$\begin{bmatrix} -P + \varepsilon_A E_A^\top E_A & A^\top P & 0 \\ PA & -P & PD_A \\ 0 & D_A^\top P & -\varepsilon_A \mathbb{I} \end{bmatrix} < 0 \tag{9.12}$$

Remark 9.1.5 *The condition of this theorem is only sufficient and therefore, if the LMI is not satisfied this doesn't imply that the system is not stable.*

The study of stability is brought to the resolution of an LMI. In the marketplace, there are many tools that can be used to solve such LMI. Among these tools we mention:

- LMI toolbox of Matlab (not free software)

- Scilab (free software)

- Yalmip and Sedumi (free softwares that are based on Matlab)

In order to show how the stability problem is solved let us consider a numerical example.

Example 9.1.1 *To show how to solve the stability problem, let us consider a dynamical free nominal system with the following data:*

$$
A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.2 & 0.3 & -0.1 \end{bmatrix}
$$

We can either solve the Lyapunov equation for this system or use Matlab to compute the eigenvalues of the matrix A. Using Matlab, the eigenvalues of A are:

$$
s_1 = -0.7945
$$
$$
s_{2,3} = 0.3472 \pm 0.3622 j
$$

that are all inside the unit circle and therefore, the system is stable.

To use the result of the previous theorem, the following program can be used to solve the robust stability problem for our system.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Stability of linear systems %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Nominal case
  clear all;
  yalmip('clear')

% Data
  A=[0 1 0; 0 0 1;-0.2 0.3 -0.1];

  n=size(A,1)

% Variables
  P=sdpvar(n,n,'symmetric'); % declaration

% LMI
  F=set(P>0)  % Initialization
  F=F+set([ -P A'*P;
            P*A -P] < 0)

% Solve
```

```
  Sol=solvesdp(F)

% Extract data from the solution
  P=double(P)
  eig(P)        % check if the eigenvalues are positive

  checkset(F)
```

If we run this program the LMI is feasible and we get:

$$P = \begin{bmatrix} 0.7397 & -0.0036 & -0.1857 \\ -0.0036 & 1.0708 & 0.0857 \\ -0.1857 & 0.0857 & 1.6233 \end{bmatrix}$$

The eigenvalues of this matrix are:

$$s_1 = 0.7017$$
$$s_2 = 1.0594$$
$$s_3 = 1.6727$$

that are all positive and therefore, the system is stable.

Example 9.1.2 *To show how to solve the robust stability problem, let us consider a dynamical free uncertain system with the following data:*

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ -0.2 & 0.3 & 0.1 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix},$$

$$E_A = \begin{bmatrix} 0.3 \\ 0.2 \\ 0.1 \end{bmatrix}.$$

It can be checked that the system is unstable using Matlab for the computation of the eigenvalues.

To use the result on the robust stability, the following program can be used to solve the robust stability problem for our system.

```
%%%%%%%%%%%%%%%%%%%
% Robust stability of linear system  %
%%%%%%%%%%%%%%%%%%%%

% Uncertain system
clear all;
yalmip('clear')

% Data
  A=[0 1 0; 0 0 1;-0.2 0.3 0.1]
  Da=[0.1; 0.2; 0.3]
  Ea=[0.3 0.2 0.1]
```

```
        n=size(A,1)

    % Variables
      P=sdpvar(n,n,'symmetric');
      eps=sdpvar(1,1,'full');

    % LMI
      F=set(P>0)  % Initilization
      F=F+set(eps>0)

      F=F+set([-P + eps*Ea'*Ea  A'*P  zeros(n,1);
               P*A                -P   P*Da;
               zeros(1,n)        Da'*P -eps*eye(1,1)] < 0)

    % Solve
      Sol=solvesdp(F)

    % Extract data from the solution
      P=double(P)
      eps=double(eps)

      eig(P)
      checkset(F)
```

If we run this program the LMI is feasible and we get:

$$\varepsilon_A = 0.8911$$

$$P = \begin{bmatrix} 0.5335 & -0.0144 & -0.1331 \\ -0.0144 & 0.7360 & -0.0442 \\ -0.1331 & -0.0442 & 1.1060 \end{bmatrix}$$

The eigenvalues of this matrix are:

$$s_1 = 0.5017$$
$$s_2 = 0.7344$$
$$s_3 = 1.1394$$

that are all positive and therefore, the system is robustly stable for all admissible uncertainties.

We considered in the previous example two stable systems and therefore, we don't need a controller to stabilize them unless we would like to improve the performances such as the reduction of the settling time and the overshoot of the outputs. In this case a controller is needed and its design must be done. In the previous chapters, we learned how to design state feedback controller either using poles placement or linear quadratic regulator. In these two approaches, we need to have some expertise to convert the performances to some equations and then get the gain of the con-

troller. In the next section, we will present another approach that doesn't require this and we will cover only the stabilization problem.

9.2 Stabilization

The stabilization problem is one of the most important control problem. It consists of designing a controller that renders the closed-loop dynamics of the system stable. The stabilizability can be used for the following purposes:

- stabilize an unstable system

- improve the performances of a given system

A nominal system is said to be stabilizable if there is a controller that makes the closed-loop dynamics of the system stable. For uncertain system, we said that the system is robustly stabilizable if there is a controller that makes the closed-loop dynamics of the system stable.

The controller that we can design to stabilize the system under study can be one of the following controllers:

- state feedback

- static output feedback

- dynamic output feedback

The choice among these controllers is based on the availability or the nonavailability of the states of the system. Therefore, if we have access to the states we recourse to the use of the state feedback controller otherwise, the states are estimated and this estimate is used for feedback.

Definition 9.2.1 *The nominal system is stabilizable if there exists a control law that makes the closed-loop stable.*

Definition 9.2.2 *The uncertain system is stabilizable if there exists a control law that makes the closed-loop stable for all admissible uncertainties.*

Let us assume that we have complete access to the states of the system and design a state feedback controller of the previous forms that stabilizes the nominal system:

$$u_k = K x_k$$

where K is gain matrix that we have to determine.

Combining the nominal system dynamics and the controller expression we get the following closed-loop system:

$$
\begin{aligned}
x_{k+1} &= [A + BK] x_k \\
&= A_{cl} x_k
\end{aligned}
$$

with $A_{cl} = A + BK$

Our objective is to design the state feedback controller of the previous form and compute its gain matrix K in order to make the closed-loop system stable. Using the results on stability of the nominal system, the closed-loop system will be stable if there exists a symmetric and positive-definite matrix $P > 0$ such that the following LMI holds:

$$\begin{bmatrix} -P & A_{cl}^{\top} P \\ P A_{cl} & -P \end{bmatrix} < 0$$

Using the expression of A_{cl} we get:

$$\begin{bmatrix} -P & [A + BK]^{\top} P \\ P[A + BK] & -P \end{bmatrix} < 0$$

where P and K are design parameters.

This inequality is nonlinear in the design parameters P and K. To overcome this, we let $X = P^{-1}$, and pre- and post-multiply the left hand side respectively by $\text{diag}(X, X)$, we get:

$$\begin{bmatrix} -X & X[A + BK]^{\top} \\ [A + BK]X & -X \end{bmatrix} < 0$$

Letting now $Y = KX$ implies in turn:

$$\begin{bmatrix} -X & XA^{\top} + Y^{\top} B^{\top} \\ AX + BY & -X \end{bmatrix} < 0$$

Based on this development, we get the following result.

Theorem 9.2.1 *There exists a state feedback controller that stabilizes the nominal system if there exist a symmetric and positive-definite matrix $X > 0$ and a matrix Y that satisfy the the following LMIs:*

$$\begin{cases} X > 0 \\ \begin{bmatrix} -X & XA^{\top} + Y^{\top} B^{\top} \\ AX + BY & -X \end{bmatrix} < 0 \end{cases}$$

The controller gain is given by $K = YX^{-1}$

As we made for the stability, we can also design a state feedback controller that stabilizes the system with a certain degree of stability $\alpha > 0$. This controller can be designed using the results of the following corollary.

Corollary 9.2.1 *Let α be a given positive scalar. There exists a state feedback controller that stabilizes the nominal system if there exist a symmetric and positive-definite matrix $X > 0$ and a matrix Y that satisfy the the following LMIs:*

$$\begin{cases} X > 0 \\ \begin{bmatrix} -X & X[A + \alpha\mathbb{I}]^{\top} + Y^{\top} B^{\top} \\ [A + \alpha\mathbb{I}]X + BY & -X \end{bmatrix} < 0 \end{cases}$$

The controller gain is given by $K = YX^{-1}$

Example 9.2.1 *To show how we can design the state feedback controller that stabilizes the nominal system let us consider the following unstable nominal system:*

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 0 & -2 & 1 \\ 2 & -1 & -2 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 1 \\ 2 & 0 \\ 1 & 1 \end{bmatrix}$$

Our goal is to design the gain, K of the state feedback controller that stabilizes our system. For this purpose, we write the following program.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Stabilizability of linear systems %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Nominal case
  clear all;
  yalmip('clear')

% Data
  A=[1 0 -1; 0 -2 1; 2 -1 -2];
  B=[0 1; 2 0; 1 1];

  n=size(A,1);
  m=size(B,2);

% Variables
  X=sdpvar(n,n,'symmetric'); % declaration
  Y=sdpvar(m,n,'full');

% LMI
  F=set(X>0)  % Initialization
  F=F+set([ -X              X*A'+Y'*B' ;
                   A*X+B*Y   -X] < 0)

% Solve
  Sol=solvesdp(F)

% Extract data from the solution
  X=double(X)
  Y=double(Y)
  eig(X)        % check if the eigenvalues are positive

 % compute the controller gain
  K=Y*inv(X)
  checkset(F)
```

If we run this program the LMI is feasible and we get:

$$X = \begin{bmatrix} 416.9192 & 109.9989 & 106.7370 \\ 109.9989 & 532.7776 & -46.9652 \\ 106.7370 & -46.9652 & 207.6739 \end{bmatrix}$$

$$Y = \begin{bmatrix} -23.2314 & 498.6871 & -88.5828 \\ -444.8524 & -252.1721 & 207.7342 \end{bmatrix}$$

The eigenvalues of the matrix X are:

$$s_1 = 142.7975$$
$$s_2 = 414.7446$$
$$s_3 = 599.8286$$

that are all positive and therefore, the matrix, X, is symmetric and positive definite. The corresponding controller is given by:

$$K = \begin{bmatrix} -0.3073 & 0.9956 & -0.0435 \\ -1.5229 & -0.0018 & 1.7826 \end{bmatrix}.$$

Let us now consider the uncertain system and see how we can design a state feedback controller that stabilizes the system for all admissible uncertainties. Combining the uncertain system dynamics and the controller expression we get the following closed-loop system:

$$\begin{aligned} x_{k+1} &= [A + BK + \Delta A + \Delta BK] x_k \\ &= [A_{cl} + \Delta A + \Delta BK] x_k \end{aligned}$$

with $A_{cl} = A + BK$

The design of the state feedback controller is brought to the computation of the controller gain matrix K that makes the closed-loop system stable for all admissible uncertainties. Based on the results on stability of uncertain system, the closed-loop system will be stable if there exist a symmetric and positive-definite matrix $P > 0$ such that the following holds:

$$\begin{bmatrix} -P & (A_{cl} + \Delta A + \Delta BK)^\top P \\ \star & -P \end{bmatrix} < 0$$

that can be rewritten as follows:

$$\begin{bmatrix} -P & A_{cl}^\top P \\ \star & -P \end{bmatrix} + \begin{bmatrix} 0 & (D_A F_A(k)E_A)^\top P \\ \star & 0 \end{bmatrix} + \begin{bmatrix} 0 & (D_B F_B(k)E_B K)^\top P \\ \star & 0 \end{bmatrix} < 0$$

Notice that we have:

$$\begin{bmatrix} 0 & 0 \\ PD_A F_A(k)E_A & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ PD_A \end{bmatrix} F_A(k) \begin{bmatrix} E_A & 0 \end{bmatrix}$$

and

$$\begin{bmatrix} 0 & 0 \\ PD_BF_B(k)E_BK & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ PD_B \end{bmatrix} F_B(k) \begin{bmatrix} E_AK & 0 \end{bmatrix}$$

Using now Lemma 9.1.2 we get:

$$\begin{bmatrix} -P & A_{cl}^\top P \\ \star & -P \end{bmatrix} + \varepsilon_A \begin{bmatrix} 0 \\ PD_A \end{bmatrix} \begin{bmatrix} 0 & (PD_A)^\top \end{bmatrix} + \varepsilon_A^{-1} \begin{bmatrix} E_A^\top \\ 0 \end{bmatrix} \begin{bmatrix} E_A & 0 \end{bmatrix}$$

$$+ \varepsilon_B \begin{bmatrix} 0 \\ PD_A \end{bmatrix} \begin{bmatrix} 0 & (PD_B)^\top \end{bmatrix} + \varepsilon_B^{-1} \begin{bmatrix} K^\top E_B^\top \\ 0 \end{bmatrix} \begin{bmatrix} E_BK & 0 \end{bmatrix} < 0$$

for any positive numbers $\varepsilon_A > 0$ and $\varepsilon_B > 0$.

Based on Schur complement the following LMI must hold for all admissible uncertainties to guarantee the stability of the closed-loop system:

$$\begin{bmatrix} -P + \varepsilon_A^{-1} E_A^\top E_A + \varepsilon_B^{-1} K^\top E_B^\top E_BK & A_{cl}^\top P & 0 & 0 \\ PA_{cl} & -P & PD_A & PD_B \\ 0 & D_A^\top P & -\varepsilon_A^{-1}\mathbb{I} & 0 \\ 0 & D_B^\top P & 0 & -\varepsilon_B^{-1}\mathbb{I} \end{bmatrix} < 0$$

Using the expression of A_{cl} and the uncertainty lemma, we get:

$$\begin{bmatrix} J & [A + BK]^\top P & 0 & 0 \\ P[A + BK] & -P & PD_A & PD_B \\ 0 & D_A^\top P & -\varepsilon_A^{-1}\mathbb{I} & 0 \\ 0 & D_B^\top P & 0 & -\varepsilon_B^{-1}\mathbb{I} \end{bmatrix} < 0$$

where $J = -P + \varepsilon_A^{-1} E_A^\top E_A + \varepsilon_A^{-1} K^\top E_B^\top E_BK$, $\varepsilon_A > 0$, $\varepsilon_B > 0$ are positive scalars, and P and K design parameters

This inequality is nonlinear in the design parameters P and K. To overcome this, we let $X = P^{-1}$, and pre- and post-multiply the left hand side respectively by $\mathrm{diag}(X, X, \mathbb{I}, \mathbb{I})$ we get:

$$\begin{bmatrix} XJX & XA^\top + XK^\top B^\top & 0 & 0 \\ AX + BKX & -X & D_A & D_B \\ 0 & D_A^\top & -\varepsilon_A^{-1}\mathbb{I} & 0 \\ 0 & D_B^\top & 0 & -\varepsilon_B^{-1}\mathbb{I} \end{bmatrix} < 0$$

with $XJX = -X + \varepsilon_A^{-1} XE_A^\top E_AX + \varepsilon_A^{-1} XK^\top E_B^\top E_BKX$

Letting $Y = KX$ and using the Schur complement imply:

$$\begin{bmatrix} -X & XA^\top + Y^\top B^\top & XE_A^\top & Y^\top E_B^\top \\ AX + BY & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 \\ E_AX & 0 & -\varepsilon_A\mathbb{I} & 0 \\ E_BY & 0 & 0 & -\varepsilon_B\mathbb{I} \end{bmatrix} < 0$$

Theorem 9.2.2 *There exists a state feedback controller that stabilizes the uncertain system if there exist a symmetric and positive-definite matrix X > 0, a matrix Y and positive scalars $\varepsilon_A > 0$ and $\varepsilon_B > 0$ that satisfy the the following LMIs:*

$$\begin{cases} X > 0 \\ \begin{bmatrix} -X & XA^\top + Y^\top B^\top & XE_A^\top & Y^\top E_B^\top \\ AX + BY & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 \\ E_A X & 0 & -\varepsilon_A \mathbb{I} & 0 \\ E_B Y & 0 & 0 & -\varepsilon_B \mathbb{I} \end{bmatrix} < 0 \end{cases}$$

The controller gain is given by $K = YX^{-1}$.

Remark 9.2.1 *The conditions of this theorem are only sufficient and therefore, if the LMIs are not satisfied this doesn't imply that the system is not stabilizable*

As we made for the stabilization of the nominal, we can also design a sate feedback controller that stabilizes the uncertain system with a certain degree of stability $\alpha > 0$. This controller can design using the results of the following corollary.

Corollary 9.2.2 *Let α be a given positive scalar. There exists a state feedback controller that stabilizes the uncertain system if there exist a symmetric and positive-definite matrix X > 0, a matrix Y and positive scalars $\varepsilon_A > 0$ and $\varepsilon_B > 0$ that satisfy the the following LMIs:*

$$\begin{cases} X > 0 \\ \begin{bmatrix} -X & X\left[A + \alpha \mathbb{I}\right]^\top + Y^\top B^\top & XE_A^\top & Y^\top E_B^\top \\ \left[A + \alpha \mathbb{I}\right]X + BY & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 \\ E_A X & 0 & -\varepsilon_A \mathbb{I} & 0 \\ E_B Y & 0 & 0 & -\varepsilon_B \mathbb{I} \end{bmatrix} < 0 \end{cases}$$

The controller gain is given by $K = YX^{-1}$.

Example 9.2.2 *To show how we can design the state feedback controller that stabilizes the uncertain system let us consider system of the previous example with the following extra data:*

$$D_A = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}, E_A = \begin{bmatrix} 0 & 0.1 & 0.4 \end{bmatrix}$$

$$D_B = \begin{bmatrix} 0.3 \\ 0.2 \\ 0.1 \end{bmatrix}, E_B = \begin{bmatrix} 0.1 & 0.2 \end{bmatrix}$$

Our goal is to design the gain, K of the state feedback controller that stabilizes our system. For this purpose, we write the following program.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Stabilizability of linear systems %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Uncertainl case
  clear all;
  yalmip('clear')

% Data
  A=[1 0 -1; 0 -2 1; 2 -1 -2];
  B=[0 1; 2 0; 1 1];

  Da=[0.1; 0.2; 0.3];
  Ea=[0 0.1 0.4];
  Db=[0.3; 0.2; 0.1];
  Eb=[0.1 0.2];

  n=size(A,1);
  m=size(B,2);

% Variables
  X=sdpvar(n,n,'symmetric'); % declaration
  Y=sdpvar(m,n,'full');
  epsa=sdpvar(1);
  epsb=sdpvar(1);

% LMI
  F=set(X>0)  % Initialization
  F=F+set(epsa>0);
  F=F+set(epsb>0);
  J=-X+epsa*Da*Da'+epsb*Db*Db';
  F=F+set([-X     X*A'+Y'*B'        X*Ea'   Y'*Eb';
           A*X+B*Y  J  zeros(n,1)     zeros(n,1) ;
           Ea*X   zeros(1,n)  -epsa*eye(1)  zeros(1,1) ;
           Eb*Y  zeros(1,n)  zeros(1,1)     -epsb*eye(1) ] < 0)

% Solve
  Sol=solvesdp(F)

% Extract data from the solution
  X=double(X)
  Y=double(Y)
  epsa=double(epsa)
  epsb=double(epsb)
  eig(X)       % check if the eigenvalues are positive

 % compute the controller gain
 K=Y*inv(X)
 checkset(F)
```

If we run this program the LMIs are feasible and we get:

$$\varepsilon_A = 292.1496, \varepsilon_B = 325.6510$$

$$X = \begin{bmatrix} 311.8397 & 112.0526 & 87.9395 \\ 112.0526 & 502.6397 & -47.2877 \\ 87.9395 & -47.2877 & 157.7885 \end{bmatrix},$$

$$Y = \begin{bmatrix} 0.4291 & 455.4875 & -71.0439 \\ -299.5657 & -247.9759 & 132.7033 \end{bmatrix}.$$

The eigenvalues of the matrix, X, are:

$$s_1 = 96.8537,$$
$$s_2 = 320.9106,$$
$$s_3 = 554.5035,$$

that are all positive and therefore, the matrix, X, is symmetric and positive definite. The corresponding controller is given by:

$$K = \begin{bmatrix} -0.3712 & 0.9941 & 0.0545 \\ -1.4066 & -0.0277 & 1.6167 \end{bmatrix}.$$

Let us now assume that we don't have access to the state vector and try to design a static feedback controller of the following form:

$$u(k) = Ky(k) \tag{9.13}$$

where K is the gain controller that we have to design and $y(k)$ is the measured output that we get through the sensors.

Combining now the system's dynamics (9.2) and the controller (9.13), we get:

$$x(k + 1) = [A + \Delta A(t)] x(k) + [B + \Delta B(t)] Ky(k)$$
$$= [A_{cl} + \Delta A_{cl}(t)] x(k)$$

where $A_{cl} = A + BK$ and $A_{cl} = \Delta A(t) + \Delta B(t)K$.

The output equation is given by:

$$y(k) = Cx(k)$$

where $y(k) \in \mathbb{R}^p$ is the output system and C is a known matrix with appropriate dimension.

Based on the previous results, the nominal system with the static output controller will be stable if the following holds:

$$\begin{bmatrix} -P & (A + BKC)^\top P \\ \star & -P \end{bmatrix} < 0.$$

Let $X = P^{-1}$ and pre- and post-multiply this inequality by $\text{diag}(X, X)$ we get:

$$\begin{bmatrix} -X & \star \\ AX + BKCX & -X \end{bmatrix} < 0.$$

The term $BKCX$ is nonlinear since it contains the two decision variables K and X. To overcome this we introduce the new condition:

$$CX = NC$$

which may be restrictive in some conditions.

Using this and letting $Y = KN$, we get the following result.

Theorem 9.2.3 *There exists a static output controller of the form (9.13) if there exist a symmetric and positive-definite matrices X and N and a matrix Y such that the following hold:*

$$\begin{cases} CX = NC \\ \begin{bmatrix} -X & \star \\ AX + BYC & -X \end{bmatrix} < 0. \end{cases} \tag{9.14}$$

The controller gain is given by $K = YN^{-1}$.

For the uncertain system, the closed-loop system will be stable is the following holds for all admissible:

$$\begin{cases} CX = NC \\ \begin{bmatrix} -X & \star \\ (A + \Delta A) X + (B + \Delta B) YC & -X \end{bmatrix} < 0. \end{cases}$$

Notice that the second inequality can be rewritten as follows:

$$\begin{bmatrix} -X & \star \\ AX + BYC & -X \end{bmatrix} + \begin{bmatrix} 0 & \star \\ \Delta AX & 0 \end{bmatrix} + \begin{bmatrix} 0 & \star \\ \Delta BYC & 0 \end{bmatrix} < 0.$$

The second terms can be in turn written as follows:

$$\begin{bmatrix} 0 & \star \\ D_A F_A(k) E_A X & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ D_A \end{bmatrix} F_A(k) \begin{bmatrix} E_A X & 0 \end{bmatrix}$$

and

$$\begin{bmatrix} 0 & \star \\ D_B F_B(k) E_B BYC & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ D_B \end{bmatrix} F_B(k) \begin{bmatrix} E_B BYC & 0 \end{bmatrix}.$$

Using this and Lemma 9.1.2, we get:

$$\begin{bmatrix} 0 & \star \\ D_A F_A(k) E_A X & 0 \end{bmatrix} \le \varepsilon_A \begin{bmatrix} 0 \\ D_A \end{bmatrix} \begin{bmatrix} 0 & D_A^\top \end{bmatrix} + \varepsilon_A^{-1} \begin{bmatrix} (E_A X)^\top \\ 0 \end{bmatrix} \begin{bmatrix} E_A X & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & \star \\ D_B F_B(k) E_B BYC & 0 \end{bmatrix} \le \varepsilon_B \begin{bmatrix} 0 \\ D_B \end{bmatrix} \begin{bmatrix} 0 & D_B^\top \end{bmatrix} + \varepsilon_B^{-1} \begin{bmatrix} (E_B BYC)^\top \\ 0 \end{bmatrix} \begin{bmatrix} E_B BYC & 0 \end{bmatrix}.$$

for any ε_A and ε_B real positive scalars.

Using this relations and Schur complement we get the following result.

Theorem 9.2.4 *There exists a static output controller of the form (9.13) if there exist a symmetric and positive-definite matrices X and N and a matrix Y such that the following hold for all admissible uncertainties:*

$$
\begin{cases}
CX = NC \\
\begin{bmatrix}
-X & \star & & \star & \star \\
AX + BYC & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & \star & \star \\
E_A X & 0 & -\varepsilon_A \mathbb{I} & \star \\
E_B Y C & 0 & 0 & \varepsilon_B \mathbb{I}
\end{bmatrix} < 0.
\end{cases}
\tag{9.15}
$$

The controller gain is given by $K = YN^{-1}$.

To ovoid the conservatism that may result from the condition we introduced previously, let us consider another approach. Before presenting this approach let us firstly introduce the following definition.

Definition 9.2.3 *The column (row) rank of a matrix G is equal to maximal number of linearly independent columns (raws) of G. A matrix $G \in \mathbb{R}^{m\times}$ is equal at most to $\min(m, n)$. This matrix is said to be full column (row) rank if its rank is equal to $\max(m, n)$.*

Remark 9.2.2 *A matrix G that has full column (row) rank implies that exists a nonsingular transformation T_r (T_l) such that we get:*

$$
T_r G = \begin{bmatrix} \mathbb{I} \\ 0 \end{bmatrix} \left(G T_l = \begin{bmatrix} \mathbb{I} & 0 \end{bmatrix} \right)
$$

It is important to notice that the transformation T_r (T_l) is not unique. In fact we can always obtain another transformation using the following:

$$
T_r = \begin{bmatrix} (G^\top B)^{-1} B^\top \\ G^{\top \perp \top} \end{bmatrix} \left(T_l = \begin{bmatrix} G^\top (GG^\top)^{-1} & G^\perp \end{bmatrix} \right)
$$

where $B^{\top\perp\top}$ is the transpose on an orthogonal basis for null space of $G^\top$ and $G^\perp$ is the orthogonal basis of the null space of the matrix G.

The following lemma will be used in developing our results.

Lemma 9.2.1 *Let $\xi \in \mathbb{R}^n$, $P \in \mathbb{R}^{n\times n}$ a symmetric and positive-definite matrix and a matrix $S \in \mathbb{R}^{m\times n}$ such that $\text{rank}(S) = r < n$, then the following statements are equivalent:*

1. $\xi^\top P \xi < 0, \forall \xi \neq 0, S\xi = 0$;
2. $\exists X \in \mathbb{R}^{n\times m}$ such that $P + XS + S^\top X^\top < 0$

In the following we will give two results to design the controller gain. In the first case, we will assume that the matrix B is a full column rank matrix, while the second result assumes that the matrix C is a full row rank matrix.

Theorem 9.2.5 *Assume that the matrix B is full column rank. If there exist symmetric and positive-definite matrix $P \in \mathbb{R}^{n \times n}$ matrices $G \in \mathbb{R}^{n \times n}$ and $L \in \mathbb{R}^{n \times p}$ with the following structures*

$$G = \begin{bmatrix} G_1 & G_2 \\ 0 & G_3 \end{bmatrix}$$

$$L = \begin{bmatrix} L_1 \\ 0 \end{bmatrix}$$

such the following LMI holds:

$$\begin{bmatrix} P - G - G^\top & [GT_r A + LC] T_r^{-1} \\ \star & -P \end{bmatrix} < 0 \tag{9.16}$$

then the closed-loop is exponentially stable and the controller gain is given by:

$$K = G_1^{-1} L_1$$

Proof: Let us assume the LMI (10.17) holds. Using now the fact that the matrix B is full column rank, which implies that there exists a matrix T_r, and the structure of the matrix L, we get:

$$L = \begin{bmatrix} L_1 \\ 0 \end{bmatrix}$$

Using the expression of the controller, ie.: $K = G_1^{-1} L_1$, obtain:

$$L = \begin{bmatrix} G_1 K \\ 0 \end{bmatrix}$$

Based on the structure of G, we can rewrite this expression as follows:

$$L = \begin{bmatrix} G_1 & G_2 \\ 0 & G_3 \end{bmatrix} \begin{bmatrix} \mathbb{I} \\ 0 \end{bmatrix} K = GT_r BK$$

Let $\bar{P} = T_r^\top P T_r$ which gives $P = T_r^{-\top} \bar{P} T_r^{-1}$. Using this the LMI (10.17) becomes:

$$\begin{bmatrix} T_r^{-\top} \bar{P} T_r^{-1} - G - G^\top & \star \\ T_r^{-\top} [A^\top + C^\top K^\top B^\top] T_r^\top G^\top & T_r^{-\top} \bar{P} T_r^{-1} \end{bmatrix} < 0$$

Pre- and post-multiply this by $\text{diag}(T_r^\top, T_r^\top)$ and its transpose we obtain:

$$\begin{bmatrix} P - T_r^\top [G - G^\top] T_r & \star \\ [A^\top + C^\top K^\top B^\top] T_r^\top G^\top T_r & -P \end{bmatrix} < 0$$

Defining M, X and H as follows:

$$M = \begin{bmatrix} P & 0 \\ 0 & -P \end{bmatrix}$$

$$X = \begin{bmatrix} T_r^\top G^\top T_r \\ 0 \end{bmatrix}$$

$$H = \begin{bmatrix} -\mathbb{I} & A + BKC \end{bmatrix}$$

the previous inequality becomes:

$$M + XH + H^\mathsf{T}X^\mathsf{T} < 0$$

Defining ξ by

$$\xi = \begin{bmatrix} x(k+1) \\ x(k) \end{bmatrix}$$

the closed-loop dynamics can be written as follows:

$$H\xi = 0$$

Using now the previous lemma, the inequality $M + XH + H^\mathsf{T}X^\mathsf{T} < 0$ is equivalent to $\xi^\mathsf{T} M\xi < 0$. This inequality is equivalent in turn to:

$$\begin{bmatrix} x^\mathsf{T}(k+1) \; x^\mathsf{T}(k) \end{bmatrix} \begin{bmatrix} P & 0 \\ 0 & -P \end{bmatrix} \begin{bmatrix} x(k+1) \\ x(k) \end{bmatrix} < 0$$

That gives:

$$x^\mathsf{T}(k+1)Px(k+1) - x^\mathsf{T}(k)Px(k) < 0$$

Substituting $x(k+1)$ by its expression for the closed-loop dynamics, we get:

$$A_{cl}^\mathsf{T}PA_{cl} - P < 0$$

which implies that the closed-loop dynamics is exponentially stable.   □

Example 9.2.3 *To show how the results developed for this approach work let us consider a dynamical discrete-time system with the following data:*

$$A = \begin{bmatrix} 0.1 & 0.4 & 0.3 \\ 0.1 & 0.3 & 0.5 \\ 0.7 & 0.7 & 0.2 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.3 & 0.8 \\ 0.3 & 0.5 \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0.4 & 0.3 & 1 \end{bmatrix}$$

It is important to see that the matrix B is full column rank. Therefore, there exists a transformation T_r such that we have:

$$T_r B = \begin{bmatrix} \mathbb{I} \\ 0 \end{bmatrix}$$

To compute the transformation, T_r, let us assume it has the following form:

$$T_r = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ t_7 & t_8 & t_9 \end{bmatrix}$$

Using the previous defing T_r we get:

$$\begin{bmatrix} 0.3 & 0.3 \\ 0.8 & 0.5 \end{bmatrix}\begin{bmatrix} t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0.3 & 0.3 \\ 0.8 & 0.5 \end{bmatrix}\begin{bmatrix} t_4 \\ t_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$t_3, t_6, \cdots, t_9 \quad \text{arbitrary values}$$

Choosing $t_3 = 0, t_6 = 1, t_7 = 0, t_8 = 0$ and $t_9 = 1$, we get:

$$T_r = \begin{bmatrix} -5.5556 & 8.8889 & 0 \\ -3.3333 & 3.3333 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Its inverse is given by:

$$T_r^{-1} = \begin{bmatrix} 0.3 & -0.8 & 0.8 \\ 0.3 & -0.5 & 0.5 \\ 0 & 0 & 1.0 \end{bmatrix}$$

The following program can be used to compute the controller gain.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Static output stabilization %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Nominal case
  clear all;
  yalmip('clear')

% Data
  A=[3 0.3 2; 1 0 1; 0.3 0.6 0.6];
  B=[1 0; 0 1; 1 0];
  C=[1 1 0; 0 1 1];

  n=size(A,1);
  m=size(B,2);
  p=size(C,1);

  % Transfrormation T is given by:
  T=[1 0 0; 1 1 -1; 1 0 -1];

% Variables declaration
  P=sdpvar(n,n,'symmetric');
  G1=sdpvar(n-p,n-p,'full');
  G2=sdpvar(n-p,p,'full');
  G3=sdpvar(p,p,'full');

  L1=sdpvar(n-p,p,'full');
```

```
G=[G1 G2; zeros(p,n-p) G3];
L=[L1; zeros(p,p)];

% LMIs
  F=set(P>0)  % Initialization

  F=F+set([P-G-G'    (G*T*A+L*C)*inv(T);
                ((G*T*A+L*C)*inv(T))'   -P ] < 0)

% Solve
  Sol=solvesdp(F)

% Extract data from the solution
  P=double(P)
  G1=double(G1)
  G2=double(G2)
  G3=double(G3)

  L1=double(L1)
  eig(P)        % check if the eigenvalues are positive

  % compute the controller gain
  K=inv(G1)*L1
  checkset(F)
```

Let us now assume that the output matrix C is full row rank. This means that there exists a transformation, T_l (not unique) such that the following holds:

$$CT_l = \begin{bmatrix} \mathbb{I} & 0 \end{bmatrix}$$

The following theorem gives a similar results when this assumption holds for C.

Theorem 9.2.6 *Assume that the matrix C is full row rank. If there exist symmetric and positive-definite matrix $P \in \mathbb{R}^{n \times n}$ matrices $G \in \mathbb{R}^{n \times n}$ and $L \in \mathbb{R}^{m \times n}$ with the following structures*

$$G = \begin{bmatrix} G_1 & 0 \\ G_2 & G_3 \end{bmatrix}, G_1 \in \mathbb{R}^{p \times p}, G_2 \in \mathbb{R}^{m \times p}, G_3 \in \mathbb{R}^{(n-p) \times (n-p)},$$

$$L = \begin{bmatrix} L_1 & 0 \end{bmatrix}, L_1 \in \mathbb{R}^{m \times p},$$

such the following LMI holds:

$$\begin{bmatrix} P - G - G^\top & \left[G^\top T_l^\top A^\top + L^\top B^\top \right] T_l^{-\top} \\ \star & -P \end{bmatrix} < 0 \tag{9.17}$$

then the closed-loop is exponentially stable and the controller gain is given by:

$$K = L_1 G_1^{-1}$$

Proof: Let us assume the LMI (9.17) holds. Using now the fact that the matrix C is full row rank, which implies that there exists a matrix T_l, and the structure of the matrix L, we get:

$$L = \begin{bmatrix} L_1 & 0 \end{bmatrix}$$

Using the expression of the controller, ie.: $K = L_1 G_1^{-1}$, we obtain:

$$L = \begin{bmatrix} KG_1 & 0 \end{bmatrix}$$

Based on the structure of G, we can rewrite this expression as follows:

$$L = K \begin{bmatrix} \mathbb{I} & 0 \end{bmatrix} \begin{bmatrix} G_1 & 0 \\ G_2 & G_3 \end{bmatrix} = KCT_l G$$

Let $\bar{P} = T_l P T_l^\top$ which gives $P = T_l^{-1} \bar{P} T_l^{-\top}$. Using this the LMI (9.17) becomes:

$$\begin{bmatrix} T_l^{-1} \bar{P} T_l^{-\top} - G - G^\top & \star \\ T_l^{-1} [A + BKC] T_l G & T_l^{-1} \bar{P} T_l^{-\top} \end{bmatrix} < 0$$

Pre- and post-multiply this by $\mathrm{diag}(T_l, T_l)$ and its transpose we obtain:

$$\begin{bmatrix} P - T_l [G - G^\top] T_l^\top & \star \\ [A + BKC] T_l G T_l^\top & -P \end{bmatrix} < 0$$

Defining M, X and H as follows:

$$M = \begin{bmatrix} P & 0 \\ 0 & -P \end{bmatrix}$$

$$X = \begin{bmatrix} T_l G T_l^\top \\ 0 \end{bmatrix}$$

$$H = \begin{bmatrix} -\mathbb{I} & [A + BKC]^\top \end{bmatrix}$$

the previous inequality becomes:

$$M + XH + H^\top X^\top < 0$$

Consider the dual system of (6.11):

$$\hat{x}(k + 1) = [A + BKC]^\top \hat{x}(k)$$

Defining ξ by

$$\xi = \begin{bmatrix} \hat{x}(k + 1) \\ \hat{x}(k) \end{bmatrix}$$

the closed-loop dynamics of the dual system can be written as follows:

$$H\xi = 0$$

Using now the previous lemma, the inequality $M + XH + H^\top X^\top < 0$ is equivalent to $\xi^\top M \xi < 0$. This inequality is equivalent in turn to:

$$\begin{bmatrix} \hat{x}^\top(k + 1) & \hat{x}^\top(k) \end{bmatrix} \begin{bmatrix} P & 0 \\ 0 & -P \end{bmatrix} \begin{bmatrix} \hat{x}(k + 1) \\ \hat{x}(k) \end{bmatrix} < 0$$

That gives:

$$\hat{x}^{\top}(k+1)P\hat{x}(k+1) - \hat{x}^{\top}(k)P\hat{x}(k) < 0$$

Substituting $\hat{x}(k+1)$ by its expression for the closed-loop dynamics, we get:

$$A_{cl}PA_{cl}^{\top} - P < 0$$

which implies that the closed-loop dynamics of the dual system is exponentially stable.   □

Example 9.2.4 *In this example, we consider the same system as in the last example.*
It is important to see that the matrix C is full row rank. Therefore, there exists a transformation T_l such that we have:

$$CT_l = \begin{bmatrix} \mathbb{I} & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

To compute the transformation, T_l, let us assume it has the following form:

$$T_r = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ t_7 & t_8 & t_9 \end{bmatrix}$$

Using the previous definition T_l we obtain this transformation. We can also use Matlab to compute such transformation matrix. This approach will be used in this example.
The following program can be used to compute the controller gain.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Static output stabilization %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Nominal case
  clear all;
  yalmip('clear')

% Data
  A=[3 0.3 2; 1 0 1; 0.3 0.6 -0.6];
  B=[1 0; 0 1; 1 0];
% C=[1 1 0; 0 1 1];
  C=[1 1 0];

  n=size(A,1);
  m=size(B,2);
  p=size(C,1);

  % Transfrormation T is given by:
  T=[0 0 1; 1 0 -1; -1 1 1];
  T=[C'*inv(C*C') null(C)];
```

```
% Variables declaration
  P=sdpvar(n,n,'symmetric');
  G1=sdpvar(p,p,'full');
  G2=sdpvar(m,p,'full');
  G3=sdpvar(n-p,n-p,'full');

 L1=sdpvar(m,p,'full');

 G=[G1 zeros(p,n-p); G2 G3];
 L=[L1 zeros(m,m)];

% LMIs
  F=set(P>0)  % Initialization

  F=F+set([P-G-G'   (inv(T)*(A*T*G+B*L))';
                   inv(T)*(A*T*G+B*L)   -P ] < 0)

% Solve
  Sol=solvesdp(F)

% Extract data from the solution
  P=double(P)
  G1=double(G1)
  G2=double(G2)
  G3=double(G3)

  L1=double(L1)
  eig(P)        % check if the eigenvalues are positive

 % compute the controller gain
 K=L1*inv(G1)
 checkset(F)
```

If we run this program, we get:

$$P = \begin{bmatrix} 2.2747 & -0.5548 & -0.7761 \\ -0.5548 & 0.4246 & 0.4775 \\ -0.7761 & 0.4775 & 0.7107 \end{bmatrix}$$

$$G_1 = \begin{bmatrix} 1.6459 \end{bmatrix}, G_2 = \begin{bmatrix} -0.5169 \\ -0.7761 \end{bmatrix}, G_3 = \begin{bmatrix} 0.4392 & 0.4795 \\ 0.4626 & 0.7099 \end{bmatrix},$$

$$L_1 = \begin{bmatrix} -1.6568 \\ -0.4629 \end{bmatrix}$$

which gives the the following gain:

$$K = \begin{bmatrix} -1.0066 \\ -0.2812 \end{bmatrix}$$

It can be seen that all the poles of the closed-loop dynamics are all inside the unit circle.

Let us now focus on the design of the output feedback controller. In this we assume that the dynamics of the system is described by:

$$\begin{cases} x(k+1) = [A + \Delta A(k)] \, x(k) + [B + \Delta B(k)] \, u(k) \\ y(k) = [C + \Delta C(k)] \, x(k) \end{cases}$$

where A, B and C keep the same definitions while $\Delta A(k)$, $\Delta B(k)$ and $\Delta C(k)$ are given by:

$$\begin{aligned} \Delta A(k) &= D_A F_A(k) E_A \\ \Delta A(k) &= D_B F_B(k) E_B \\ \Delta A(k) &= D_C F_C(k) E_C \end{aligned}$$

with $F_A^\top(k) F_A(k) \le \mathbb{I}$, $F_B^\top(k) F_B(k) \le \mathbb{I}$ and $F_C^\top(k) F_C(k) \le \mathbb{I}$.

The structure of the controller we will use here is given by:

$$\begin{cases} \hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L\left[Cx(k) - C\hat{x}(k)\right] \\ u(k) = K\hat{x}(k) \end{cases}$$

where K and L are the controller gain to be determined.

Remark 9.2.3 *It is important to notice that the dynamics of the controller can be rewritten as follows:*

$$\begin{cases} \hat{x}(k+1) = [A + BK - LC] \, \hat{x}(k) + Ly(k) \\ u(k) = K\hat{x}(k) \end{cases}$$

which corresponds to the dynamic output feedback controller's dynamics.

Let $e(k)$ be defined by:

$$e(k) = x(k) - \hat{x}(k)$$

which implies:

$$\begin{aligned} e(k+1) &= x(k+1) - \hat{x}(k) \\ &= Ax(k) + Bu(k) - A\hat{x}(k) - Bu(k) - LC\left[x(k) - \hat{x}(k)\right] \\ &= Ae(k) - LCe(k) \\ &= [A - LC] \, e(k) \end{aligned}$$

From the other side we have:

$$\begin{aligned} x(k+1) &= Ax(k) + BK\hat{x}(k) \\ &= Ax(K) + BK\left[x(k) - e(k)\right] \\ &= [A + BK] \, x(k) - BKe(k) \end{aligned}$$

Now if we define $\eta(k)$ by:

$$\eta(k) \;=\; \begin{bmatrix} x(k) \\ e(k) \end{bmatrix}$$

the augmented dynamics give:

$$\eta(k+1) \;=\; \tilde{A}\eta(k)$$

where

$$\tilde{A} = \begin{bmatrix} A + BK & -BK \\ 0 & A - LC \end{bmatrix}$$

The nominal closed-loop dynamics is stable if there exists a symmetric and and positive-definite matrix, $\tilde{P}$, such that the following holds:

$$\begin{bmatrix} -\tilde{P} & \tilde{A}^\top \tilde{P} \\ \tilde{P}\tilde{A} & -\tilde{P} \end{bmatrix} < 0$$

Now if we let:

$$\tilde{P} = \begin{bmatrix} P & 0 \\ 0 & Q \end{bmatrix}$$

and using the expression of $\tilde{A}$ we get:

$$\begin{bmatrix} -P & 0 & [A+BK]^\top P & 0 \\ 0 & -Q & -[BK]^\top P & [A-LC]^\top Q \\ P[A+BK] & -PBK & -P & 0 \\ 0 & Q[A-LC] & 0 & -Q \end{bmatrix} < 0$$

This condition can be rewritten as follows:

$$\begin{bmatrix} -P & 0 & [A+BK]^\top P & 0 \\ 0 & -Q & -[BK]^\top P & [A-LC]^\top Q \\ P[A+BK] & -PBK & -P & 0 \\ 0 & Q[A-LC] & 0 & -Q \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -PBK & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -[BK]^\top P & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} < 0$$

Noticing that:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -PBK & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -PB \\ 0 \end{bmatrix} \begin{bmatrix} 0 & K & 0 & 0 \end{bmatrix}$$

Using Lemma 9.1.2, we get:

$$
\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -PBK & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \le \varepsilon \begin{bmatrix} 0 \\ 0 \\ -PB \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & -[PB]^\top & 0 \end{bmatrix}
$$

$$
+ \varepsilon^{-1} \begin{bmatrix} 0 \\ K^\top \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & K & 0 & 0 \end{bmatrix}
$$

for $\varepsilon > 0$.

Using Schur complement, the previous condition will hold if this holds:

$$
\begin{bmatrix} -P & 0 & [A+BK]^\top P & 0 \\ 0 & -Q + \varepsilon^{-1} K^\top K & 0 & [A-LC]^\top Q \\ P[A+BK] & 0 & -P + \varepsilon PB[PB]^\top & 0 \\ 0 & Q[A-LC] & 0 & -Q \end{bmatrix} < 0
$$

Using again Schur complement this condition will hold if the following holds:

$$
\begin{bmatrix} J_P & 0 \\ 0 & J_Q \end{bmatrix} < 0
$$

where

$$
J_P = -P + [A+BK]^\top P \left[P - \varepsilon PBB^\top P \right]^{-1} P[A+BK]
$$
$$
J_Q = -Q + \varepsilon^{-1} K^\top K + [A-LC]^\top Q[A-LC]
$$

If the following holds, the closed-loop dynamics is stable for given K and L:

$$
\begin{bmatrix} -P & [A+BK]^\top P \\ P[A+BK] & -P + \varepsilon PBB^\top P \end{bmatrix} < 0
$$

$$
\begin{bmatrix} -Q + \varepsilon^{-1} K^\top K & [A-LC]^\top Q \\ Q[A-LC] & -Q \end{bmatrix} < 0
$$

To design the gain parameters K and L, first of all, let $X = P^{-1}$ and pre- and post-multiply the first inequality by $\operatorname{diag}[X, X]$, we get:

$$
\begin{bmatrix} -X & X[A+BK]^\top \\ [A+BK]X & -X + \varepsilon BB^\top \end{bmatrix} < 0
$$

If we define $Y = KX$ we get the following LMI that can compute ε and the gain K:

$$
\begin{bmatrix} -X & XA^\top + Y^\top B^\top \\ AX + BY & -X + \varepsilon BB^\top \end{bmatrix} < 0
$$

With the ε and the gain K computed by the previous LMI and we define $W = QL$ we get the LMI that permits to compute the gain L:

$$
\begin{bmatrix} -Q + \varepsilon^{-1} K^\top K & A^\top Q - C^\top W^\top \\ QA - WC & -Q \end{bmatrix} < 0
$$

Theorem 9.2.7 *There exists a stabilizing dynamic output feedback if there exist symmetric and positive-definite matrices X and Q, matrices Y and W and a scalar ε such that the following LMIs holds:*

$$\begin{bmatrix} -X & XA^\top + Y^\top B^\top \\ AX + BY & -X + \varepsilon BB^\top \end{bmatrix} < 0 \tag{9.18}$$

$$\begin{bmatrix} -Q + \varepsilon^{-1} K^\top K & A^\top Q - C^\top W^\top \\ QA - WC & -Q \end{bmatrix} < 0 \tag{9.19}$$

The controller gain are given by:

$$K = YX^{-1}$$
$$L = Q^{-1}W$$

Remark 9.2.4 *The LMIs of this theorem are independent and therefore, we can solve the first one to get the gain K and the scalar ε. With these two parameters, we can solve the second LMI to get the gain L of the controller.*

9.3 $\mathscr{H}_\infty$ Stabilization

Previously, we treated the stabilization problem of systems without external disturbances, but practically we are always facing external disturbances that we can not neglect in the design phase. If these disturbances can be modeled by a gaussian process (expectation is null and the variance is constant), we can recourse to the linear quadratic gaussian problem that gives a solution to the problem by solving a Ricatti equation. This assumption on the external signal is in general very hard to verify and therefore we need another alternate to solve such problem. $\mathscr{H}_\infty$ control technique has been propose to overcome this with less assumptions. In fact, the $\mathscr{H}_\infty$ control theory requires only that the external disturbance has finite energy, which is always satisfied in practice.

Practical systems are always affected by external disturbances that may degrade the system performance. To overcome the negative effects of the external disturbances that are supposed to have finite energy or finite average power, the $\mathscr{H}_\infty$ technique was proposed. Contrary to optimal control which handles the case of external disturbances that must satisfy some special assumptions, $\mathscr{H}_\infty$ control requires only that the external disturbance have finite energy or finite average power. $\mathscr{H}_\infty$ control is a way minimize the worst-case gain of the system. This optimization problem can be stated as a game optimization problem with two players; the designer, who is seeking a controller that minimizes the gain, and nature which seeks an external disturbance that maximizes the gain.

The goal of $\mathscr{H}_\infty$ control is to seek a controller (state-feedback, dynamic output feedback, static output feedback) that minimizes the $\mathscr{H}_\infty$-norm of the system

closed-loop transfer matrix between the controlled output $z(k)$ and the external disturbance $w(k)$ that belongs to $\mathscr{L}_2[0, T]$, with $\Delta A(k) = \Delta B(k) \equiv 0$, that is,

$$\|G_{zw}\|_\infty = \sup_{\|w(k)\|_{2,[0,T]} \neq 0} \frac{\|z(k)\|_{2,[0,T]}}{\|w(k)\|_{2,[0,T]}}, \tag{9.20}$$

where $\|G_{zw}\|$ is the transfer matrix between the output $z(k)$ and the external disturbance $w(k)$.

The $\mathscr{H}_\infty$ control problem can be defined on either finite or infinite horizon ($T \rightarrow \infty$). In the rest of this section, we develop the finite horizon case. To get the infinite horizon case, we let T go to infinity with the appropriate assumptions.

The $\mathscr{H}_\infty$-norm cost function (9.20) is not acceptable as an objective function since this cost depends on the controller; that is, the supremum makes this function independent of a particular disturbance input.

A quadratic objective function that yields tractable solutions of the differential game is referred to as a suboptimal solution to the $\mathscr{H}_\infty$ optimization control problem. It can be obtained by considering the following bound on the closed-loop $\mathscr{H}_\infty$ norm:

$$\|G_{zw}\|_\infty = \sup_{\|w(k)\|_{2,[0,T]} \neq 0} \frac{\|z(k)\|_{2,[0,T]}}{\|w(k)\|_{2,[0,T]}} < \gamma,$$

where γ is referred to as the performance bound.

This suboptimal controller must also satisfy the following bound:

$$\|G_{zw}\|_\infty^2 = \sup_{\|w(k)\|_{2,[0,T]} \neq 0} \frac{\|z(k)\|_{2,[0,T]}^2}{\|w(k)\|_{2,[0,T]}^2} < \gamma^2. \tag{9.21}$$

To make the supremum satisfy this inequality, the following should hold:

$$\frac{\|z(k)\|_{2,[0,T]}^2}{\|w(k)\|_{2,[0,T]}^2} \leq \gamma^2 - \varepsilon^2, \tag{9.22}$$

which gives

$$\|z(k)\|_{2,[0,T]}^2 - \gamma^2 \|w(k)\|_{2,[0,T]}^2 \leq -\varepsilon^2 \|w(k)\|_{2,[0,T]}^2. \tag{9.23}$$

Note that the satisfaction of this inequality for all disturbance inputs and some ε is equivalent to the bound on the closed-loop $\mathscr{H}_\infty$ norm (9.22). Therefore, the left-hand side of (9.23) can be used as an objective function of our $\mathscr{H}_\infty$ optimization control problem. Therefore, the optimization problem we should solve is given by

$$\min_{u(.)} \left[\sum_{k=0}^{T} \left[z^\top(k) z(k) - \gamma^2 w^\top(k) w(k) \right] \right],$$

subject to the free nominal system.

When the uncertainties are present in the dynamics, the robust $\mathscr{H}_\infty$ control consists of making the gain from the exogenous $w(t)$ to the controlled output $z(t)$, (l_2-gain) less than or equal to $\gamma > 0$, that is,

$$\sum_{k=0}^{T} \|z(k)\|^2 \leq \gamma^2 \sum_{k=0}^{T} \|w(k)\|^2,$$

for all $T > 0$ and for all admissible uncertainties. Note that T can be chosen to be infinite.

Mathematically the robust $\mathcal{H}_\infty$ control problem can be stated as follows. Given a positive γ, find a controller that robustly stabilizes the system and guarantees

$$\sup_{w(.)\in l_2[0,\infty]} \frac{\|z_k\|_2^2}{\|w_k\|_2^2} \leq \gamma^2$$

for all admissible uncertainties.

Let us consider the following class of discrete-time linear systems:

$$\begin{cases} x(k + 1) = Ax(k) + Bu(k) + Ew(k) \\ z(k) = Cx(k) + Du(k) + Fw(k) \\ y(k) = Gx(k) + Hw(k) \end{cases} \qquad (9.24)$$

where $x(k) \in \mathbb{R}^n$ is the state vector and $u(k)$ is the control input; $w(k) \in \mathbb{R}^l$ is the disturbance input which belongs to $l_2[0,\infty)$ ($l_2[0,\infty)$ is the space of square summable infinite sequence and for $w = \{w(k)\} \in l_2[0,\infty)$, its norm is given by $\|w\|_2 = \sqrt{\sum_{k=0}^{\infty} |w(k)|^2}$.); $y(k) \in \mathbb{R}^m$ is the measurement output and $z(k)$ is the objective signal to be attenuated, A, B, C, D, E, F, G and H ar real known with appropriate dimensions.

In this section, the state-feedback and output-feedback control will be considered respectively for system (9.24) when the state variables are assumed to be fully or partly available for feedback. The state-feedback controller has the following form

$$u(k) = Kx(k) \qquad (9.25)$$

and the output-feedback controller is assumed to be dynamic and has the following structure:

$$\begin{cases} x_c(k + 1) = A_d x_c(k) + B_d y(k) \\ u(k) = C_d x_c(k) + D_d y(k) \end{cases} \qquad (9.26)$$

Accordingly, the closed-loop system (1) with (9.25) is given by

$$\begin{cases} x(k + 1) = \hat{A}x(k) + \hat{E}w(k) \\ z(k) = \hat{C}x(k) + \hat{F}w(k) \end{cases} \qquad (9.27)$$

where

$$\begin{aligned} \hat{A} &= A + BK, \\ \hat{C} &= C + DK, \\ \hat{E} &= E, \\ \hat{F} &= F. \end{aligned}$$

Likewise, for the output-feedback case, by defining $\xi(k) \triangleq \left[x^\top(k)\ x_c^\top(k) \right]^\top$, the corresponding closed-loop system resulted from (1) and (9.26) is given by

$$\begin{cases} \xi(k+1) = \tilde{A}\xi(k) + \tilde{E}w(k) \\ \quad z(k) = \tilde{C}\xi(k) + \tilde{F}w(k) \end{cases} \tag{9.28}$$

where

$$\tilde{A} = \begin{bmatrix} A + BD_dG & BC_d \\ B_dG & A_d \end{bmatrix},$$

$$\tilde{E} = \begin{bmatrix} E + BD_dH \\ B_dH \end{bmatrix}$$

$$\tilde{C} = \begin{bmatrix} C + DD_dG & DC_d \end{bmatrix},$$

$$\tilde{F} = [F + DD_dH]$$

Obviously, the resulting system (9.27) and (9.28) are also discrete-time. Now, for more precise description of the main objective of this section, we also introduce the following definition for the underlying systems and for more details on this concept we the reader is referred to [2] and the references therein.

Definition 9.3.1 *Given a scalar $\gamma > 0$, system (9.27) or (9.28) is said to be stable and has an $\mathscr{H}_\infty$ noise attenuation performance index γ if it is stable and under zero initial condition, $\|z\|_2 < \gamma \|w\|_2$ holds for all nonzero $w(k) \in l_2[0, \infty)$.*

Therefore, the purpose of this session is to design a $\mathscr{H}_\infty$ state-feedback and output-feedback controller, respectively, such that the resulting closed-loop systems (9.27) and (9.28) are stable and has a prescribed $\mathscr{H}_\infty$ performance index.

Let us firstly develop the $\mathscr{H}_\infty$ performance criterion for the closed-loop systems (9.27) and (9.28). To this end, we establish a general closed-loop system model as follows:

$$\begin{cases} x(k+1) = \bar{A}x(k) + \bar{E}w(k) \\ \quad z(k) = \bar{C}x(k) + \bar{F}w(k) \end{cases} \tag{9.29}$$

where state vector $x(k) \in \mathbb{R}^l$ $(l \geq n)$, and the construction of system matrix $\bar{A}$, $\bar{E}, \bar{C}$, and $\bar{F}$ are different when applying the state-feedback or output-feedback controllers, respectively. Now, the following lemma gives a sufficient condition for a bounded $\mathscr{H}_\infty$ performance criterion (i.e., the so-called bounded real lemma (BRL)) for system (9.29).

Lemma 9.3.1 *Consider system (9.29) and let $\gamma > 0$ be a given constant. If there exists a matrix $P > 0$ such that*

$$\begin{bmatrix} \bar{A}^\top P\bar{A} - P & \bar{A}^\top P\bar{E} & \bar{C}^\top \\ \star & -\gamma^2 I + \bar{E}^\top P\bar{E} & \bar{F}^\top \\ \star & \star & -\mathbb{I} \end{bmatrix} < 0 \tag{9.30}$$

then system (9.29) is stable with a $\mathscr{H}_\infty$ performance index γ.

Proof. Notice that if the LMI (9.30) holds, then it results $\bar{A}^\top P\bar{A} - P < 0$ holds too and this implies that the closed-loop dynamics is stable.

Let us now focus on the proof of the $\mathscr{H}_\infty$ performance. For this purpose, consider the system (9.29) and construct a Lyapunov function as

$$V(x_k) = x_k^\top P x_k, \tag{9.31}$$

where P satisfy (9.30). Then, one has

$$\Delta V(x_k, k) \triangleq V(x_{k+1}, k+1) - V(x_k, k)$$
$$= x_{k+1}^\top P x_{k+1} - x_k^\top P x_k \tag{9.32}$$

Using now the expression of $x(k+1)$, we get:

$$\Delta V(x_k, k) = x(k)\left[A^\top P A - P\right]x(k) + x^\top(k)A^\top P E w(k) + w^\top(k)E^\top P A x(k)$$
$$+ w^\top(k)E^\top P E w(k) \tag{9.33}$$

Now, to establish the H_∞ performance for the system, consider the following performance index:

$$J \triangleq \sum_{k=0}^{\infty}\left[z^\top(k)z(k) - \gamma^2 w^\top(k)w(k)\right]$$

Under zero initial condition, $V(x(k))\,|_{k=0}= 0$, and we have

$$J \leq \sum_{k=0}^{\infty}\left[z^\top(k)z(k) - \gamma^2 w^\top(k)w(k) + \Delta V\right]$$
$$= \sum_{k=0}^{\infty} \zeta^\top(k)\Phi\zeta(k)$$

where $\zeta(k) \triangleq \left[\, x^\top(k)\ w^\top(k)\,\right]^\top$ and

$$\Phi \triangleq \begin{bmatrix} \bar{A}^\top P \bar{A} - P + \bar{C}^\top\bar{C} & \bar{A}^\top P \bar{E} + \bar{C}^\top\bar{F} \\ \star & -\gamma^2\mathbb{I} + \bar{E}^\top P \bar{E} + \bar{F}^\top\bar{F} \end{bmatrix}$$

Note that $\Phi < 0$ is equivalent to:

$$\begin{bmatrix} \bar{A}^\top P \bar{A} - P & \bar{A}^\top P \bar{E} \\ \star & -\gamma^2 I + \bar{E}^\top P \bar{E} \end{bmatrix} + \begin{bmatrix} \bar{C}^\top \\ \bar{F}^\top \end{bmatrix}\mathbb{I}^{-1}\begin{bmatrix} \bar{C} & \bar{F} \end{bmatrix} < 0.$$

By Schur complement, one has

$$\begin{bmatrix} \bar{A}^\top P \bar{A} - P & \bar{A}^\top P \bar{E} & \bar{C}^\top \\ \star & -\gamma^2 I + \bar{E}^\top P \bar{E} & \bar{F}^\top \\ \star & \star & -\mathbb{I} \end{bmatrix} < 0.$$

Therefore, inequalities (9.30) guarantees $\Phi < 0$, i.e., $J < 0$ which means that $\|e\|_2 < \gamma\,\|w\|_2$, this completes the proof. $\qquad\square$

9.3.1 State-Feedback Control

In this subsection, we design a state-feedback controller of the form (9.25). The following theorem presents a sufficient condition of the existence of the controllers for system (9.24).

First of all notice that the LMI (9.30) can be rewritten as follows for any invertible matrix G:

$$\Phi = \eta_1^\top \Phi_1 \eta_1$$

with

$$\eta_1 = \begin{bmatrix} G^{-1} & 0 & 0 \\ G^{-1}A & 0 & 0 \\ 0 & \mathbb{I} & 0 \\ 0 & 0 & \mathbb{I} \end{bmatrix}$$

$$\Phi_1 = \begin{bmatrix} \bar{A}G + G^\top \bar{A}^\top - G^\top PG & G^\top \bar{A}^\top - G & 0 & G^\top \bar{C}^\top \\ \bar{A}G - G^\top & G^\top PG - G^\top - G & G^\top P\bar{E} & 0 \\ 0 & \bar{E}PG & -\gamma^2\mathbb{I} + \bar{E}^\top P\bar{E} & \bar{F}^\top \\ \bar{C}G & 0 & \bar{F} & -\mathbb{I} \end{bmatrix}$$

Similarly, Φ_1 can be decomposed in turn as follows:

$$\Phi_1 = \eta_2^\top \Phi_2 \eta_2$$

with

$$\eta_2 = \begin{bmatrix} \mathbb{I} & 0 & 0 & 0 \\ 0 & \mathbb{I} & 0 & 0 \\ 0 & G & 0 & 0 \\ 0 & 0 & \mathbb{I} & 0 \\ 0 & 0 & 0 & \mathbb{I} \end{bmatrix}$$

$$\Phi_2 = \begin{bmatrix} \bar{A}G + G^\top \bar{A}^\top - G^\top PG & G^\top \bar{A}^\top - G & 0 & 0 & G^\top \bar{C}^\top \\ \bar{A}G - G^\top & -2G - 2G^\top - G^\top - G\ G^\top + \mathbb{I} & 0 & 0 \\ 0 & G + \mathbb{I} & P - 2\mathbb{I} & P\bar{E} & 0 \\ 0 & 0 & \bar{E}^\top P & -\gamma^2\mathbb{I} + \bar{E}^\top P\bar{E} & \bar{F}^\top \\ \bar{C}G & 0 & 0 & \bar{F} & -\mathbb{I} \end{bmatrix}$$

In turn we can decompose Φ_2 as follows:

$$\Phi_2 = \eta_3^\top \Phi_3 \eta_3$$

with

$$\eta_3 = \begin{bmatrix} \mathbb{I} & 0 & 0 & 0 & 0 \\ 0 & \mathbb{I} & 0 & 0 & 0 \\ 0 & 0 & \mathbb{I} & 0 & 0 \\ 0 & 0 & 0 & \mathbb{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbb{I} \\ G & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Phi_3 = \begin{bmatrix} J_1 & \star & \star & \star & \star & \star \\ \bar{A}G - G^\top & J_2 & \star & \star & \star & \star \\ 0 & G + \mathbb{I} & P - 2\mathbb{I} & \star & \star & \star \\ 0 & 0 & \bar{E}^\top P & -\gamma^2\mathbb{I} + \bar{E}^\top P\bar{E} & \star & \star \\ \bar{C}G & 0 & 0 & \bar{F} & -\mathbb{I} & \star \\ G + \mathbb{I} & 0 & 0 & 0 & 0 & -\bar{E}^\top P\bar{E} - 2\mathbb{I} \end{bmatrix}$$

with $J_1 = \bar{A}G + G^\top \bar{A}^\top - G^\top - G$ and $J_2 = -2G - 2G^\top$.

Using now all these decomposition we get:

$$\Phi = \eta_3^\top \eta_2^\top \eta_1^\top \Phi_3 \eta_1 \eta_2 \eta_3$$

The LMI, $\Phi < 0$ holds if $\Phi_3 < 0$ holds.

Using the now the expressions of $\bar{A}$, $\bar{E}$, $\bar{C}$, and $\bar{F}$ and letting $Y = KG$, the LMI $\Phi_3 < 0$ becomes:

$$\Phi_3 = \begin{bmatrix} \bar{J}_1 & \star & \star & \star & \star & \star \\ AG + BY - G^\top & J_2 & \star & \star & \star & \star \\ 0 & G + \mathbb{I} & P - 2\mathbb{I} & \star & \star & \star \\ 0 & 0 & E^\top P & -\gamma^2 \mathbb{I} + E^\top PE & \star & \star \\ CG + DY & 0 & 0 & F & -\mathbb{I} & \star \\ G + \mathbb{I} & 0 & 0 & 0 & 0 & -E^\top PE - 2\mathbb{I} \end{bmatrix} < 0$$

with $\bar{J}_1 = AG + G^\top A^\top + BY + Y^\top B - G^\top - G$ and $J_2 = -2G - 2G^\top$.

Based on this development, we get the following result.

Theorem 9.3.1 *Let $\gamma > 0$ be a positive scalar. There exists a controller (9.25) such that the resulting closed-loop system (9.27) is stable and achieves a prescribed $\mathcal{H}_\infty$ performance index γ if there exists a symmetric and positive-definite matrix $P > 0$, a nonsingular matrix G and a matrix Y such that the following LMI holds:*

$$\begin{bmatrix} \bar{J}_1 & \star & \star & \star & \star & \star \\ AG + BY - G^\top & J_2 & \star & \star & \star & \star \\ 0 & G + \mathbb{I} & P - 2\mathbb{I} & \star & \star & \star \\ 0 & 0 & E^\top P & -\gamma^2 \mathbb{I} + E^\top PE & \star & \star \\ CG + DY & 0 & 0 & F & -\mathbb{I} & \star \\ G + \mathbb{I} & 0 & 0 & 0 & 0 & -E^\top PE - 2\mathbb{I} \end{bmatrix} < 0$$

with $\bar{J}_1 = AG + G^\top A^\top + BY + Y^\top B - G^\top - G$ and $J_2 = -2G - 2G^\top$. Moreover, the controller gain is given by

$$K = YG^{-1} \tag{9.34}$$

Remark 9.3.1 *The LMI of this theorem can be solved easily by convex optimization softwares. Also, the optimal $\mathcal{H}_\infty$ performance index γ can be obtained by by letting $\delta = \gamma^2$ in the LMI of the theorem and solve the corresponding LMI by minimizing δ. The corresponding disturbance rejection is $\gamma = \sqrt{\delta}$.*

Now, we give the following numerical example to verify the results obtained in the above Theorem.

Example 9.3.1 *Consider a system of the structure (9.24) with the following data:*

$$A = \begin{bmatrix} 1.00 & -1.25 \\ 2.50 & 2.50 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.50 \\ 0.10 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

$$D = 0.8,$$

$$E = \begin{bmatrix} 0.08 \\ 0.10 \end{bmatrix},$$

$$F = 0.6.$$

Our purpose in this example is to design a state-feedback $\mathcal{H}_\infty$ controller of the form of (9.25) such that the resulting closed-loop system is stable with an optimal $\mathcal{H}_\infty$ performance index.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% State feedback H-8 control
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Uncertainl case
clear all;
yalmip('clear')

% Data
A=[1 0 -1; 0 -2 1; 2 -1 -2];
B=[0 1; 2 0; 1 1];
E=[0.1; 0.2; -0.1]

C=[0.1 0.2; 0.3 0.5; 1 -1];
D=[0.1 0.4; -0.2 -0.3];
F=[0.3; 0.2; 0.1];


n=size(A,1);
m=size(B,2);

% Variables
P=sdpvar(n,n,'symmetric'); % declaration
Y=sdpvar(m,n,'full');
G=sdpvar(n,n,'full');

% LMI
F=set(P>0)  % Initialization

J1=A*G+G'*A'+B*Y+Y'*B'-G-G';
J2=2*G-2*G';
J3=-delta*eye(1)+E'*P*E;
```

```
J4=-E'*P*E-2*eye(n);
F=F+set([J1 G'*A'+Y'*B'-G zeros(n,n) zeros(n,1) G'*C'+Y'*D' G'+eye(n);
A*G+B*Y-G' J2  G'+eye(n)   zeros(n,1)   zeros(n,m) zeros(n);
zeros(n) G+eye(n) P-2*eye(n)  P*E     zeros(n,m) zeros(n);
zeros(1,n) zeros(1,n) E'*P   J3  F'   zeros(1,n)
C*G+D*Y zeros(m,n) zeros(m,n)  F -eye(m)   zeros(m,n) ;
G+eye(n) zeros(n) zeros(n) zeros(n,1)   zeros(,m) J4] < 0)

% Solve
Sol=solvesdp(F)

% Extract data from the solution
P=double(P)
Y=double(Y)
G=double(G)

eig(P)       % check if the eigenvalues are positive
eig(G)

% compute the controller gain
K=Y*inv(G)
checkset(F)
```

By solving the LMI of Theorem 9.3.1, the optimal $\mathcal{H}_\infty$ performance index is given $\gamma^\star =$, and the corresponding controller gains are:

$$K = \begin{bmatrix} -6.05 & -2.57 \end{bmatrix}$$

Furthermore, applying the above controllers and giving two possible system modes evolution, the state response of the closed-loop system are shown in Figures 1-2 under given initial condition $x_0 = [-1.2\,0.6]^\top$.

9.3.2   Static Output Feedback $\mathcal{H}_\infty$ Control

Let us consider the design of the static output feedback controller with the following form:

$$u(k) = Ky(k)$$

where $y(k)$ is the measured output and $K \in \mathbb{R}^{m\times p}$ is the controller gain to be computed.

Lemma 9.3.2 *Consider a system of the form (9.24) with $x(0) = 0$. If there exists a Lyapunov function $V(x) = x^\top Px$, with P a symmetric and positive-definite matrix, satisfying the following:*

$$\Delta V = V(x(k + 1)) - V(x(k)) < \gamma^2 \|w(k)\|^2 - \|z(k)\|^2, \forall k$$

then the $\mathcal{H}_\infty$ performance is satisfied. Moreover, the closed-loop system is stable.

Theorem 9.3.2 *Assume that the matrix B is full column rank. If there exist a symmetric and positive-definite matrix P and matrices $S \in \mathbb{R}^{n \times n}$ and $L \in \mathbb{R}^{n \times p}$ with the following structures*

$$S = \begin{bmatrix} S_1 & S_2 \\ 0 & S_3 \end{bmatrix}, L = \begin{bmatrix} L_1 \\ 0 \end{bmatrix}$$

such that the following LMI holds:

$$\begin{bmatrix} J & \star & \star & \star \\ 0 & -\mathbb{I} & \star & \star \\ \left[[S T_r A + LG] T_r^{-1}\right]^\top & C T_r^{-1} & -P & \star \\ [S T_r E + LH]^\top & F & 0 & -\gamma^2 \mathbb{I} \end{bmatrix} < 0 \qquad (9.35)$$

where $J = P - S - S^\top$, then the system (9.24) is stable under the static output feedback controller and satisfies the $\mathscr{H}_\infty$ performance, i.e.: $\|z\|_2^2 < \gamma^2 \|w\|_2^2$. The stabilizing controller gain K is given by:

$$K = S_1^{-1} L_1$$

Proof: To prove this theorem, let us first of all assume that the LMI (9.35) holds. Let us alos define $\bar{P} = T_r^\top P T_r$, which is equivalent to $P = T_r^{-1} \bar{P} T_r^{-1}$. Using this, the LMI (9.35) becomes:

$$\begin{bmatrix} \bar{J} & \star & \star & \star \\ 0 & -\mathbb{I} & \star & \star \\ \left[[S T_r A + LG] T_r^{-1}\right]^\top & C T_r^{-1} & -P & \star \\ [S T_r E + LH]^\top & F & 0 & -\gamma^2 \mathbb{I} \end{bmatrix} < 0$$

where $\bar{J} = T_r^{-1} \bar{P} T_r^{-1} - S - S^\top$.

Pre- and post-multiply this inequality by diag $[T_r^\top, \mathbb{I}, T_r^\top, \mathbb{I}]$ and its transpose, we get:

$$\begin{bmatrix} \bar{J} & \star & \star & \star \\ 0 & -\mathbb{I} & \star & \star \\ [T_r^\top [S T_r A + LG]]^\top & C & -P & \star \\ [T^t op_r [S T_r E + LH]]^\top & F & 0 & -\gamma^2 \mathbb{I} \end{bmatrix} < 0$$

where $\bar{J} = \bar{P} - T_r^\top [S + S^\top] T_r$.

Using now the structures of the matrices S and L, we get:

$$\begin{aligned} L &= \begin{bmatrix} L_1 \\ 0 \end{bmatrix} = \begin{bmatrix} S_1 K \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} S_1 & S_2 \\ 0 & S_3 \end{bmatrix} \begin{bmatrix} \mathbb{I} \\ 0 \end{bmatrix} K = S T_r B K \end{aligned}$$

which implies:

$$\begin{bmatrix} \bar{J} & \star & \star & \star \\ 0 & -\mathbb{I} & \star & \star \\ [T_r^\top [ST_rA + LG]]^\top & C & -P & \star \\ [T^t op_r [ST_rE + LH]]^\top & F & 0 & -\gamma^2\mathbb{I} \end{bmatrix} < 0$$

Defining $\mathcal{P}$, $\mathcal{X}$ and $\mathcal{H}$ as follows:

$$\mathcal{P} = \begin{bmatrix} P & 0 & 0 & 0 \\ 0 & \mathbb{I} & 0 & 0 \\ 0 & 0 & -P & 0 \\ 0 & 0 & 0 & -\gamma^2\mathbb{I} \end{bmatrix},$$

$$\mathcal{X} = \begin{bmatrix} T_r^\top ST_r & 0 \\ 0 & \mathbb{I} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\mathcal{H} = \begin{bmatrix} -\mathbb{I} & 0 & A + BKC & E + BKC \\ 0 & -\mathbb{I} & C & F \end{bmatrix}$$

this inequality becomes $\mathcal{P} + \mathcal{X}\mathcal{H} + \mathcal{H}^\top\mathcal{X}^\top < 0$.

If we define, $\xi(k)$ by

$$\xi(k) = \begin{bmatrix} x(k+1) \\ z(k) \\ x(k) \\ w(k) \end{bmatrix}$$

the closed-loop dynamics can be rewritten as follows:

$$\mathcal{H}\xi(k) = 0$$

Using Finsler's lemma, $\mathcal{P} + \mathcal{X}\mathcal{H} + \mathcal{H}^\top\mathcal{X}^\top < 0$ is equivalent to $\xi^\top\mathcal{P}\xi(k) < 0$ which gives in turn the following:

$$x^\top(k+1)Px(k+1) - x^\top Px(k) < \gamma^2 w^\top(k)w(k) - z^\top(k)z(k)$$

Based on the previous lemma, the closed-loop dynamics satisfies the $\mathcal{H}_\infty$ performance and is stable. This ends the proof of the theorem.   □

9.3.3 Output-Feedback Control

Now let us consider the output-feedback case. T We will consider the following dynamics:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + B_w w(k) \\ z(k) = C_z z(k) \\ y(k) = C_y x(k) \end{cases}$$

where $x(k)$ and $u(k)$ are respectively the state and the control, A, B, C_z and C are known matrices.

The controller we consider is given by:

$$\begin{cases} \hat{x}(k+1) = A\hat{x}(k) + Bu(k) + LC\left[x(k) - \hat{x}(k)\right] \\ u(k) = K\hat{x}(k) \end{cases}$$

where $\hat{x}(k)$ is the state of the controller and the gain K and L are to be determined.
If we define $e(k)$ as:

$$e(k) = x(k) - \hat{x}(k)$$

we have:

$$\begin{aligned} x(k+1) &= Ax(k) + BK\hat{x}(k) + B_w w(k) \\ &= [A + BK]\,x(k) - BKe(k) + B_w w(k) \\ e(k+1) &= [A - LC]\,e(k) + B_w w(k) \end{aligned}$$

Let $\eta(k)$ be defined by:

$$\eta(k) = \begin{bmatrix} x(k) \\ e(k) \end{bmatrix}$$

we get the following dynamics for the augmented system:

$$\begin{aligned} \eta(k+1) &= \tilde{A}\eta(k) + \tilde{B}_w w(k) \\ z(k) &= \tilde{C}_z \eta(k) \end{aligned}$$

where

$$\tilde{A} = \begin{bmatrix} A + BK & -BK \\ 0 & A - LC \end{bmatrix},$$

$$\tilde{B}_w = \begin{bmatrix} B_w \\ B_w \end{bmatrix}, \tilde{C}_z = \begin{bmatrix} C_z & 0 \end{bmatrix},$$

Theorem 9.3.3 *Let K and L be given gains and γ a positive scalar. If there exist a symmetric and positive-definite matrix $\tilde{P}$ such that the following LMI holds:*

$$\begin{bmatrix} -\tilde{P} & 0 & \tilde{A}^\top \tilde{P} & \tilde{C}_z^\top \\ \star & -\gamma^2 \mathbb{I} & \tilde{B}_w \tilde{P} & 0 \\ \star & \star & -\tilde{P} & 0 \\ \star & \star & \star & -\mathbb{I} \end{bmatrix} < 0$$

then closed-loop system is stable and satisfies the $\mathcal{H}_\infty$ performance.

Proof: Since $\tilde{C}_z^\top \tilde{C}_z$ is positive definite matrix, it results using the Schur complement that the closed-loop dynamics is stable.

Let $V(k) = \eta^\top(k)\tilde{P}\eta(k)$, where $\tilde{P}$ is a symmetric and positive-definite matrix solution of the LMI of the theorem.

Now let us compute $\Delta V(k)$, i.e.:

$$
\begin{aligned}
\Delta(k) &= V(k+1) - V(k) \\
&= \left[\tilde{A}\eta(k) + \tilde{B}_w w(k)\right]^\top \tilde{P}\left[A\eta(k) + \tilde{B}_w w(k)\right] - \eta(k)\tilde{P}\eta(k) \\
&= \left[\eta^\top(k)\; w^\top(k)\right]\begin{bmatrix} \tilde{A}^\top\tilde{P}\tilde{A} - P & \tilde{A}^\top\tilde{P}\tilde{B}_w \\ \star & \tilde{B}^\top\tilde{P}\tilde{B}_w \end{bmatrix}\begin{bmatrix} \eta(k) \\ w(k) \end{bmatrix}
\end{aligned}
$$

Now to establish the $\mathscr{H}_\infty$ performance, let us consider the following index:

$$
J = \sum_{k=0}^{\infty} z^\top(k)z(k) - \gamma^2 w^\top(k)w(k)
$$

where γ is a positive scalar.

Under zero initial conditions, we have:

$$
\begin{aligned}
J &= \sum_{k=0}^{\infty} z^\top(k)z(k) - \gamma^2 w^\top(k)w(k) + \Delta V \\
&= \sum_{k=0}^{\infty} \zeta^\top \Psi \zeta(k)
\end{aligned}
$$

where

$$
\Psi = \begin{bmatrix} \tilde{A}^\top\tilde{P}\tilde{A} - P & \tilde{A}^\top\tilde{P}\tilde{B}_w \\ \star & \tilde{B}^\top\tilde{P}\tilde{B}_w - \gamma^2\mathbb{I} \end{bmatrix}.
$$

Using Schur complement, we get:

$$
\Phi = \begin{bmatrix} -\tilde{P} & 0 & \tilde{A}^\top\tilde{P} & \tilde{C}_z^\top \\ \star & -\gamma^2\mathbb{I} & \tilde{B}_w\tilde{P} & 0 \\ \star & \star & -\tilde{P} & 0 \\ \star & \star & \star & -\mathbb{I} \end{bmatrix}.
$$

Therefore, the LMI of the theorem guarantees that $\Phi < 0$, i.e.: $J < 0$ which means that $\|e\|_2 \leq \gamma\|w\|_2$ which completes the proof of the theorem. $\square$

To design the controller, i.e.: the gains K and L, let us define:

$$
\tilde{X} = \tilde{P}^{-1} = \begin{bmatrix} X & 0 \\ 0 & X \end{bmatrix}
$$

Since the condition in the previous theorem is nonlinear in the design parameters K, L and X. To overcome this let us pre- and post-multiply the condition of the theorem by $\mathrm{diag}\,[X, \mathbb{I}, X, \mathbb{I}]$, we get:

$$
\begin{bmatrix} -\tilde{X} & 0 & \tilde{X}\tilde{A}^\top & \tilde{X}\tilde{C}_z^\top \\ \star & -\gamma^2\mathbb{I} & \tilde{X}\tilde{B}_w & 0 \\ \star & \star & -\tilde{X} & 0 \\ \star & \star & \star & -\mathbb{I} \end{bmatrix} < 0
$$

The nonlinear appears in the term $\tilde{A}\tilde{X}$. In fact, using the expressions of $\tilde{A}$ and $\tilde{X}$ we get:

$$
\begin{aligned}
\tilde{A}\tilde{X} &= \begin{bmatrix} A + BK & -BK \\ 0 & A - LC \end{bmatrix} \begin{bmatrix} X & 0 \\ 0 & X \end{bmatrix} \\
&= \begin{bmatrix} AX + BKX & -BKX \\ 0 & AX - LCX \end{bmatrix}
\end{aligned}
$$

Now if we define $Y = KX$ and add the new condition $CX = NC$ and define $W = LN$, we get the LMI condition that allows the design of the parameters K and L. The following theorem gives the design of the controller gains.

Theorem 9.3.4 *Let γ a positive scalar. If there exist symmetric and positive-definite matrices X, N and mtarix Y such that the following LMI holds:*

$$
\begin{bmatrix} -\tilde{X} & 0 & \tilde{X}\tilde{A}^\top & \tilde{X}\tilde{C}_z^\top \\ \star & -\gamma^2 \mathbb{I} & \tilde{X}\tilde{B}_w & 0 \\ \star & \star & -\tilde{X} & 0 \\ \star & \star & \star & -\mathbb{I} \end{bmatrix} < 0
$$

with

$$
\begin{bmatrix} AX + BY & -BY \\ \star & AX - WC \end{bmatrix},
$$

with the following constraint:

$$
CX = NC
$$

then closed-loop system is stable and satisfies the $\mathcal{H}_\infty$ performance and the gains are given by:

$$
\begin{aligned}
K &= YX^{-1} \\
L &= WN^{-1}
\end{aligned}
$$

9.4 Conclusion

This chapter covers some advanced concepts of control. It tackles the stability and the stabilization problem and their robustness. The uncertainties we considered in this chapter are of norm bounded type. LMI conditions are developed either for stability and stabilization. The controller gain is obtained from the resolution of an LMI. Some examples are provided to show how to solve the stability and the stabilization problems.

9.5   Problems

1. For the following dynamics check the stability and determine the best degree of stability we can get when the system is stable. Use Matlab to solve such problems.

   (a)
   $$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

   (b)
   $$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{bmatrix}$$

   (c)
   $$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix}$$

   (d)
   $$A = \begin{bmatrix} 0.3 & 1 & 0 \\ 0.1 & 0 & 1 \\ 0.2 & 0 & 0 \end{bmatrix}$$

   (e)
   $$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.006 & -0.11 & 0.6 \end{bmatrix}$$

   (f)
   $$A = \begin{bmatrix} 2.1 & 1 & 0 \\ -1.46 & 0 & 1 \\ 0.336 & 0 & 0 \end{bmatrix}$$

2. For the following dynamics check the robust stability and determine the best degree of stability we can get when the system is stable. Use Matlab to solve such problems.

   (a)
   $$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ 0.4 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0.4 \end{bmatrix},$$

   (b)
   $$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ 0.4 \\ 0.0 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0 & 0.4 \end{bmatrix},$$

(c)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} D_A = \begin{bmatrix} 0.2 \\ 0.1 \\ 0.3 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0.2 & 0.1 \end{bmatrix},$$

(d)

$$A = \begin{bmatrix} 0.3 & 1 & 0 \\ 0.1 & 0 & 1 \\ 0.2 & 0 & 0 \end{bmatrix} D_A = \begin{bmatrix} 0.1 \\ -0.1 \\ 0.2 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & -0.1 & 0.2 \end{bmatrix},$$

(e)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.006 & -0.11 & 0.6 \end{bmatrix} D_A = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.1 \end{bmatrix}, E_A = \begin{bmatrix} 0 & 0 & 0.4 \end{bmatrix},$$

(f)

$$A = \begin{bmatrix} 2.1 & 1 & 0 \\ -1.46 & 0 & 1 \\ 0.336 & 0 & 0 \end{bmatrix} D_A = \begin{bmatrix} 0.1 \\ 0.0 \\ 0.0 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0 & 0 \end{bmatrix},$$

3. For the following dynamics design the state feedback controller that stabilizes the closed-loop dynamics and determine the best degree of stability maximal we can get for the system. Use Matlab to solve such problems.

(a)

$$A = \begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix} B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 1 & -0.7 \end{bmatrix}$$

(b)

$$A = \begin{bmatrix} 3.1 & -2.3 & 0.2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} C = \begin{bmatrix} 0 & 1 & -0.7 \end{bmatrix}$$

(c)

$$A = \begin{bmatrix} -3.1 & -2.3 & -0.2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

(d)

$$A = \begin{bmatrix} -6.1 & -11.6 & -7.1 & -0.6 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} C = \begin{bmatrix} 0 & 0 & 1 & 5 \end{bmatrix}$$

4. For the following dynamics design the state feedback controller that assures the robust stability of the closed -loop dynamics and determine the best degree of stability that we can get for each system. Use Matlab to solve such problems.

(a)

$$A = \begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ 0.4 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0.4 \end{bmatrix},$$

(b)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 2 & 3 \end{bmatrix}, D_A = \begin{bmatrix} 0 \\ 0 \\ 0.4 \end{bmatrix}, E_A = \begin{bmatrix} 0 & 0 & 0.2 \end{bmatrix},$$

(c)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0.1 & -0.2 & 1 \end{bmatrix} D_A = \begin{bmatrix} 0.2 \\ 0 \\ 0 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & 0 & 0.1 \end{bmatrix},$$

(d)

$$A = \begin{bmatrix} 0.5 & 1 & 0 \\ -0.1 & 0 & 1 \\ 0.2 & 0 & 0 \end{bmatrix} D_A = \begin{bmatrix} 0.1 \\ 0 \\ 0 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & -0.1 & 0 \end{bmatrix},$$

(e)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.6 & 0.11 & 0.5 \end{bmatrix} D_A = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.2 \end{bmatrix}, E_A = \begin{bmatrix} 0 & 0 & 0.1 \end{bmatrix},$$

(f)

$$A = \begin{bmatrix} 1.1 & 1 & 0 \\ 1.46 & 0 & 1 \\ 0.3 & 0 & 0 \end{bmatrix} D_A = \begin{bmatrix} 0.3 \\ 0.0 \\ 0.0 \end{bmatrix}, E_A = \begin{bmatrix} 0.2 & 0 & 0 \end{bmatrix},$$

5. Consider a dynamical system:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + B_w w(k) \\ z(k) &= Cx(k) \\ y(k) &= Cx(k) \end{aligned}$$

with the following data:

$$A = \begin{bmatrix} -6.1 & -11.6 & -7.1 & -0.6 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$B_w = \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 0.1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 5 \end{bmatrix}$$

(a) design a state feedback controller that assures the $\mathcal{H}_\infty$ performance
(b) design a static output feedback controller that assures the $\mathcal{H}_\infty$ performance
(c) design a dynamical output feedback controller that assures the $\mathcal{H}_\infty$ performance

6. Consider a dynamical system:

$$x(k + 1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

with the following data:

$$A = \begin{bmatrix} -5.1 & -8.6 & -3.1 & 0.6 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 5 \end{bmatrix}$$

(a) design a state feedback controller
(b) design a static output feedback controller
(c) design a dynamical output feedback controller

10
Guaranteed Cost Control Problem

After reading this chapter, the reader will be able to :

1. define the guaranteed cost problem

2. design a stabilizing controller (state feedback, static output feedback, dynamic output feedback) for uncertain discrete-time systems

3. establish LMI conditions for the design of the stabilizing controllers

4. use existing tools to solve such problem

10.1 Introduction

In the previous chapter, we presented different results that can be used to deal with nominal and uncertain systems and also with systems that are subject to external disturbances. LMI conditions were developed to design controller to achieve the desired goal. In this chapter we consider uncertain discrete-time with norm bounded uncertainties and try to present results that can be used to design a controller such that the resulting closed-loop system is asymptotically stable while an upper bound on the closed-loop value of the associated linear quadratic cost function is guaranteed.

Guaranteed cost control for uncertain discrete-time systems is a control problem that has been a research topic of recurring interest in recent years. Its aim is to design a controller such that the resulting closed-loop system is asymptotically stable while an upper bound on the closed-loop value of an associated cost function is guaranteed. Various approaches have been developed and a great number of results on this topic have been reported in the literature using either the Riccati equation approach or the linear matrix inequality (LMI) approach. The LMI approach offers the possibility of solving the problem easily since tools for this purpose are already available.

In this chapter, we consider the guaranteed cost control problem for uncer- tain discrete-time systems. The uncertainties of the dynamics of the discrete-time system are assumed to be time-varying but norm-bounded. A linear quadratic cost function is defined as a performance measure for the closed-loop system. Attention is focused on the design of a state feedback controller which ensures not only the asymptotic stability of the closed-loop system but also an upper bound on the closed-loop value of the cost function. A sufficient condition for the solvability of this problem in the LMI setting is obtained. The static output feedback and the dynamic output feedback controllers are also tackled.

The rest of the chapter is organized as follows. In Section 2, the problem is stated. In Section 3, we developed the sufficient condition to design the state feedback controller such that the resulting closed-loop system is asymptotically stable while an upper bound on the closed-loop value of the associated linear quadratic cost function is guaranteed. Same results are developed in Section 4 for the static output feedback and the dynamic output feedback controllers.

10.2   Problem Statement

Let the dynamics of the discrete time system be described by the following:

$$\begin{cases} x(k + 1) = [A + \Delta A(k)]\, x(k) + [B + \Delta B(k)]\, u(k) \\ x(0) = x_0 \end{cases} \tag{10.1}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$ represent the values of the state and the input at the in-stant kT, A and B are known matrices with appropriate dimensions, $\Delta A(k)$ and $\Delta B(k)$ represent the uncertainties of the system that are assumed to have the following forms:

$$\begin{cases} \Delta A(k) = D_A F_A(k) E_A \\ \Delta B(k) = D_B F_B(k) E_B \end{cases} \tag{10.2}$$

with $F_A^\top(k)F_A(k) \leq \mathbb{I}$ and $F_B^\top(k)F_B(k) \leq \mathbb{I}$.

Associated with this system we have the following cost function:

$$J = \sum_{k=0}^{\infty} \left[x^\top(k)Qx(k) + u^\top(k)Ru(k) \right] \tag{10.3}$$

where Q is a symmetric and positive-definite matrix and R is a symmetric and positive-definite matrix.

The controller we will consider in the chapter has the following form:

$$u(k) = Kx(k) \tag{10.4}$$

Combining the system dynamics and the controller expression we obtain the following expression for the closed-loop dynamics:

$$\begin{cases} x(k+1) = [A + BK + \Delta A(k) + \Delta B(k)K]\, x(k) \\ x(0) = x_0 \end{cases}$$

The corresponding cost is given by:

$$J = \sum_{k=0}^{\infty} \left[x^\top(k) \left[Q + K^\top R K \right] x(k) \right]$$

The objective of this chapter is to design a state feedback controller with the form (10.4) such that the closed-loop dynamics is stable for all admissible uncertainties with a guaranteed cost. The design variable is the gain of the controller K that we have to determine.

Definition 10.2.1 *For the uncertain system (10.1) and cost (10.3), if there exist a control law $u^\star(.)$ and a positive scalar $J^\star$ such that for all admissible uncertainties, the closed-dynamics is asymptotically stable and $J \leq J^\star$, then $J^\star$ is said to be a guaranteed cost and $u^\star(.)$ is said to be a guaranteed cost control law*

In the rest of this chapter we will try to solve this control and develop LMI condition that can be used to design the state feedback controller that gives the desired objective.

10.3 State Feedback Control Design

The aim of this section is to design the state feedback controller such that the closed-dynamics is asymptotically stable and $J \leq J^\star$. First of all we will assume the existence of the controller and determine under which condition, the closed-dynamics is asymptotically stable and $J \leq J^\star$. Then, based on this condition we design the gain of the state feedback controller.

The following result gives the condition under which the closed-dynamics is asymptotically stable and $J \leq J^\star$.

Theorem 10.3.1 *Let K be a given gain. If there is exist symmetric and positive-definite matrices P, U and V, and positive scalars ε_A and ε_B such that the following LMI holds:*

$$
\begin{bmatrix}
-P & A^\top P + K^\top B^\top P & E_A^\top & K^\top E_B^\top & \mathbb{I} & K^\top \\
\star & -P + \varepsilon_A PDD^\top P + \varepsilon_B PDD^\top P & 0 & 0 & 0 & 0 \\
E_A & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\
E_B K & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\
\mathbb{I} & 0 & 0 & 0 & -U & 0 \\
K & 0 & 0 & 0 & 0 & -V
\end{bmatrix} < 0 \qquad (10.5)
$$

then the closed-loop system is stable and the upper bound of the cost is $J \leq x^\top(0)Px(0)$.

Proof: From the LMI (10.5) and the results of the previous chapter, we conclude that the closed-loop dynamics is stable. Let the Lyapunov function candidate be given by:

$$
V(x(k)) = x^\top(k)Px(k) \qquad (10.6)
$$

where P is solution of the LMI (10.5).

Using this Lyapunov function and the closed-loop dynamics, we get:

$$
\begin{aligned}
\Delta V(k) &= V(x(k+1)) - V(x(k)) \\
&= x^\top(k+1)Px(k+1) - x^\top(k)Px(k) \\
&= x^\top(k)\left[\mathscr{V}^\top P\mathscr{V} - P\right]x(k)
\end{aligned}
$$

with $\mathscr{V} = [A + BK + \Delta A(k) + \Delta B(k)K]$

Notice that

$$
\begin{bmatrix}
-P & \mathscr{V}^\top P \\
P\mathscr{V} & -P
\end{bmatrix}
$$

that can be rewritten as follows:

$$
\begin{bmatrix}
-P & A^\top P + K^\top B^\top P \\
\star & -P
\end{bmatrix}
+
\begin{bmatrix}
0 & \Delta A^\top(k)P \\
\star & 0
\end{bmatrix}
+
\begin{bmatrix}
0 & K^\top \Delta B^\top(k)P \\
\star & 0
\end{bmatrix}
$$

On the other hand, notice that:

$$
\begin{bmatrix}
0 & 0 \\
PD_A F_A(k)E_A & 0
\end{bmatrix}
=
\begin{bmatrix}
0 \\
PD_A
\end{bmatrix}
F_A(k)
\begin{bmatrix}
E_A & 0
\end{bmatrix}
$$

$$
\begin{bmatrix}
0 & 0 \\
PD_B F_B(k)E_B K & 0
\end{bmatrix}
=
\begin{bmatrix}
0 \\
PD_B
\end{bmatrix}
F_B(k)
\begin{bmatrix}
E_B K & 0
\end{bmatrix}
$$

Using Lemma 9.1.2, we get:

$$
\begin{bmatrix}
0 & \Delta A^\top(k)P \\
0 & 0
\end{bmatrix}
+
\begin{bmatrix}
0 & \Delta A^\top(k)P \\
0 & 0
\end{bmatrix}^\top
\leq \varepsilon_A
\begin{bmatrix}
0 & 0 \\
0 & PD_A D_A^\top P
\end{bmatrix}
$$
$$
+ \varepsilon_A^{-1}
\begin{bmatrix}
E_A^\top E_A & 0 \\
0 & 0
\end{bmatrix}
$$

$$
\begin{bmatrix} 0 & K^\top \Delta B^\top(k)P \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & K^\top \Delta B^\top(k)P \\ 0 & 0 \end{bmatrix}^\top \leq \varepsilon_B \begin{bmatrix} 0 & 0 \\ 0 & PD_BD_B^\top P \end{bmatrix}
$$

$$
+\varepsilon_B^{-1} \begin{bmatrix} K^\top E_B^\top E_B K & 0 \\ 0 & 0 \end{bmatrix}
$$

Using Schur complement, if the following holds, the system is stable:

$$
\begin{bmatrix} -P & A^\top P + K^\top B^\top P & E_A^\top & K^\top E_B^\top \\ \star & -P + \varepsilon_A PD_A D_A^\top P + \varepsilon_B PD_B D_B^\top P & 0 & 0 \\ E_A & 0 & -\varepsilon_A \mathbb{I} & 0 \\ E_B K & 0 & 0 & -\varepsilon_B \mathbb{I} \end{bmatrix} < 0
$$

Based on the LMI (10.5) of the Theorem 1, we get:

$$
\Delta V(k) + x^\top(k) \left[Q + K^\top RK \right] x(k) \leq 0
$$

with $Q = U^{-1}$ and $R = V^{-1}$.

Using the fact that the closed-loop dynamics is stable, we get:

$$
-x^\top(0)Px(0) + \sum_{k=0}^{\infty} x^\top(k) \left[Q + K^\top RK \right] x(k) \leq 0
$$

which gives

$$
J \leq x^\top(0)Px(0)
$$

This ends the proof of the theorem.   □

The design of the controller which assures that the closed-loop dynamics is asymptotically stable and the cost J is bounded for all admissible uncertainties is brought to the determination of the gain K. The condition (10.5) of the previous theorem can be used for this purpose since it already assures the stability of the closed-loop dynamics and the corresponding gain is bounded. This LMI can not be used directly since it is nonlinear in the decision variables P and K. Some transformations for this LMI are needed.

For this purpose, let $X = P^{-1}$ and pre- and post-multiplying the condition (10.5) respectively by diag $[X, X, \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I}]$ we get:

$$
\begin{bmatrix} -X & XA^\top + XK^\top B^\top & XE_A^\top & XK^\top E_B^\top & X & XK^\top \\ \star & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 & 0 & 0 \\ E_A X & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\ E_B K X & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\ X & 0 & 0 & 0 & -U & 0 \\ KX & 0 & 0 & 0 & 0 & -V \end{bmatrix} < 0
$$

Letting $Y = KX$, we get the following theorem.

Theorem 10.3.2 *There exists a guaranteed cost control law if there is exist symmetric and positive-definite matrices X, U and V, a matrix Y and positive scalars ε_A and ε_B such that the following LMI holds:*

$$
\begin{bmatrix}
-X & XA^\top + Y^\top B^\top & XE_A^\top & Y^\top E_B^\top & X & Y^\top \\
\star & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 & 0 & 0 \\
E_A X & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\
E_B Y & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\
X & 0 & 0 & 0 & -U & 0 \\
Y & 0 & 0 & 0 & 0 & -V
\end{bmatrix} < 0 \qquad (10.7)
$$

then the closed-loop system is stable under the control law $u(k) = YX^{-1}x(k)$ and the upper bound of the cost is $J \leq x^\top(0)X^{-1}x(0)$.

In the next example, we show how to solve the guaranteed cost control problem for the class of system we are considering. We will also give the code that can be used for this purpose.

Example 10.3.1 *To show how to solve the guaranteed control problem for the class of discrete-time systems, let us consider a system of the form (10.1) with the following data:*

$$
A = \begin{bmatrix} 0.7 & 0 & -0.5 \\ 0.05 & 0.8 & 0 \\ 0 & 0.3 & 0.6 \end{bmatrix}, B = \begin{bmatrix} 0.3 \\ 0 \\ 0.6 \end{bmatrix},
$$

$$
D_A = \begin{bmatrix} 0.1 \\ 0 \\ 0.2 \end{bmatrix}, D_B = \begin{bmatrix} 0.2 \\ 0 \\ 0.1 \end{bmatrix},
$$

$$
E_A = \begin{bmatrix} 0.3 & 0 & 0.2 \end{bmatrix}, E_B = \begin{bmatrix} 0.2 \end{bmatrix}
$$

First of all, we can check that the system is unstable. In fact the eigenvalues of the matrix A are:

$$
s_1 = 3.6992
$$
$$
s_2 = 0.8122
$$
$$
s_3 = 0.5886
$$

One pole is outside the unit circle and therefore even the nominal system is unstable. We can then use the results of the previous theorem to stabilize it and at the same time guarantee that the cost remains bounded for all admissible uncertainties. The following program can be used to compute the controller gain.

```
%%%%%%%%%%%%%%%%%%
% Guaranteed cost control for uncertain System  %
%%%%%%%%%%%%%%%%%%

clear all;
yalmip('clear')
```

```
% Data
A=[3.7 0 -0.5; 0.05 0.8 0; 0 0.3 0.6];
B=[0.3; 0; 0.6];

Da=[0.1; 0; 0.2];
Ea=[0.3 0 0.2];

Db=[0.2; 0; 0.1];
Eb=[0.2];

n=size(A,1);
m=size(B,2);

% Variables
X=sdpvar(n,n,'symmetric');
Y=sdpvar(m,n,'full');
U=sdpvar(n,n,'symmetric');
V=sdpvar(m,m,'symmetric');

epsa=sdpvar(1,1,'full');
epsb=sdpvar(1,1,'full');

% LMI
F=set(X>0)  % Initilization
F=F+set(X>0)
F=F+set(U>0)
F=F+set(V>0)

F=F+set(epsa>0)
F=F+set(epsb>0)

J=-X+epsa*Da*Da' +epsa*Db*Db' ;
F=F+set([-X     X*A'+Y'*B'   X*Ea'     Y'*Eb'       X        Y';
A*X+B*Y   J    zeros(n,1) zeros(n,1) zeros(n)  zeros(n,1);
Ea*X zeros(1,n)  -epsa*eye(1)   zeros(1)  zeros(1,n)  zeros(1);
Eb*Y zeros(1,n)    zeros(1)    -epsb*eye(1) zeros(1,n)  zeros(1) ;
X     zeros(n)   zeros(n,1)    zeros(n,1)  -U zeros(n,1);
Y     zeros(1,n) zeros(1)    zeros(1)     zeros(1,n)    -V] < 0)

% Solve
Sol=solvesdp(F)

% Extract data from the solution
X=double(X)
Y=double(Y)
U=double(U)
V=double(V)

epsa=double(epsa)
```

```
epsb=double(epsb)

K=Y*inv(X)

eig(X)
checkset(F)
```

If we run this program we get:

$$\varepsilon_A = 3.2836$$

$$\varepsilon_B = 80.7197$$

$$X = \begin{bmatrix} 1.3115 & 0.7176 & 5.7987 \\ 0.7176 & 27.9978 & -3.2042 \\ 5.7987 & -3.2042 & 30.2667 \end{bmatrix}$$

$$Y = \begin{bmatrix} -6.5607 & -10.5129 & -16.8854 \end{bmatrix}$$

$$U = \begin{bmatrix} 70.7693 & 3.1020 & 15.6204 \\ 3.1020 & 131.3457 & 0.6395 \\ 15.6204 & 0.6395 & 151.8819 \end{bmatrix}$$

$$V = \begin{bmatrix} 203.4407 \end{bmatrix}$$

The matrices X, U and V are all symmetric and positive-definite and therefore the LMI is feasible. The corresponding gain is given by:

$$K = \begin{bmatrix} -20.0355 & 0.5198 & 3.3356 \end{bmatrix}.$$

In this section we assumed that the states are available for feedback. More often this is not true for different reasons like the absence of appropriate sensors or for price reasons. Therefore, a controller that uses only the measured outputs (states) is preferable in this case. The next section tackles the design of controllers that use the measured states. Two types of controllers are presented, the static feedback output and the dynamic feedback output controllers.

10.4   Output Feedback Control

The dynamics in this case is given by:

$$\begin{cases} x(k + 1) = [A + \Delta A(k)] \, x(k) \, [B + \Delta B(k)] \, u(k) \\ y(k) = [C + \Delta C(k)] \, x(k) \, [D + \Delta D(k)] \, u(k) \end{cases} \tag{10.8}$$

where $u(k) \in \mathbb{R}^m$ is the control input and $y(k) \in R^p$ is the measured output and A, B, C and D are known known real matrices with appropriate dimensions and $\Delta A(k)$, $\Delta B(k)$, $\Delta C(k)$ and $\Delta D(k)$ are the system uncertainties that are supposed to satisfy the following:

$$\Delta A(k) = D_A F_A(k) E_A$$
$$\Delta B(k) = D_B F_B(k) E_B$$
$$\Delta C(k) = D_C F_C(k) E_C$$
$$\Delta D(k) = D_D F_D(k) E_D$$

with D_A, E_A, D_B, E_B, D_D, E_C, D_D and E_D are known matrices.

The corresponding cost function for the system (10.8) is given by 10.3).

Let us first of all consider the case of the following controller for the case when $D = 0$:

$$u(k) = Ky(k) = KCx(k) \tag{10.9}$$

where K is a the controller gain that we have to determine.

Using the expression of the controller, the closed-loop system becomes:

$$x(k + 1) = [A + \Delta A(k) + BKC + \Delta B(k)KC]\, x(k)$$

This dynamics will be stable and has a guaranteed cost for all admissible uncertainties if there exist symmetric and positive definite matrices P, U, and V and positive scalars ε_A and ε_B such that the following holds:

$$\begin{bmatrix} -P & A^\top P + C^\top K^\top B^\top P & E_A^\top & C^\top K^\top E_B^\top & \mathbb{I} & C^\top K^\top \\ \star & -P + \varepsilon_A P D_A D_A^\top P + \varepsilon_B P D_B D_B^\top P & 0 & 0 & 0 & 0 \\ E_A & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\ E_B KC & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\ \mathbb{I} & 0 & 0 & 0 & -U & 0 \\ KC & 0 & 0 & 0 & 0 & -V \end{bmatrix} < 0$$

Proceeding similarly as we did previously we get after posing $X = P^{-1}$ pre- and post- multiplying this by diag $[X, X, \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I}]$, we get:

$$\begin{bmatrix} -X & XA^\top + XC^\top K^\top B^\top & XE_A^\top & XC^\top K^\top E_B^\top & X & XC^\top K^\top \\ \star & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 & 0 & 0 \\ E_A & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\ E_B KCX & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\ X & 0 & 0 & 0 & -U & 0 \\ KCX & 0 & 0 & 0 & 0 & -V \end{bmatrix} < 0$$

Let us assume that the following holds:

$$CX = NC$$

Using this and posing $Y = KN$, we get the following result.

Theorem 10.4.1 *There exists a guaranteed cost control law if there is exist symmetric and positive-definite matrices X, U, V and N, a matrix Y and positive scalars ε_A and ε_B such that the following LMI holds:*

$$
\begin{bmatrix}
-X & XA^\top + C^\top Y^\top B^\top & XE_A^\top & C^\top Y^\top E_B^\top & X & C^\top Y^\top \\
\star & -X + \varepsilon_A D_A D_A^\top + \varepsilon_B D_B D_B^\top & 0 & 0 & 0 & 0 \\
E_A X & 0 & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\
E_B Y C & 0 & 0 & -\varepsilon_B \mathbb{I} & 0 & 0 \\
X & 0 & 0 & 0 & -U & 0 \\
Y C & 0 & 0 & 0 & 0 & -V
\end{bmatrix} < 0 \quad (10.10)
$$

with the following constraint:

$$
CX = NC \tag{10.11}
$$

then the closed-loop system is stable under the control law $u(k) = YN^{-1}x(k)$ and the upper bound of the cost is $J \leq x^\top(0)X^{-1}x(0)$.

Let us now concentrate on the design of the output dynamic feedback controller. The control law we are interested by in this section is given by:

$$
\begin{cases}
\hat{x}(k+1) = K_A \hat{x}(k) + K_B y(k) \\
u(k) = K_C \hat{x}(k)
\end{cases} \tag{10.12}
$$

where K_A, K_B and K_C are gains to be determined.

Our goal is to design a control law of the form (10.12) such that the closed-loop dynamics is stable and the cost is bounded for all admissible uncertainties.

In the rest of this section we will restrict ourself to the following expression for the uncertainties:

$$
\begin{aligned}
\Delta A(k) &= D_A F_A(k) E_A \\
\Delta C(k) &= D_C F_C(k) E_C
\end{aligned}
$$

where D_A, E_A, D_C and E_C are known matrices and $F_A(k)$ and $F_C(k)$ are the uncertainties that satisfies $F_A^\top(k)F_A(k) \leq \mathbb{I}$ and $F_C^\top(k)F_C(k) \leq \mathbb{I}$.

Letting $D = 0$ and the uncertainties on the matrix B are equal to zero in the dynamics and combining the system and the controller dynamics, we get the following one:

$$
\eta(k+1) = \left[\tilde{A} + \tilde{D}F(k)\tilde{E} \right] \eta(k) \tag{10.13}
$$

where

$$\eta(k) = \begin{bmatrix} x(k) \\ \hat{x}(k) \end{bmatrix},$$

$$\tilde{A} = \begin{bmatrix} A & BK_C \\ K_B C & K_A \end{bmatrix},$$

$$\tilde{D} = \begin{bmatrix} D_A & 0 \\ 0 & K_B D_C \end{bmatrix},$$

$$\tilde{E} = \begin{bmatrix} E_A & 0 \\ E_C & 0 \end{bmatrix}$$

$$F(k) = \begin{bmatrix} F_A & 0 \\ 0 & F_C \end{bmatrix}.$$

The corresponding cost function becomes:

$$J = \sum_{k=0}^{\infty} \eta^{\top}(k)\tilde{Q}\eta(k) \tag{10.14}$$

where

$$\tilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & K_C^{\top}RK_C \end{bmatrix}$$

There are different ways to design the dynamic output feedback controller. In the rest of this section, we will develop one among these ways (see [2]). Before designing the gains K_A, K_B and K_C let us establish under which conditions the closed-loop dynamics for given gains will be robustly stable and has guaranteed cost. The following theorem gives such results.

Theorem 10.4.2 *Let K_A, K_B and K_C be given gains. If there exist symmetric and positive-definite matrices X, U and V and positive scalars ε_A and ε_C such that the following LMI holds:*

$$\begin{bmatrix} -X & 0 & XA^{\top} & XC^{\top}K_B^{\top} & XE_A^{\top} & XE_C^{\top} & X & 0 \\ \star & -X & XK_C^{\top}B^{\top} & XK_A^{\top} & 0 & 0 & 0 & XK_C^{\top} \\ \star & \star & -X+\varepsilon_A D_A D_A^{\top} & 0 & 0 & 0 & 0 & 0 \\ \star & \star & \star & -X+\varepsilon_C K_B D_C D_C^{\top}K_B^{\top} & 0 & 0 & 0 & 0 \\ \star & \star & \star & \star & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\ \star & \star & \star & \star & \star & -\varepsilon_C \mathbb{I} & 0 & 0 \\ \star & \star & \star & \star & \star & \star & -U & 0 \\ \star & \star & \star & \star & \star & \star & \star & -V \end{bmatrix} < 0 \tag{10.15}$$

then the closed-loop system is stable and the upper bound of the cost is $J \leq x^{\top}(0)X^{-1}x(0)$.

Proof: From the LMI (10.15) and the results of the previous chapter, we conclude that the closed-loop dynamics is stable. Let the Lyapunov function candidate be given by:

$$V(\eta(k)) = \eta^{\top}(k)\tilde{P}\eta(k)$$

where $\tilde{P} = \tilde{X}^{-1}$ is solution of the LMI (10.15).

Using this Lyapunov function and the closed-loop dynamics, we get:

$$
\begin{aligned}
\Delta V(k) &= V(\eta(k+1)) - V(\eta(k)) \\
&= \eta^\top(k+1)\tilde{P}\eta(k+1) - \eta^\top(k)\tilde{P}\eta(k) \\
&= \eta^\top(k)\left[\mathcal{V}^\top\tilde{P}\mathcal{V} - \tilde{P}\right]\eta(k)
\end{aligned}
$$

with $\mathcal{V} = \left[\tilde{A} + \Delta\tilde{A}(k)\right]$

Notice that

$$
\begin{bmatrix} -\tilde{P} & \mathcal{V}^\top\tilde{P} \\ \tilde{P}\mathcal{V} & -\tilde{P} \end{bmatrix}
$$

that can be rewritten as follows:

$$
\begin{bmatrix} -\tilde{P} & \tilde{A}^\top\tilde{P} \\ \star & -\tilde{P} \end{bmatrix} + \begin{bmatrix} 0 & \Delta\tilde{A}^\top(k)\tilde{P} \\ \star & 0 \end{bmatrix}
$$

On the other hand, notice that:

$$
\begin{bmatrix} 0 & 0 \\ \tilde{P}\tilde{D}F(k)\tilde{E} & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{P}\tilde{D} \end{bmatrix} F(k) \begin{bmatrix} \tilde{E} & 0 \end{bmatrix}
$$

Using the lemma 9.1.2, we get:

$$
\begin{aligned}
\begin{bmatrix} 0 & \Delta\tilde{A}^\top(k)\tilde{P} \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \Delta\tilde{A}^\top(k)\tilde{P} \\ 0 & 0 \end{bmatrix}^\top &\leq \varepsilon \begin{bmatrix} 0 & 0 \\ 0 & \tilde{P}\tilde{D}\tilde{D}^\top\tilde{P} \end{bmatrix} \\
&+ \varepsilon^{-1} \begin{bmatrix} \tilde{E}^\top\tilde{E} & 0 \\ 0 & 0 \end{bmatrix}
\end{aligned}
$$

for $\varepsilon > 0$.

Using Schur complement, if the following holds, the system is stable:

$$
\begin{bmatrix} -\tilde{P} + \varepsilon\tilde{E}^\top\tilde{E} & \tilde{A}^\top\tilde{P} \\ \star & -\tilde{P} + \varepsilon\tilde{P}\tilde{D}\tilde{D}^\top\tilde{P} \end{bmatrix} < 0
$$

Based on the LMI (10.15) of the theorem, we get:

$$
\Delta V(k) + \eta^\top(k)\tilde{Q}\eta(k) \leq 0
$$

with $Q = U^{-1}$ and $R = V^{-1}$.

Using the fact that the closed-loop dynamics is stable, we get:

$$
-x^\top(0)Px(0) + \sum_{k=0}^{\infty} x^\top(k)\left[Q + K^\top RK\right]x(k) \leq 0
$$

which gives

$$
J \leq x^\top(0)Px(0)
$$

This ends the proof of the theorem.   □

Theorem 10.4.3 *Consider the system (10.8) with the dynamic output feedback controller, then system (10.8) is robustly stabilizable if there exist symmetric and positive-definite matrices X, U and V, a matrix Y and positive scalars ε_A and ε_C such that the following LMI is feasible:*

$$
\begin{bmatrix}
-X & 0 & XA^\top & XC^\top C & XE_A^\top & XE_C^\top & X & 0 \\
\star & -X & Y^\top B^\top & XA^\top & 0 & 0 & 0 & Y^\top \\
\star & \star & -X + \varepsilon_A D_A D_A^\top & 0 & 0 & 0 & 0 & 0 \\
\star & \star & \star & -X + \varepsilon_C C^\top D_C D_C^\top C & 0 & 0 & 0 & 0 \\
\star & \star & \star & \star & -\varepsilon_A \mathbb{I} & 0 & 0 & 0 \\
\star & \star & \star & \star & \star & -\varepsilon_C \mathbb{I} & 0 & 0 \\
\star & \star & \star & \star & \star & \star & -U & 0 \\
\star & \star & \star & \star & \star & \star & \star & -V
\end{bmatrix} < 0 \text{(10.16)}
$$

Moreover

1. *the dynamic output feedback controller and the control law are given by:*

$$
\begin{aligned}
\hat{x}(k+1) &= A\hat{x}(k) + C^\top y(k) \\
u(k) &= YX^{-1}\hat{x}(k)
\end{aligned}
$$

2. *the cost function satisfies:*

$$
J \le x^\top(0)X^{-1}x(0)
$$

Remark 10.4.1 *The feasible solution of the previous theorem allows us to construct a family of controllers which represents an advantage of this approach. The controller that may assure the minimum bound of interest.*

To design the controller that assures the minimum bound for the cost notice that $x^\top(0)X^{-1}x(0) < \varrho$ for a given initial vector $x(0)$ can be rewritten as follows:

$$
\begin{bmatrix}
-\varrho & x^\top(0) \\
x(0) & -X
\end{bmatrix} < 0
\tag{10.17}
$$

The following optimization problem will give the desired controller:

$$
\min \varrho \text{ s.t.: } (10.16)\text{-}(10.17)
$$

Corollary 10.4.1 *Let X, U, V, Y, ϱ, ε_A and ε_C be the solution of the optimization problem, then the controller (10.12) with:*

$$
\begin{aligned}
K_A &= A \\
K_B &= C^\top \\
K_C &\ YX^{-1}
\end{aligned}
$$

will stabilize the system and at the same time assure the minimum cost.

10.5   Conclusion

In this chapter we considered uncertain discrete-time systems and design a state feedback controller that stabilized the closed-loop dynamics and at the sane time guarantees that the associated cost remains bounded for all admissible uncertainties. LMI condition is established to design the state feedback controller gain.

10.6   Problems

1. For the following dynamical discrete-time systems with the following data:

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 5 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}, E_A = \begin{bmatrix} 0.2 & -0.1 \end{bmatrix},$$

$$D_B = \begin{bmatrix} -0.1 \\ 0.2 \end{bmatrix} E_B = \begin{bmatrix} 0.1 \end{bmatrix}, Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} R = \begin{bmatrix} 0.1 \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ -2 & -5 \end{bmatrix}, B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, D_A = \begin{bmatrix} -0.1 \\ -0.2 \end{bmatrix}, E_A = \begin{bmatrix} 0.1 & -0.11 \end{bmatrix},$$

$$D_B = \begin{bmatrix} 0.1 \\ -0.1 \end{bmatrix}, E_B = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix}, Q = \begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix} R = \begin{bmatrix} 5 \end{bmatrix}$$

(a) write a Matlab program to solve the guaranteed cost problem
(b) give initial conditions and write a simulation program to plot the behavior of the states versus time

2. For the following dynamical discrete-time system with the following data:

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ -0.1 \end{bmatrix}, E_A = \begin{bmatrix} -0.1 & -0.1 \end{bmatrix},$$

$$D_B = \begin{bmatrix} 0.1 \\ -0.2 \end{bmatrix} E_B = \begin{bmatrix} -0.1 \end{bmatrix}$$

(a) design the following controllers:

   i. state feedback

   ii. static output feedback

   iii. dynamic output feedback

   that assure the guaranteed cost
(b) give initial conditions and write a simulation program to plot the behavior of the states versus time when $F(k) = 0.1\sin(k)$

3. For the following dynamical discrete-time system with the following data:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ -1 & 1 & 0 \\ -1 & -2 & -3 \end{bmatrix}, B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, D_A = \begin{bmatrix} 0.1 \\ -0.1 \\ 0 \end{bmatrix}, E_A = \begin{bmatrix} -0.1 & -0.1 & -0.1 \end{bmatrix},$$

$$D_B = \begin{bmatrix} 0.1 \\ -0.2 \\ -0.1 \end{bmatrix} E_B = \begin{bmatrix} -0.1 & -0.2 \end{bmatrix}$$

(a) design the following controllers:
   i. state feedback
   ii. static output feedback
   iii. dynamic output feedback

   that assure the guaranteed cost

(b) give initial conditions and write a simulation program to plot the behavior of the states versus time when $F(k) = 0.5 \cos(k)$

Part VII

Case Studies

11
Case Studies

After reading this chapter the reader will:

1. experiment his knowledge in regard to the design and real time implementation of mechatronic systems

2. live the design of mechatronic systems from A to Z

3. be able to perform the different phases of the design of mechatronic systems

4. be able to solve the control problem and establish the control law that we have to implement in real time

5. be able to write programs in C language using the interrupt concept for real time implementation

11.1   Introduction

In the previous chapters we developed some concepts that we illustrated their applications by academic examples to show the readers how these results apply. More specifically, we have seen how to design the mechatronic systems and we have

presented the different steps that we must follow to success the design of the desired mechatronic system. We have presented the approaches that we must use in the design of:

- the mechanical part

- the electronic circuit

- the program in C for the real-time implementation

These tools were applied for some practical systems and more details were given to help the reader to execute his own design.

For the control algorithms most of the examples we presented were academic with perfect models. Unfortunately, for a practical system the model we will have is just a realization that may describe the system in some particular conditions, and for some reasons this model will not work perfectly as expected during the real-time implementation of the algorithms. This can be caused by different neglected dynamics that may change the behavior for some frequencies.

The aim of this chapter is to show the reader how we can implement in real-time the theoretical results we developed earlier in the previous chapters for practical systems. We will proceed gradually and show all the steps to make it easy for the readers. The case studies we consider in this chapter are those that are discussed and designed in the previous chapters.

11.2   Velocity Control of the dc Motor Kit

As a first example, let us consider the speed control of the dc motor driving a mechanical part. The choice of this example is very important since most of the systems will use such dc motor. The dc motor we will consider is manufactured by Maxon company. This motor is very important since it comes with a gearbox (ratio 6:1) and an encoder that gives one hundred pulses per revolution, which gives 600 pulses per revolution that we exploit by using quadrature method to bring it to two thousand four hundred pulses per revolution. The system we are using in this example for the real-time implementation of control algorithms we presented earlier if more flexible and offer more advantages.

The data sheet of this motor gives all the important parameters and therefore the transfer function of this actuator can be obtained easily. The load we are considering in this example is a small disk with graduation that we would like to control in speed and later on in position. This setup is illustrated in Figs. 11.1-(11.2). The disk we are considering has a diameter equal to 0.06 m and a mass equal to 0.050 Kg. With these data and the one of the data sheet of the dc motor, we can get the transfer function between the velocity of the disk and the input voltage.

Let us first of all concentrate on speed control of the load. In this case to establish the transfer function of this system (dc motor actuator and its load) we can either use the data sheet, the information we have on the disk and the results in Boukas [1]
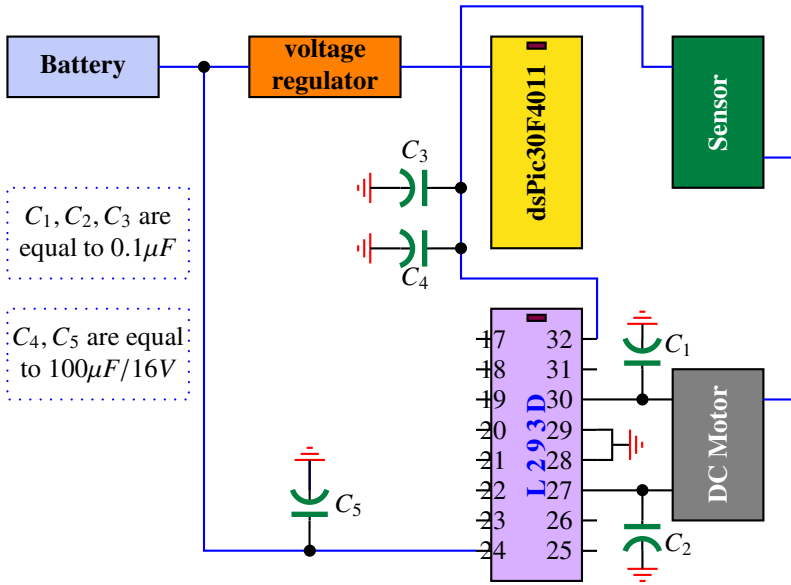
Fig. 11.1 Electronic circuit of dc motor kit

or proceed with an identification. Using the first approach the results of Chapter 2 in [1], we get:

$$G(s) = \frac{K}{\tau s + 1}$$

with

$$K = 48.91$$
$$\tau = 63.921\ ms$$

For the identification of our system, we can do it in real-time using the real-time implementation setup and appropriate C program. Since the microcontroller owns limited memory, the identification can be done into two steps. Firstly, in a first experiment the gain, K, is determined, then using this gain we compute the steady state value that can be used to compute the constant time τ.

To design the controller we should first of all specify the performances we would like that our system has. As a first performance, we require that our system is stable. It is also needed that the system speed will have a good behavior at the transient regime with a zero error at the steady state regime for a step reference. For the transient we would like that our load has a settling time with 5% less or equal to $\frac{3\tau}{5}$ and an overshoot less or equal to 5%.

To accomplish the design of the appropriate controller we can either make the design in continuous-time and then obtain the algorithm we should program in the software part, or proceed with all the design in the discrete-time directly. In the rest of this example, we will opt for the second approach.
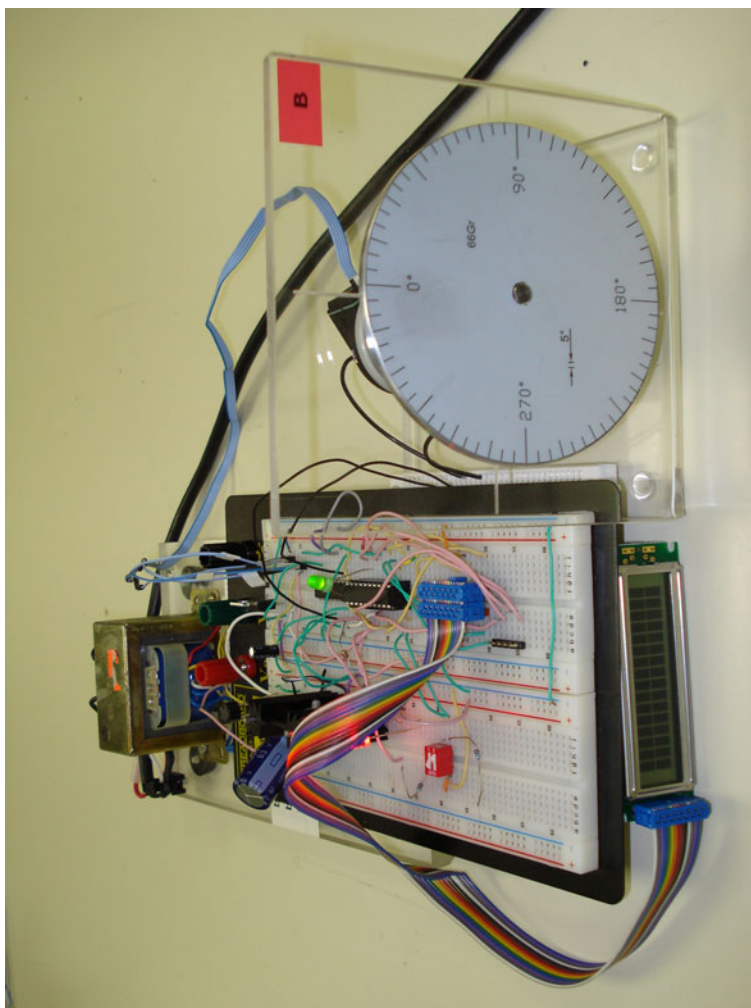
Fig. 11.2 Real-time implementation setup

From the expression of the system transfer function, and the desired performances it results that we need at least a proportional and integrator (PI) controller. The transfer function of this controller is given by:

$$C(s) = K_P + \frac{K_I}{s}$$

where K_P and K_I are gains to be determined to force the load to have the performances we imposed.

Using a zero-order-holder and the $\mathscr{Z}$-transform table we get:

$$
\begin{aligned}
G(z) &= \frac{Kz\left(1 - e^{-\frac{T}{\tau}}\right)\left(1 - z^{-1}\right)}{(z - 1)\left(z - e^{-\frac{T}{\tau}}\right)} \\
&= \frac{K\left(1 - e^{-\frac{T}{\tau}}\right)}{z - e^{-\frac{T}{\tau}}}
\end{aligned}
$$

For the controller, using the trapezoidal discretisation we get:

$$
\begin{aligned}
C(z) &= \frac{U(z)}{E(z)} = K_P + K_I \frac{T}{2} \frac{z + 1}{z - 1} \\
&= \frac{\left(K_P + \frac{TK_I}{2}\right)z + \left(-K_P + \frac{TK_I}{2}\right)}{z - 1}
\end{aligned}
$$

Dividing the numerator and the denominator by z and going back to time, we get:

$$
u(k) = u(k - 1) + \left(K_P + \frac{TK_I}{2}\right)e(k) + \left(-K_P + \frac{TK_I}{2}\right)e(k - 1)
$$

Combining the transfer function of the actuator and its load and the one of the controller we get the following closed-loop transfer function:

$$
F(z) = \frac{K\left(1 - e^{-\frac{T}{\tau}}\right)\left(K_P + \frac{TK_I}{2}\right)z + K\left(1 - e^{-\frac{T}{\tau}}\right)\left(-K_P + \frac{TK_I}{2}\right)}{z^2 + \left(K\left(K_P + \frac{TK_I}{2}\right)\left(1 - e^{-\frac{T}{\tau}}\right) - 1 - e^{-\frac{T}{\tau}}\right)z + K\left(1 - e^{-\frac{T}{\tau}}\right)\left(-K_P + \frac{TK_I}{2}\right) + e^{-\frac{T}{\tau}}}
$$

Using now the desired performances, it is easy to conclude that the dominant poles are

$$
s_{1,2} = -\zeta\omega_n \pm jw_n \sqrt{1 - \zeta^2}
$$

where ζ and ω_n represent respectively the damping ratio and the natural frequency of the closed-loop of our system control.

From control theory (see Boukas [1]) it is well known that the overshoot $d\%$ and the settling time t_s at 5% are given by:

$$
\begin{aligned}
d\% &= 100e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} \\
t_s &= \frac{3}{\zeta\omega_n}
\end{aligned}
$$

Using our performances and these expressions we conclude that:

$$
\begin{aligned}
\zeta &= 0.707 \\
\omega_n &= \frac{5}{\tau\zeta} = 110.6387 rad/s
\end{aligned}
$$

which give the following dominant poles:

$$
s_{1,2} = -78.2216 \pm 78.2452 j
$$

Using the transformation $z = e^{Ts}$ with $T = \frac{\tau}{10} = 0.0064$, we obtain the following dominant poles in the $\mathscr{Z}$-domain:

$$z_{1,2} = 0.5317 \pm 0.2910j$$

With these poles we have the following characteristic equation:

$$\begin{aligned} \Delta_d &= (z - 0.5317 - 0.2910j)(z - 0.5317 + 0.2910j) \\ &= z^2 - 1.0634z + 0.3674 \end{aligned}$$

Using now the poles placement technique we get:

$$1 + C(z)G(z) = \Delta_d$$

which implies:

$$K_I = \frac{0.3040}{KT\left(1 - e^{-\frac{T}{\tau}}\right)}$$

$$K_P = \frac{-0.4308 + 2e^{-\frac{T}{\tau}}}{2K\left(1 - e^{-\frac{T}{\tau}}\right)}$$

Using the values of K, T and τ, we get the following expression for the gains K_P and K_I:

$$K_P = 0.1480$$
$$K_I = 10.1951$$

Remark 11.2.1 *Cautions have to be made in this case since we don't care about the positions of the zero of the transfer function and therefore we may have some surprises when implementing this controller. It is clear that the performances we will get (settling time and the overshoot) will depend on the positions of the zero. For more details on this matter we refer the reader to Boukas [1].*

To implement now this PI control algorithm and assure the desired performances we will use a microcontroller from Microship[1]. This choice is due to our experience with this type of microcontroller. The reader can keep in mind that any other microcontroller from other manufacturer with some small changes will do the job. In this example we will use the microcontroller dsPIC30F4011 from Microhip.

The code for our implementation is made in *C*-language. This language is adopted for its simplicity. The implementation has the following structure:

```
//
// Put here the include
//

    #include "p30F4011.h"        // proc specific header

//
```

---

[1] See www.microchip.com

```
 // Define a struct
 //
  typedef struct {
      // PI Gains
      float K_P;        // Propotional gain
      float K_I;         // Integral gain
    //
    // PI Constants
    //
    float Const1_pid;  // KP + T KI/2
    float Const2_pid;  // -KP + T KI/2
    float Reference;   // speed reference

    //
    // System variables
    //
    float y_k;    // y_m[k] -> measured output at time k
    float u_k;    // u[k]    -> output at time k
    float e_k;    // e[k]    -> error at time k

    //
    // System past variables
    //
    float u_prec;      // u[k-1] -> output at time k-1
    float e_prec;      // e[k-1] -> error at time k-1

  }PIStruct;

  PIStruct thePI;

  thePI.Const1= thePI.K_P+T*thePI.K_I/2;
  thePI.Const2=-thePI.K_P+T*thePI.K_I/2;
  thePI.Reference=600;


//
// Functions
//
float ReadSpeed(void);

float ComputeControl(void);

float SendControl(void);

//
// Interrupt program here using Timer 1 (overflow of counter Timer 1)
//
void __ISR _T1Interrupt(void)     // interrupt routine code
  {
```

```
        // Interrupt Service Routine code goes here
        float Position_error;

        //
        // Read speed
        //
         thePI.y_m=ReadSpeed();

        thePI.e_k= thePI.Reference-thePI.y_m;

        //
        // Compute the control
        //
        ComputeContrl();

        //
        // Send control
        //
        SendControl();

        IFS0bits.T1IF=0;    // Disable the interrupt
  }


int main ( void )           // start of main application code
{
// Application code goes here
int i;

// Initialize the variables Reference and ThePID.y_m (it can be read
 from inputs) Reference =  0x8000;  // Hexadecimal number
(0b...  Binary number) ThePID = 0x8000;

// Initialize the registers
TRISC=0x9fff; // RC13 and RC14 (pins 15 and 16) are configured as outputs
IEC0bits.T1IE=1;   // Enable the interrupt on Timer 1

// Indefinite loop
while (1)
{
}
return 0
}


      % ReadSpeed function
      int ReadSpeed (void)
          {
          }
```

```
% ComputeControl function
int ComputeControl (void)
{
thePI.u_k=thePI.u_prec+thePI.Const1*thePI.e_k+thePI.Const2*thePI.e_prec;
}

% SendControl function
int Send Control (void)
{
sendControl()

//
// Update past data
//
thePI.u_prec=thePI.u_k;
ThePI.e_prec=thePI.e_k;
}
```

As it can be seen from this structure, first of all we notice that the system will enter the loop and at each interrupt the call for the functions:

- ReadSpeed;
- ComputeControl;
- SendControl;

is made and the appropriate action is taken.

The ReadSpeed function returns the load speed at each sampling time that will be used by the ComputeControl function. The SendControl function sends the appropriate voltage to the actuator via the L293D chip.

Using the compiler HighTec C to get the hex code and the PicKit-2 to upload the file in the memory of the microcontroller. For more detail on how to get the hex code we invite the reader to the manual of the compiler HighTec C or the compiler C30 of Microchip.

The state approach in this case is trivial and we will not develop it.

11.3   Position Control of the dc Motor Kit

Let us focus on the load position control. Following similar steps as for the load velocity control developed in the previous section, we need firstly to choose the desired performances we would like our system will have. The following performances are imposed:

- system stable in the closed-loop;

- settling time t_s at 2% equal to the best one we can have

- overshoot equal to 5%

- steady-state equal to zero for a step function as input

Using the performances and the transfer function, it is easy to conclude that a proportional controller K_P is enough to satisfy these performances.

In this example, we will use the continuous-time approach for the design of the controller. Based on the past chapter, the model of our system is given by:

$$G(s) = \frac{K}{s\,(\tau s + 1)}$$

where K and τ take the same values as for the speed control.

Let the transfer controller be give by:

$$C(s) = K_P$$

Using these expression, the closed-loop transfer function is given by:

$$
\begin{aligned}
F(s) &= \frac{C(s)G(s)}{1 + C(s)G(s)} \\
&= \frac{\frac{KK_P}{\tau}}{s^2 + \frac{1}{\tau}s + \frac{KK_P}{\tau}}
\end{aligned}
$$

Since the system is of type 1, it results that the error to a step function as input is equal to zero with a proportional controller.

Based on the specifications, the following complex poles:

$$s_{1,2} = -\zeta w_n \pm j w_n \sqrt{1 - \zeta^2}$$

will do the job and the corresponding characteristic equation is given by:

$$s^2 + 2\zeta w_n s + w_n^2 = 0.$$

Equating this with the one of the closed-loop system we get:

$$
\begin{aligned}
2\zeta w_n &= \frac{1}{\tau} \\
w_n^2 &= \frac{KK_P}{\tau}.
\end{aligned}
$$

To determine the best settling time t_s at 2 %, notice that we have:

$$t_s = \frac{4}{\zeta w_n}.$$

Using now the fact:

$$\zeta w_n = \frac{1}{2\tau}$$

we obtain:

$$t_s = 8\tau$$

Therefore the best settling time at %2 we can have with this controller is 8 times the constant time of the system. Any value less than will be attainable. In fact, this is trivial if we look to the root locus of the closed-loop system when varying K_P. This is given by Fig. 11.3. To fix the gain of the controller the desired poles $s_{1,2} = -7.5 \pm j$, we use the figure and choosing a $\zeta = 0.707$. This gives $K_P = 0.1471$.
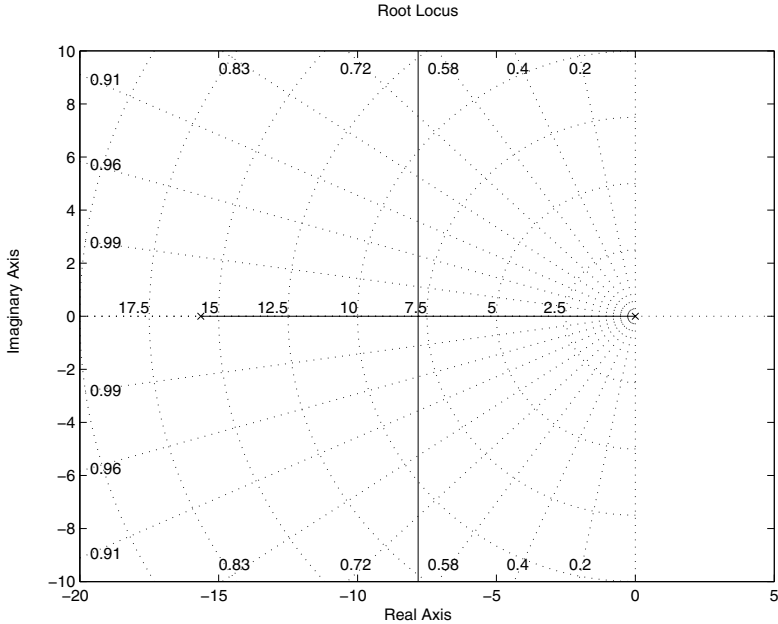
Fig. 11.3 Root locus of the dc motor with a proportional controller

Using this controller, the time response for a step function of amplitude equal to 30 degrees is represented by the Fig. 11.4 from which we conclude that the designed controller satisfies all the desired performances with a settling time at %2 equal to 0.5115 s. But if we implement this controller, the reality will be different from simulation since the backlash of the gearbox is not included in the used model and therefore in real-time the result will be different and the error will never be zero. To overcome this problem we can use a proportional and derivative controller that may give better settling time at %2. Let the transfer function of this controller be given by:

$$C(s) = K_P + K_D s$$

where K_P and K_D are the gain to be determined.

Remark 11.3.1 *It is important to notice that the use of a proportional and deriva-tive controller will introduce a zero in the closed-loop transfer that may improve the settling time if it is well placed. Depending on its position, the overshoot and the settling time will be affected. For more details on this matter we refer the reader to [1].*

Time response for a step function with 30 degrees as amplitude
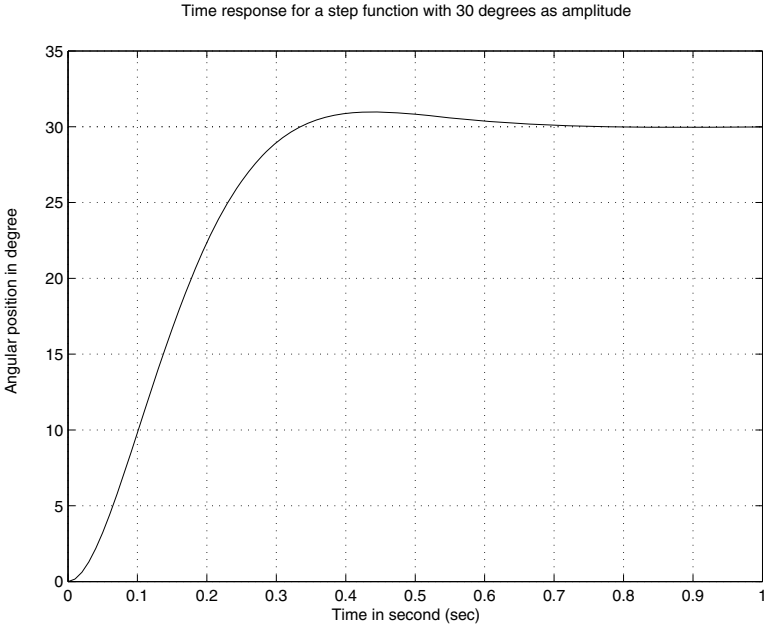
Fig. 11.4 Time response for a step function with 30 degrees as amplitude

With this controller the closed-loop transfer function is given by:

$$
\begin{aligned}
G(s) &= \frac{C(s)G(s)}{1 + C(s)G(s)} \\
&= \frac{\frac{KK_D s + KK_P}{\tau}}{s^2 + \frac{1 + KK_D}{\tau}s + \frac{KK_P}{\tau}}
\end{aligned}
$$

As before two complex poles are used for the design of the controller. If we equate the two characteristic equations we get:

$$
\begin{aligned}
2\zeta w_n &= \frac{1 + KK_D}{\tau} \\
w_n^2 &= \frac{KK_P}{\tau}.
\end{aligned}
$$

In this case we have two unknown variables K_P and K_D and two algebraic equations which determines uniquely the gains. Their expressions are given by:

$$K_P = \frac{\tau w_n^2}{K}$$

$$K_D = \frac{2\tau \zeta w_n}{K}$$

Using now the desired performances, we conclude similarly as before that the steady error to an input equal to step function of amplitude equal to 30 degrees for instance is equal to zero and the damping ratio ζ corresponding to an overshoot equal to %5 is equal to 0.707. The settling time, t_s at % 2, that we may fix as a proportion of the time constant of the system, gives:

$$w_n = \frac{4}{\zeta t_s}.$$

Now if we fix the settling time at 3τ, we get:

$$w_n = 29.4985.$$

Using these values we get the following ones for the controller gains:

$$K_P = 1.1374$$

$$K_D = 0.0545.$$

which gives the following complex poles:

$$s_{1,2} = -28.6763 \pm 6.9163\,j.$$

and the zero at:

$$z = -20.8618.$$

Using this controller the time response for an input with an amplitude equal to 30 degrees is represented in Fig. 11.5. As it can be seen from this figure that the overshoot and the settling time are less those obtained using the proportional controller.

To implement either the proportional or the proportional and derivative controllers we need to get the recurrent equation for the control law. For this purpose, we need to discretize the transfer function of the controller using the different methods presented earlier. Let us use the trapezoidal method which consists to replace s by $\frac{2}{T}\frac{z-1}{z+1}$. This gives:

$C(z) = K_p$ for the proportional controller

$C(z) = K_P + K_D \dfrac{2}{T}\dfrac{z-1}{z+1}$ for the proportional and derivative controller

If we denote by $u(k)$ and $e(k)$ by the control and the error between the reference and the output at instant kT, we get the following expressions:

1. for the proportional

$$u(k) = K_P e(k)$$
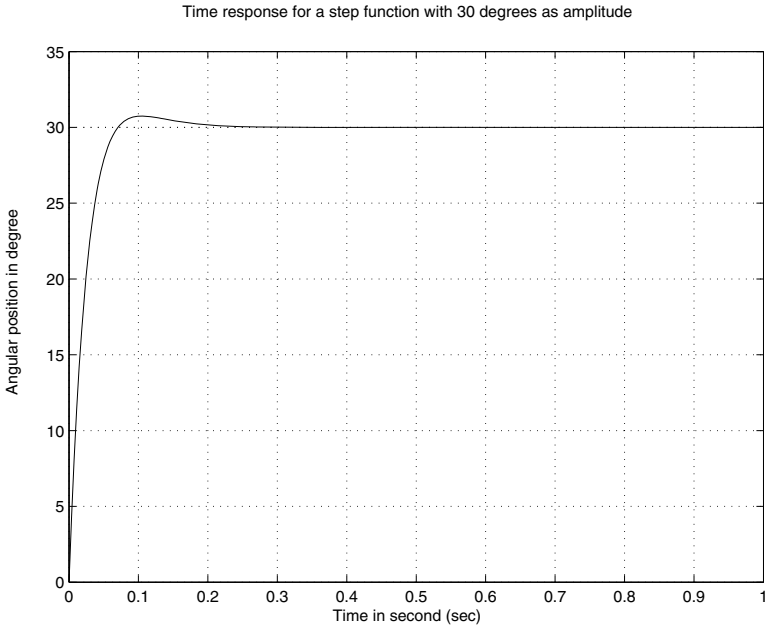
Fig. 11.5 Time response for a step function with 30 degrees as amplitude

2. for the proportional and derivative controller

$$u(k) \; = -u(k-1) + \left(K_P + \frac{2K_D}{T}\right)e(k) + \left(K_P - \frac{2K_D}{T}\right)e(k-1)$$

The implementation is this controller uses the same function with some minors changes. The listing the corresponding functions is:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Main program                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

main
% Data

% Variables

% While loop
  while (1)
     do
        ReadSpeed;
        ComputeControl;
        SendControl;
     end;
```

```
% ReadSpeed function

% ComputeControl function

% SendControl function
```

Let us now use the state space representation of this example and design a state feedback controller that will guarantee the desired performances. For this case we will firstly assume the complete access to the states and secondly we relax this assumption by assuming that we have access only to the position. As we did previously we can proceed either in the continuous-time or in the discrete-time.

Previously we establish the state space description of this system and it is given by:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix} u(t)$$

where $x(t) \in \mathbb{R}^2$ ($x_1(t) = \theta(t)$ and $x_2(t) = \dot{\theta}(t)$), and $u(t) \in \mathbb{R}$ (the applied voltage).

From the desired performances with a settling time at %2 equal to 3τ, we get the same dominant poles as before, and therefore the same characteristic equation, $\Delta_d(s) = s^2 + 2\zeta w_n s + w_n^2 = 0$ (with $\zeta = 0.707$ and $w_n = \frac{4}{3\zeta\tau}$). Using the controller expression the closed-loop characteristic equations given by:

$$\det(s\mathbb{I} - A + BK) = 0$$

By equalizing these two equations we get the following gains:

$$K_1 = 1.1146$$
$$K_2 = 0.0326$$

Using this controller the time response for an input with an amplitude equal to 30 degrees is represented in Fig. 11.6. As it can be seen from this figure that the overshoot and the settling time are those we would like to have. It is important to notice the existence of the error at the steady state regime. This error can be eliminated if we add an integral action in the loop. For more detail on this we refer the reader to [1].

For the second case since we don't have access to the load speed we can either compute it from the position or use an observer to estimate the system state. As it was said earlier, the poles that we use for the design of the observer should be faster than those used in the controller design.

Choosing the following poles ($s_{1,2}$ 4 times the real part of those used in the design of the controller):

$$s_{1,2} = -4\zeta w_n \pm j w_n \sqrt{1 - \zeta^2}$$
$$= -83.4218 \pm 20.8618 j$$
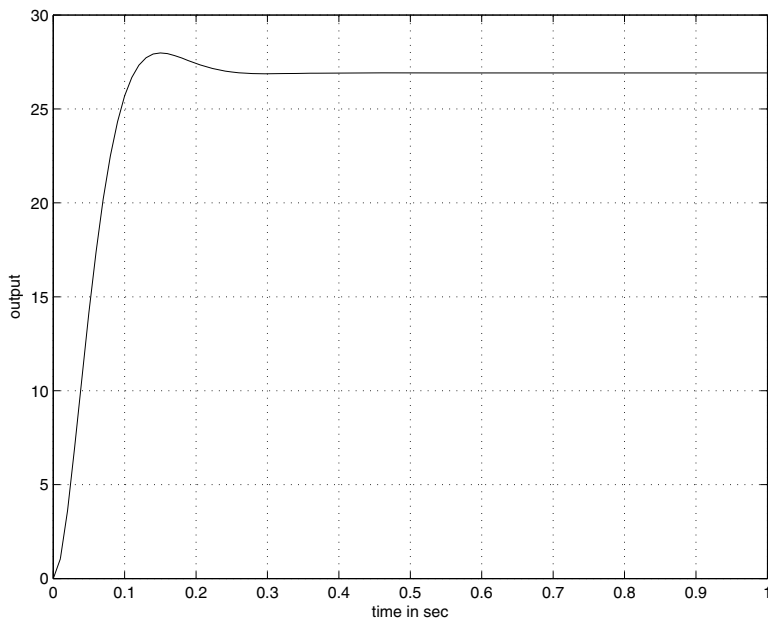
Fig. 11.6 Time response for a step function with 30 degrees as amplitude

we get the following gains for the observer:

$$L_1 = 151.2$$
$$L_2 = 5029.4$$

The gains of the controller remain the same as for the complete access case to the state vector.

In the following Matlab we provide the design of the controller and the observer at the same time and give simulation that shows the behavior of the states of the system and observer with respect to time.

```
        clear all

%data
tau=0.064
k=48.9
A = [0 1;0 -1/tau];
B = [0 ; k/tau];
C = [1 0];
D = 0;

% controller design
K = acker(A,B,[-3+3*j -3-3*j]);
L = acker(A',C',[-12+3*j -12-3*j])';
```

```
% Simualation data
Ts = 0.01;
x0 = [1 ; 1];
z0 = [1.1 ; 0.9];
Tf = 2;    %final time

%augmented system
Ah = [A            -B*K;
      L*C     A-B*K-L*C];
Bh = zeros(size(Ah,1),1);
Ch = [C D*K];
Dh = zeros(size(Ch,1),1);
xh0 = [x0 ; z0];
t=0:Ts:Tf;
u = zeros(size(t));
m = ss(Ah,Bh,Ch,Dh);

%simulation
[y,t,x] = lsim(m,u,t,xh0);

%plotting
figure;
plot(t,y);
title('Output');
xlabel('Time in sec')
ylabel('Output')
grid

figure;
plot(t,x(:,1:size(A,1)));
title('States of the system');
xlabel('Time in sec')
ylabel('System states')
grid

figure;
plot(t,x(:,size(A,1)+1:end));
title('states of the observer');
xlabel('Time in sec')
ylabel('Observer states')
grid
```

Figs. (11.7)-(11.9) gives an illustration of the output, the system's states and the observer's states.
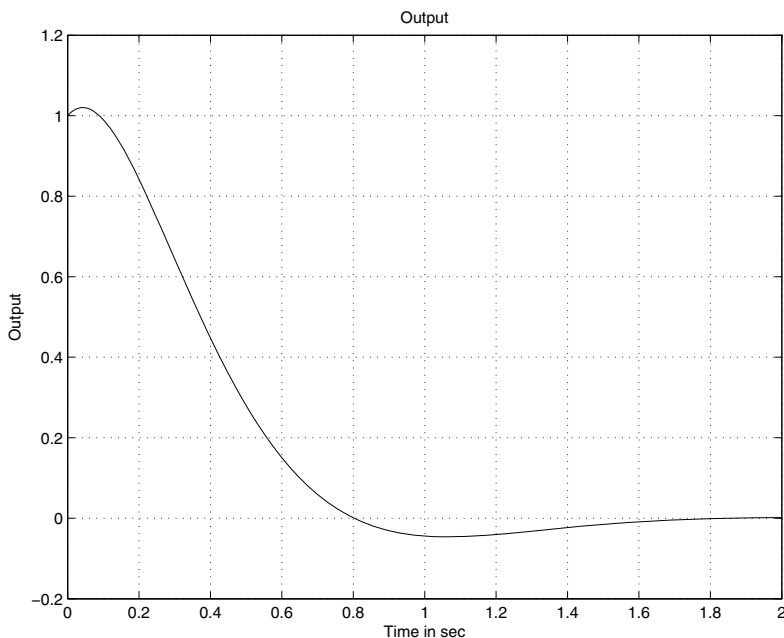
Fig. 11.7 Output versus time

We can also design the state feedback controller using the linear quadratic regulator. In fact, if we chose the following matrices for the cost function:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$$
$$R = 10$$

Remark 11.3.2 *In general, there is no magic rule for the choice the matrices for the cost function. But in general the fact that we use a high value for the control for instance will force the control to take small values and may prevent saturation.*

Using these matrices and the Matlab function, **lqr**, we get:

$$K = \begin{bmatrix} 0.3162 & 0.6875 \end{bmatrix}.$$

We can also design a state feedback controller using the results on robust control part. Since the system has no uncertainties and there is no external disturbance, we can design a state feedback controller for the nominal dynamics. Using the system data and Maltlab, we get:

$$X = \begin{bmatrix} 1.1358 & -0.3758 \\ -0.3758 & 1.1465 \end{bmatrix}$$
$$Y = \begin{bmatrix} -0.0092 & 0.0228 \end{bmatrix}$$
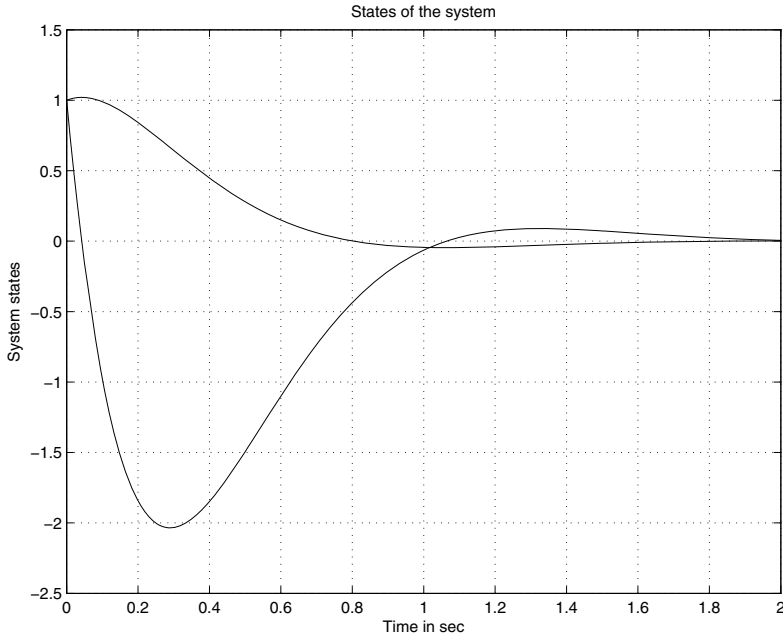
Fig. 11.8 System's states versus time

The corresponding controller gain is given:

$$K = \begin{bmatrix} -0.0017 & 0.0193 \end{bmatrix}.$$

Remark 11.3.3 *Since we have the continuous-time model for the dc motor kit, we have use it to design the controller gain. In this case we have solved the following LMI:*

$$AX + XA^\top + BY + Y^\top B^\top < 0$$

The gain K is given by: $K = YX^{-1}$.

For more detail on the continuous time case we refer the reader to Boukas [2] and the references therein.

11.4   Balancing Robot Control

The balancing robot is a challenging system from the control perspective since it is an unstable system in open-loop. This system has attracted a lot of researchers and many designs have been proposed for this purpose. Here we will present the design developed and tested in mechatronic laboratory at École Polytechnique de Montréal.
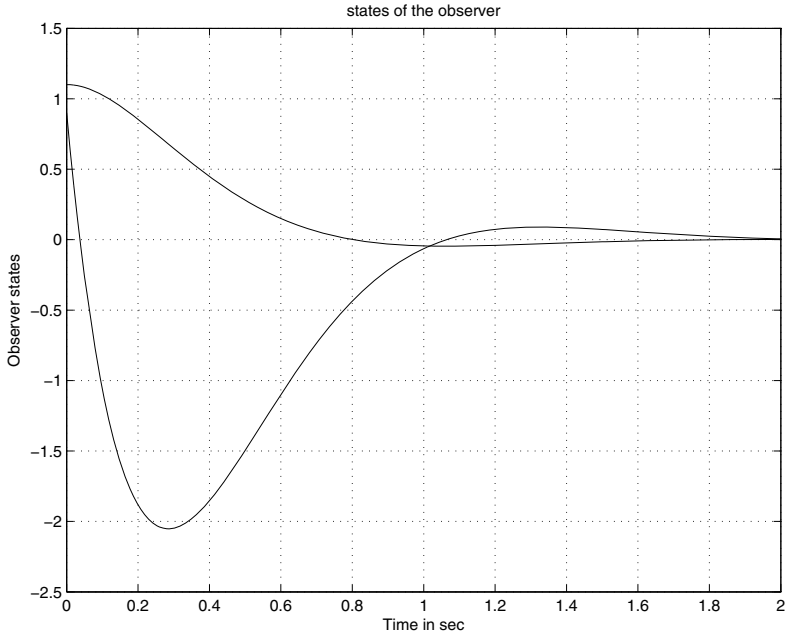
Fig. 11.9 Observer's states versus time

Figs. 11.10-11.11 give an idea of the robot. It was developed for research purpose and to allow the mechatronics students to implement their control algorithm and get familiar with complex system. The robot has two independent wheels each one driven by a dc motor via a gear with a ratio of 1:6. Each motor has an encoder to measure the speed of the shaft. The two motors are attached to the body of the robot. Other sensors like the accelerometer and gyroscope are used to measure the tilt angle. Appropriate filters are introduced to eliminate the noises of the measures and therefore get a useful signal for control.

The brain of the robot is built around a Microchip dsPIC of the family 30F4011. All the programming is done in C and inserted in the dsPIC, after producing the executable code by C30 of Microchip, using the PCKit2.

If we refer to Chapter 4, the mathematical model is given by:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$
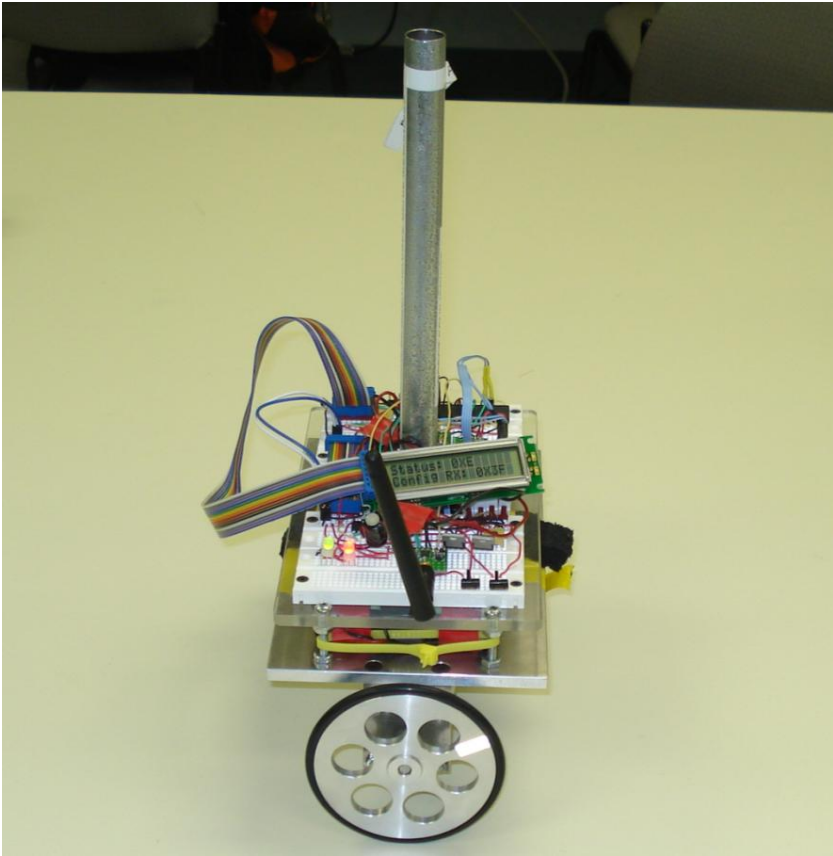
Fig. 11.10 Balancing robot

where

$$x(t) = \begin{bmatrix} \psi(t) \\ \dot{\psi}(t) \\ x(t) \\ \dot{x}(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 147.2931 & -0.4864 & 0 & -10.6325 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0429 & 0 & -0.9371 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 1.4687 \\ 0 \\ 0.1295 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$
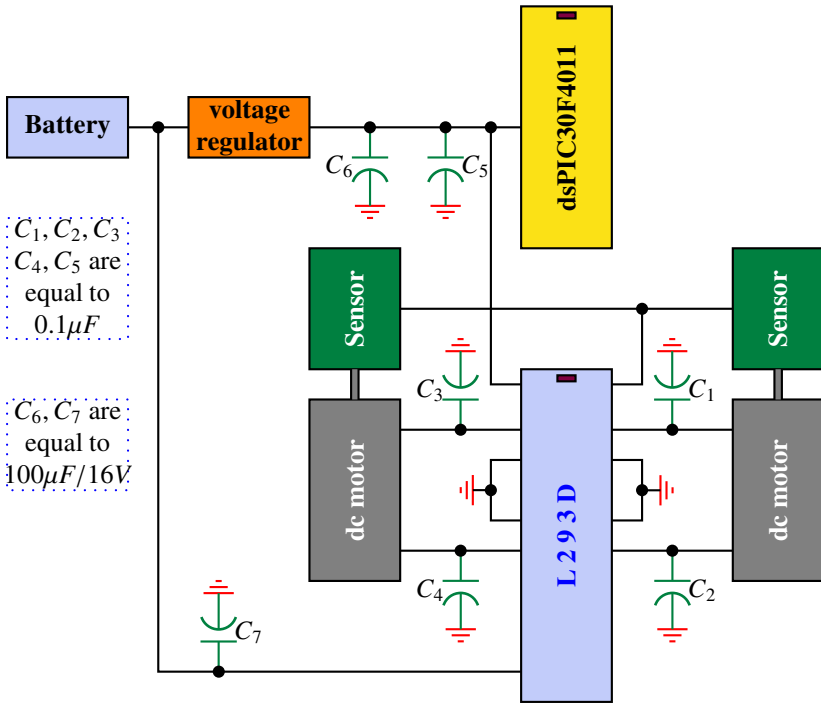
Fig. 11.11 Electronic circuit of the balancing robot

Since the system is unstable in open loop, let us design a state feedback controller that gives the following performances:

1. the system is stable in closed-loop;

2. overshoot less or equal to 5 %;

3. a settling time at 2 % equal to 1.5 s;

From the specifications we get:

$$\zeta = 0.707$$
$$w = \frac{4}{\zeta \times 1.5} = 3.7718 \text{ rad/s}$$

The corresponding dominant pair of poles is given by:

$$s_{1,2} = -2.6667 \pm 2.6675 j.$$

Since the matrix A is of rank four, we need to place two more poles to determine the state feedback controller gain, K. Let us chose the following dominated poles:

$$s_3 = -13.3335$$
$$s_4 = -13.3335$$

Using the function *acker*, we get the following gain:

$$K = \begin{bmatrix} 339.5604 & 27.5946 & -132.5973 & -76.8450 \end{bmatrix}.$$
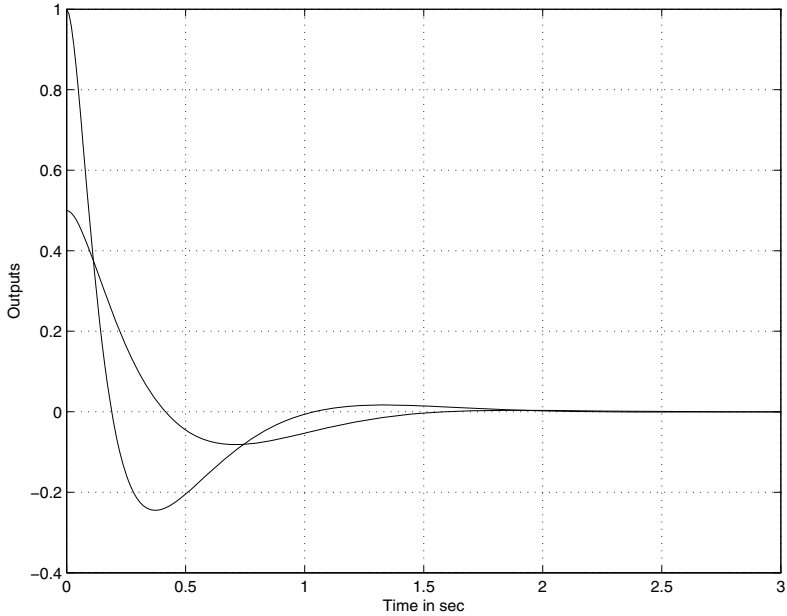
Fig. 11.12 Outputs versus time

The simulation results with this controller are illustrated in Figs. 11.12-11.13. The system starts from an initial condition $x_0^\top = \begin{bmatrix} 1 & 0 & 0.5 & 0 \end{bmatrix}$ with a zero input. If we try to send an input reference we will have errors in states or in outputs. To overcome this an integral action needs to be added. If we denote by $\tilde{x}(t) = \int_0^t (x_r(t) - x(t))\, dt$, with $x_r(t)$ is the position reference and following [1], we get:

$$\dot{\eta}(t) = \tilde{A}\eta(t) + \tilde{B}u(t)$$
$$y(t) = \tilde{C}\eta(t)$$
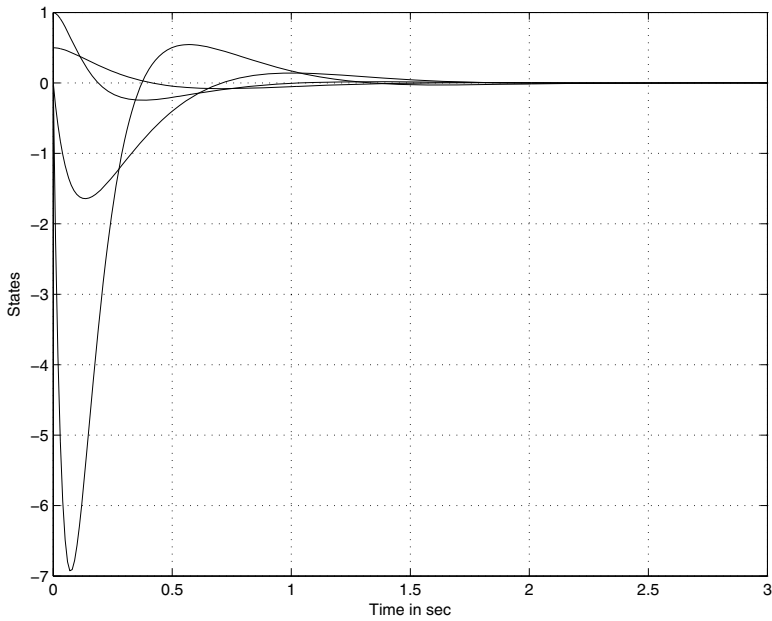
Fig. 11.13 States versus time

where

$$\eta(t) \;=\; \begin{bmatrix} x(t) \\ \tilde{x}(t) \end{bmatrix},$$

$$\tilde{A} \;=\; \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix},$$

$$\tilde{B} \;=\; \begin{bmatrix} B \\ 0 \end{bmatrix},$$

$$\tilde{C} \;=\; \begin{bmatrix} C & 0 \end{bmatrix}$$

The new dynamics become of order five and we need to fix three dominated poles plus the dominating poles that come from the specifications. These poles are fixed to the following ones:

$$s_{3,4} \;=\; -13.3335 \pm 2.6675\,j$$
$$s_5 \;=\; -13.3335$$

Using the function *acker*, we get the following gain:

$$K \;=\; \begin{bmatrix} 339.5604 & 27.5946 & -132.5973 & -76.8450 & 0.1 \end{bmatrix}.$$

We can also design a state feedback controller using the linear quadratic control technique. In fact if we chose the following matrices:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 20 \end{bmatrix}$$

$$R = \begin{bmatrix} 10 \end{bmatrix}$$

and solving the Ricatti equation using the Matlab function *lqr*, we get:

$$K = \begin{bmatrix} 218.9494 & 17.7264 & -1.0000 & -15.7658 \end{bmatrix}$$

The corresponding eigenvalues for the closed-loop are given by:

$$s_1 = -12.7439$$
$$s_2 = -11.5936$$
$$s_3 = -0.9420$$
$$s_4 = -0.1371$$

The simulation results with this controller can be obtained similarly and the detail is omitted.

For this system, we can design also a state feedback controller using the robust control theory. This can be done either in continuous-time or discrete-time. Since our model is in continuous-time we will do the design using the LMI in continuous-time. Solving the appropriate LMI (the one we used for the dc motor kit with different sizes for the matrices, we get:

$$X = \begin{bmatrix} 0.0595 & -0.2156 & 0.1053 & 0.0177 \\ -0.2156 & 1.9238 & -0.0705 & 0.7732 \\ 0.1053 & -0.0705 & 1.4496 & -0.3439 \\ 0.0177 & 0.7732 & -0.3439 & 1.5028 \end{bmatrix}$$

$$Y = \begin{bmatrix} -7.2149 & 27.5211 & -13.6001 & 7.4801 \end{bmatrix}$$

which gives the following gain for the state feedback controller:

$$K = \begin{bmatrix} -174.2563 & -10.5741 & 6.0437 & 13.8536 \end{bmatrix}$$

The corresponding eigenvalues for the closed-loop are given by:

$$s_{1,2} = -6.9770 \pm 6.4079j$$
$$s_{3,4} = -0.6028 \pm 0.9598j$$

The advantage of this method is that we don't have to provide poles as for the case of pole placement technique. On the top of this the LMI technique can handle more appropriately the presence of saturations in the input if it is the case.

As a lst example, let consider the $\mathcal{H}_\infty$ control problem for the two wheels robot. In this case, we add a term in the state dynamic. This term is $Bw(t)$ where $w(t)$ is

the external disturbance that has finite energy. Solving the appropriate LMI with $\gamma = 0.1$, we get:

$$X = \begin{bmatrix} 1.1233 & -2.7517 & 0.2742 & 1.2667 \\ -2.7517 & 61.0512 & -1.5973 & 14.4055 \\ 0.2742 & -1.5973 & 2.0232 & -3.4932 \\ 1.2667 & 14.4055 & -3.4932 & 16.4930 \end{bmatrix}$$

$$Y = \begin{bmatrix} -143.3219 & 316.5973 & -60.7376 & -33.8808 \end{bmatrix}$$

which gives the following gain for the state feedback controller:

$$K = \begin{bmatrix} -193.7547 & -9.4711 & 39.7088 & 29.5092 \end{bmatrix}$$

The corresponding eigenvalues for the closed-loop are given by:

$$s_{1,2} = -3.8372 \pm 8.9414 j$$
$$s_{3,4} = -1.9189 \pm 2.0781 j$$

For the design of other controllers can be obtained easily and we let this as an exercise for the reader since the design is brought to write a program of Matlab similar to those we give in the text.

11.5   Magnetic Levitation System

In this section we will present the magnetic levitation system we presented earlier. This mechatronic system developed in our mechatronic laboratory is composed of two parts: a fixed one that represents the coil and that generates the electromagnetic force and a ferromagnetic object which we would like to place at a certain position by acting on the electromagnetic force generated by the coil. The objective of the system is to control the vertical position of the moving object by adjusting the current in the electromagnet through the input voltage. The object position is measured using a Hall effect sensor. An elecgronic circuit build around a dsPIC30F4011 supplies the coil through an L298, an integrate circuit, with a current that is proportional to the command voltage of the actuator. As the magnetic force can be only attractive, the mutual conductance amplier turns off for negative commands. This system is illustrated by Fig. 11.14.

The mathematical model for this system is given by the following equation:

$$m\ddot{l}(t) = mg - F_1 - F_2$$

where m is the mass of the moving object, $l(t) \in \mathbb{R}^+$ is the distance measured from the electromagnet, F_1 and F_2 are respectively the force generated by the coil when the current is $i(t)$ and the electromagnetic force between the electromagnet and permanent magnet placed of the head of the moving object.

The expression of these force are given by:

$$F_1 = K_1 \frac{i^2(t)}{l^2(t)}$$

$$F_2 = K_2 \frac{1}{l^2(t)}$$

This model in nonlinear that we can linearize to get the following (see Chapter 1):

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

where

$$x(t) = \begin{bmatrix} x_1(t)(position) \\ x_2(t)(velocity) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 \\ \frac{2[sign(u_2)k_c u_e^2 + k_p R^2]}{mR^2 x_e^3} & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \frac{-2sign(u)k_c u_e}{mR^2 x_e^2} \end{bmatrix}$$

$$C = \begin{bmatrix} \frac{-3C_p}{0.032 x_e^4} & 0 \end{bmatrix}$$

$$D = \frac{C_b}{0.032R}$$

The data of the system is given by Table 11.1. Using this data, the matrices are given by:

$$A = \begin{bmatrix} 0 & 1 \\ 2490.8 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ -1.2711 \end{bmatrix},$$

$$C = \begin{bmatrix} 473.5711 & 0 \end{bmatrix},$$

$$D = \begin{bmatrix} -0.0833 \end{bmatrix},$$

It is important to notice that the system is unstable in open loop since it has a pole with a positive real part. This can be checked by computing the eigenvalues of the matrix A.

Let us design a state feedback controller to guarantee the following performances:

1. the system is stable in closed-loop

2. an overshoot less or equal to 0.2 %

3. a settling time at % 2 equal to 0.05 sec
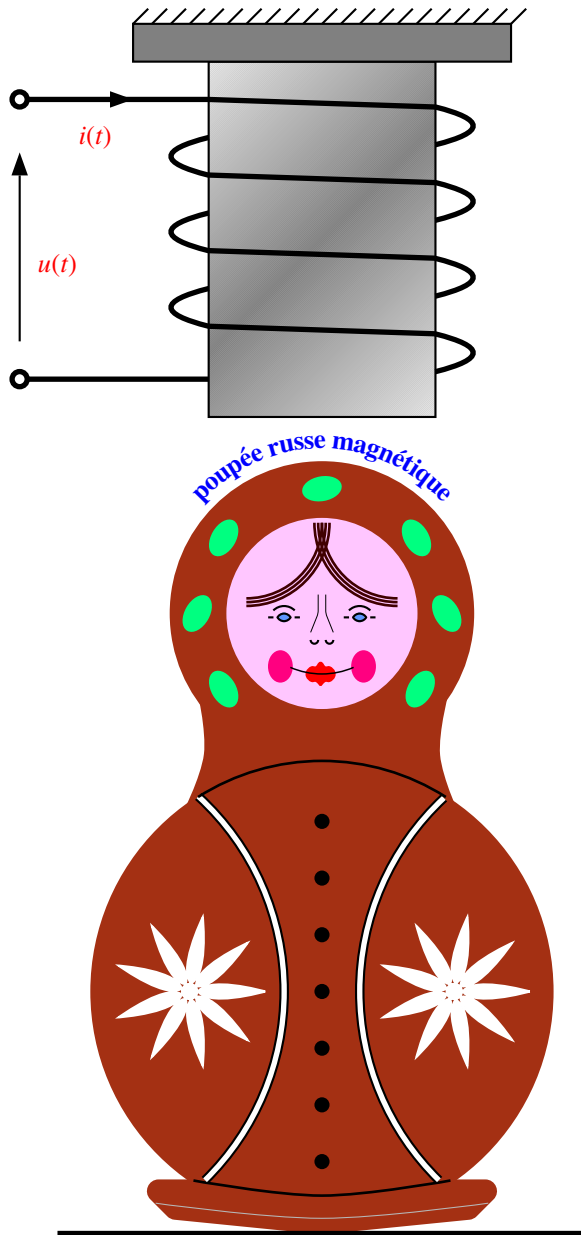
Fig. 11.14 Magnetic levitatios system

Table 11.1 Data of the magnetic levitation system

Variable	value
R	62.7 Ω
L	60 mH
m (object mass)	7.64 g
k_c	5.9218 10^{-4}
k_p	4.0477 10^{-6}
C_b	-0.1671
C_p	$-1.9446\ 10^{-8}$
diameter of the permanent magnet	9 mm

Since the overshoot is less or equal to 0.2 %, it results that $\zeta = 0.9$. The time settling time at 2 % is given by:

$$t_s = \frac{4}{\zeta w_n}$$

where w_n is natural pulse.

If we fix the settling time at 0.05 sec we get:

$$w_n = \frac{4}{0.05 \times 0.9} = 88.8889$$

The dominant poles for the design are then given by:

$$s_{1,2} = -\zeta w_n \pm j w_n \sqrt{1 - \zeta^2} = -80.000 \pm 38.75 j$$

Using this pair of poles we get:

$$K_1 = -175.6$$
$$K_2 = -125.9$$

Using this controller the time response starting from given initial conditions is represented in Fig. 11.15. As it can seen from this figure that the overshoot and the settling time are those we would like to have.

For the second case since we don't have access to the load speed we can either compute it from the position or use an observer to estimate the system state. As it was said earlier, the poles that we use for the design of the observer should be faster than those used in the controller design.

Choosing the following poles ($s_{1,2}$ 4 times the real part of those used in the design of the controller):

$$s_{1,2} = -4\zeta w_n \pm j w_n \sqrt{1 - \zeta^2}$$
$$= -320.0 \pm 38.75 j$$
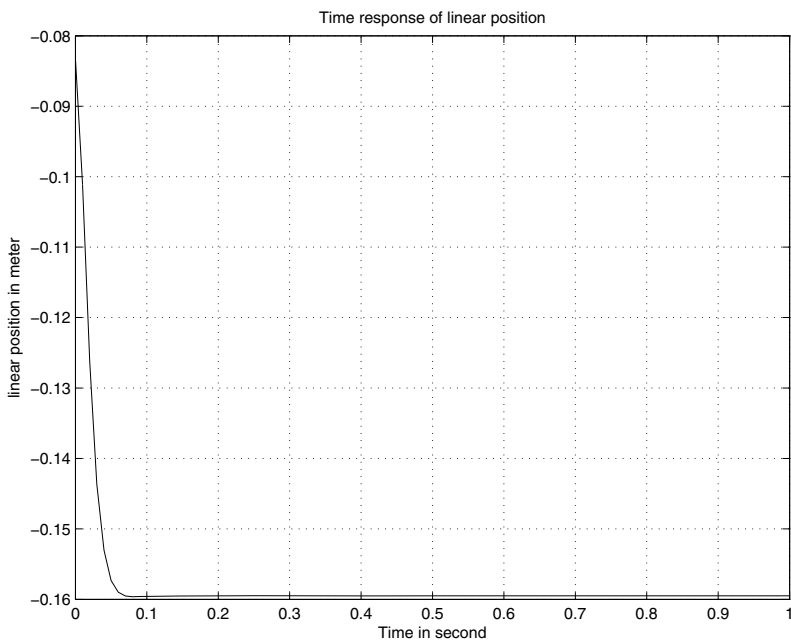
Fig. 11.15 Time response for moving object

we get the following gains for the observer:

$$L_1 \; = 1.351$$
$$L_2 \; = 224.5927$$

We can experiment all the other controller as we did for the dc motor kit and the two wheels robot but we prefer let this part as exercise for the reader to practice the tools. Notice we invite him/her to make the design in continuous-time and discrete-time cases and compare the results. The dimension of this system allows that.

A sample for programming the state feedback control of this system is given bellow:

```
 #include <p30fxxxx.h>
#include <stdio.h>
#include <stdlib.h>
#include <adc10.h>
#include <math.h>
#include <uart.h>

//
// Configuration
//
```

```c
//  Interne frequency (30 MIPS) instructions/sec
// Number of samples: 7,37*16/4 = 29480000

_FOSC(CSW_FSCM_OFF & FRC_PLL16);
_FWDT(WDT_OFF);
_FBORPOR(PBOR_OFF & MCLR_DIS);
_FGS(CODE_PROT_OFF);
_FICD( ICS_NONE );
__C30_UART=2;


//
//  Variables

    #define Freq_pic 29480000  // PIC Frequency

    #define a11 7.8510061454215840e-001
    #define a12 2.2727760074413661e-004
    #define a21 5.0829248960838420e+001
    #define a22 1.0000643300984893e+000
    #define b11 3.7771272752681438e-005
    #define b12 4.5392069137012870e-004
    #define b21 8.7496385285734044e-003
    #define b22 1.0852723324638587e-001

    #define Ts 2.2727272727272727e-004
    #define u_max 1.1789999999999999e+001
    #define ref_tension 5.0000000000000000e+000
    #define ref_pic 1.0240000000000000e+003
    #define duty_cycle_ref 5.8481764206955049e+001
    #define x_ref 7.8768775539549939e-003
    #define u_ref 2.0000000000000000e+000
    #define y_ref 8.5691877396730676e-001
    #define K0 5.2128678707944724e+004
    #define K1 3.9336557697049994e+002


        double y[2] = {0.0, 0.0};
        double u[2] = {0.0, 0.0};
        double y_tilde[2] = {0.0, 0.0};
        double tension_tilde[2] = {0.0, 0.0};
        double tension = 0.0;
        double duty_cycle_tilde = 0.0;
        double lim_Sup = 0.0;
        double lim_Inf = 0.0;
        double position_tilde[2] = {0.0, 0.0};
```

```
        double vitesse_tilde[2] = {0.0, 0.0};
        double integrale_tilde[2] = {0.0, 0.0};
        double duty_cycle = 0.0;
        double temps_total = 0.0;
        double n = 6553500.0/65536.0;
        int compteur = 0;
        int compteur_freq = 0;
        int uart_flag = 1;
        unsigned long Val_reg = 0;

//
// Functions
//

void init(void){

INTCON1bits.NSTDIS=0; // Activation of the level of interruption
TRISE = 0;  // Configuration of PORTE as output
TRISD = 0;  //Configuration of PORTD as output
PORTEbits.RE8 = 1;
PORTEbits.RE2 = 0;
PORTDbits.RD0 = 1;
ADPCFG= 0xFFFF; // Configuration of the pins of PORTB as digital
I/O
}

void init_ADC (void){

SetChanADC10(ADC_CHX_POS_SAMPLEA_AN3AN4AN5 &
ADC_CHX_NEG_SAMPLEA_NVREF);

ConfigIntADC10(ADC_INT_DISABLE);

OpenADC10(ADC_MODULE_ON & ADC_IDLE_CONTINUE & ADC_FORMAT_INTG
 & ADC_CLK_AUTO & ADC_AUTO_SAMPLING_ON & ADC_SAMPLE_SIMULTANEOUS,
 ADC_VREF_AVDD_AVSS & ADC_SCAN_OFF & ADC_CONVERT_CH_0ABC
 & ADC_SAMPLES_PER_INT_1 & ADC_ALT_BUF_OFF & ADC_ALT_INPUT_OFF,
 ADC_SAMPLE_TIME_1 & ADC_CONV_CLK_SYSTEM  & ADC_CONV_CLK_32Tcy,
 ENABLE_AN4_ANA & ENABLE_AN5_ANA, SCAN_NONE);
}

void init_Timer1 (void){

INTCON1bits.NSTDIS=0; // Activation of mode 16 bits of the Timer1
T1CONbits.TON = 1; // Autorisation du Timer1
```

```
T1CONbits.TGATE = 0; // Dsactivation du mode Timer Gate
T1CONbits.TSIDL=1;  // Synchronisation of Timer1 sur le Idle mode
T1CONbits.TCKPS = 0;   // Choice of the Prescaler 1:1
                          (1=1:8, 2=1:64)
T1CONbits.TCS=0;     // Selection of the interne clock (0=FOSC/4)
IFS0bits.T1IF = 0; // Put to zero the overflow bit for the
                        interrupt of Timer1
IEC0bits.T1IE = 1; // Activation of the interruption of Timer1
PR1 = 6699;  // Sampling frequency at 4400 Hz environ
IPC0bits.T1IP = 5;  // Priority 5 for the interruption of the
                        Timer1

}

/* ROUTINE D'INITIALISATION DE L'UART */

void init_UART (void){

ConfigIntUART2(UART_RX_INT_DIS & UART_RX_INT_PR0
& UART_TX_INT_DIS & UART_TX_INT_PR0);

    // Configuration of the register U2MODE

U2MODEbits.UARTEN = 1; // UART pins controlled by UART
U2MODEbits.USIDL = 0;  // UART communication continue in
                          Idle Mode

U2MODEbits.WAKE = 1;   // Wake up enable in sleep Mode
U2MODEbits.LPBACK = 0; // Loopback mode disabled
U2MODEbits.ABAUD = 0;  // Autobaud process disabled
U2MODEbits.PDSEL = 0;  // 8-bit data,  no parity
U2MODEbits.STSEL  = 0; // 1 stop-bit.

// Configuration du registre U2STA

U2STAbits.UTXISEL = 0;  // Transmission Interrupt Mode
                           Selection bit
U2STAbits.UTXBRK = 0;   // UxTX pin operates normally
U2STAbits.UTXEN = 1;    // Transmit enable
U2STAbits.URXISEL = 1;  // Interrupt occurs when one charater
                           is received
U2STAbits.ADDEN = 0;    // Address detect disabled

U2BRG = 31;             // Value for 57600 bps baudrate
}
```

```
//
// Initialization of the complementary  mode PWM
//
void init_PWM (void){

Val_reg = 1023;  // Frquence de 30000 Hz environ
lim_Sup = (u_max*(2*Val_reg + 1)/(2*Val_reg + 2)) - u_ref;
lim_Inf = -u_max - u_ref;

PTCONbits.PTEN = 1;  // Activation of the time base
PTCONbits.PTSIDL = 1;  // Configuration in Idle Mode
PTCONbits.PTCKPS = 0; // Selection de 4TCY ( Prescale: 00 = 1:1;
                        01= 1:4; 10 = 1:16; 11 = 1:64)
PTCONbits.PTMOD = 0;  // Selection of the free running mode

PTMRbits.PTDIR = 0; // Increment of the time base
PTMRbits.PTMR = Val_reg; // Register value of the Time base

PTPER = Val_reg;  // Value of the signal period

PWMCON1bits.PMOD1 = 0;  // Selection the mode PWM complementary
PWMCON1bits.PEN1H = 1; // Activation of the pins in mode PWM
PWMCON1bits.PEN1L = 1; // Activation of the pins in mode PWM

DTCON1bits.DTAPS = 0; // Time base unit is 1TCY
DTCON1bits.DTA = 0;  // Value of the DT for the unity A
PDC1 = 0; //  zero of the dutycycle

}

void __attribute__((interrupt, auto_psv)) _T1Interrupt (void){

if (IFS0bits.T1IF){

PORTEbits.RE2 = !PORTEbits.RE2;
PDC1 = (2.0 * (Val_reg + 1) * duty_cycle)/100.0; // Calcul de la
            valeur du registre PDC1
y[0] = (ReadADC10(2)*ref_tension)/ref_pic; // Signal of the
                                            sensor in Volts
y_tilde[0] = y[0] - y_ref;


position_tilde[1] = position_tilde[0];
vitesse_tilde[1] = vitesse_tilde[0];
```

```
integrale_tilde[1] = integrale_tilde[0];
y_tilde[1] = y_tilde[0];
tension_tilde[1] = tension_tilde[0];

position_tilde[0] = (a11*position_tilde[1]+a12*vitesse_tilde[1]
                     +b11*tension_tilde[1]+b12*y_tilde[1]);
vitesse_tilde[0] = (-a21*position_tilde[1]+a22*vitesse_tilde[1]
                    +b21*tension_tilde[1]+b22*y_tilde[1]);
tension_tilde[0] = (K0*position_tilde[0]) +
                   (K1*vitesse_tilde[0]);// + N*ref;

if(tension_tilde[0]>lim_Sup){tension_tilde[0] = lim_Sup;}
// Saturation of the tension tilde
if(tension_tilde[0]<lim_Inf){tension_tilde[0] = lim_Inf;}

tension = u_ref + tension_tilde[0];
duty_cycle_tilde = tension_tilde[0]*(50.0/u_max);
duty_cycle = duty_cycle_ref + duty_cycle_tilde; // Computation
                    of the duty cycle in percentage

temps_total += Ts;

compteur_freq = 0;

compteur++;
if(compteur == 10){ // Print data every 1 ms
compteur = 0.0;
uart_flag = 1;
}

IFS0bits.T1IF = 0; // put to zero of the overflow bit
}
}

/* PROGRAMME PRINCIPAL */

int main (void){

    init();
    init_PWM();
    init_ADC();
    init_UART();
    init_Timer1();
    while(1){
        if (uart_flag){
```

```
    printf("%lf %lf %lf %lf %lf %lf %lf\n\r",
    temps_total, tension, u_ref, y[0], y_ref,
    position_tilde[0], x_ref);
    uart_flag = 0;
    }

  }
}
```

11.6   Conclusion

This chapter covered some case studies that were developed in our mechatronics laboratory at École Polytechnique de Montréal. We covered all the steps for the design of a mechatronic system with different kind of details. The emphasis is put on the design of the control algorithm of each presented system.

11.7   Problems

1. Let us consider a dynamical discrete-time system with the following data:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -2 & -3 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix},$$

$$D = \begin{bmatrix} 0 \end{bmatrix}$$

with the following norm bounded uncertainties:

$$D_A = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix},$$

$$E_A = \begin{bmatrix} 0.1 & -0.2 & -0.1 \end{bmatrix},$$

$$D_B = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix},$$

$$E_B = \begin{bmatrix} 0.1 & -0.2 \end{bmatrix},$$

$$D_C = \begin{bmatrix} 0.1 \end{bmatrix},$$

$$E_C = \begin{bmatrix} 0.1 & 0.2 & -0.1 \end{bmatrix},$$

(a) design for the nominal system the following controller:
   i. state feedback controller
   ii. static output feedback controller
   iii. dynamic output feedback controller
(b) design for the uncertain system the following controller:
   i. state feedback controller
   ii. static output feedback controller
   iii. dynamic output feedback controller
(c) if we have an extra term in the state dynamics that add external disturbance: $x(k + 1) = [A + \Delta A] x(k) + [B + \Delta B] u(k) + B_w w(k)$. Design the controllers (state feedback, static output, dynamic output feedback) for the nominal and uncertain system that assure the $\mathcal{H}_\infty$ performance.
(d) design the state-feeback, static output feedback and dynamic feedback controllers that assure the guaranteed cost for all admissible uncertainties.

2. In this problem we invite you to proceed with the design of a small boat that you can control using a joystick to make it moving in a small lac for instance.

   (a) give a schematic design (batteries, motors, etc.)
   (b) establish the mathematical model
   (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

3. In this problem we invite you to proceed with the design of a small aircraft that you can control using a joystick to make it flying.

   (a) give a schematic design (batteries, motor, etc.)
   (b) establish the mathematical model
   (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

4. In this problem, we ask for the design of a vacuum cleaner. This device should be automatic and avoid obstacle in its environment. It is also important to design a cheap one that can communicate wireless via an emitter and receiver and a joystick

    (a) give a schematic design (electronic circuit, motor, etc.)
    (b) establish the mathematical model
    (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

5. In this problem we invite you to proceed with the design of a mechatronic system that control the position of a small ball of ping pong. Pressed air can be used to position the ball.

    (a) give a schematic design (electronic circuit, motor, etc.)
    (b) establish the mathematical model
    (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

6. In this problem we would like to design a one leg robot that can move using one wheel while remaining in a vertical position. Provide the design of such mechatronic system.

    (a) give a schematic design (electronic circuit, motor, etc.)
    (b) establish the mathematical model
    (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

7. Solar energy is an alternate source of power that can be used. In this problem we ask you to design a solar system that maximize the energy generated by the solar panel.

    (a) give a schematic design (electronic circuit, motor, etc.)
    (b) establish the mathematical model
    (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

8. In this problem we ask to design a hoover that can be controlled to seal on water via a emitter and a receiver using a joystick.

    (a) give a schematic design (electronic circuit, motor, etc.)
    (b) establish the mathematical model
    (c) fix the specifications you would like to have and design the appropriate controller that can give such performances

AppendixA
C Language Tutorial

The aim of this appendix is to review the C language to refresh the memory of the reader and help him to start writing his programs without reading huge books on the subject. Our intention is not replace these books and readers that are not familiar with the subject are strongly encouraged to consult book of Kernighan and Ritchie[1].

Firstly we invite the reader to download a C compiler and a text editor. To experiment the different programs in C we will give, the reader must type the programs, save them and then compile them. For more details on how to do this we refer the reader to the manual of the used C compiler.

To start our tutorial, let us consider our first simple program. This program is intended to write "Welcome to mechatronics course". The listing of this program is given by the following lines:

```
#include < stdio.h>

void main()
    {
        printf("\nWelcome to mechatronics course\n");
    }
```

To see the output of this simple program we need to have an editor and a C compiler.

---

[1] Brian W. C. Kernighan and Dennis M. Ritchie, The C Programming Language– ANSI C, Prentice Hall, 1988

Each C program is composed by variables and functions and must have a "main" function. Except some reserved words all the variables and the functions must be declared before they can be used. The variables can take one of the following types:

- *integer*

- *real*

- *character*

- etc.

The functions specify the tasks the program has to perform and for which it is designed for. The "main" function establishes the overall logic of the code.

Let us examine the previous program. The first statement

```
#include < stdio.h>
```

includes a specification of the C I/O library. The ".h" files are by convention "header files" which contain definitions of variables and functions necessary for the functioning of a program, whether it can be a user-written section of code, or as part of the standard C libaries.

The directive

```
#include
```

tells the C compiler to insert the contents of the specified file at that point in the code.

The notation

```
<...>
```

instructs the C compiler to look for the file in certain "standard" system directories.

The *void* preceeding "main" indicates that main is of "void" type–that is, it has no type associated with it, which means in another sense that it cannot return a result during the execution.

The ";" denotes the end of a statement. Blocks of statements are put in braces {···}, as in the definition of functions. All C statements are defined in free format, i.e., with no specified layout or column assignment. White space (tabs or spaces) is never significant, except inside quotes as part of a character string.

The statement *printf* line prints the message "Welcome to mechatronics course" on "stdout" (the output stream corresponding to the X-terminal window in which you run the code), while the statement

```
\n
```

prints a "new line" character, which brings the cursor onto the next line. By construction, the function printf never inserts this character on its own and this is let to the programmer.

The standard C language has some reserved keywords that can not be used by the programmer for naming variables or other purpose and he must used them as it is suggested otherwise mistakes will appear during the compilation. These keywords are listed in the Table A.1.

Table A.1 List of C language keywords

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Table A.2 Number representations

Base	Representation	Chiffres permis	Example
Decimal (10)		0123456789	5
Binary (2)	0b...	01	0b10101010
Octal (8)	0...	01234567	05
Hexadecimal (16)	0x...	0123456789ABCDEF	0x5A

Table A.3 Integer representations

Type	Size (bits)	Min	Max
char, signed char	8	-128	+127
unsigned char	8	0	255
short, signed short	16	-32768	+32767
unsigned short	16	0	65535
int, signed int	16	-32768	+32767
unsigned int	16	0	65535
long, signed long	32	-2^{31}	$+2^{31} - 1$
unsigned long	32	0	$2^{32} - 1$
long long, signed long long	64	-2^{63}	$+2^{63} - 1$
unsigned long long	64	0	$2^{64} - 1$

Table A.4 Decimal representations

Type	Taille (bits)	Emin	Emax	Nmin	Nmax
float	32	-128	+127	2^{-126}	2^{128}
double	32	-128	+127	2^{-126}	2^{128}
long double	64	-1022	+1023	2^{-1022}	2^{1024}

Let us look to the constants and variables. The constants have to be defined before it can be used. The structure we used to define these constant is:

```
#define name value
```

The word name is the one we give to the constant and the value is the value that this constant takes. The following example gives some constants:

```
#define acceleration 9.81
#define pi      3.14
#define mot "welcome to mechatronics course''
#define False 0
```

In general when we manipulate data, we use different number representation. Table A.2 gives the bases we are usually using when programming microcontroller.

For the variables, we must declare them before also. The following syntax is used:

```
type <name>;
```

where the type is one of the following:

- int (for integer variable)

- short (for short integer)

- long (for long integer)

- float (for single precision real (floating point) variable)

- double (for double precision real (floating point) variable)

- char (for character variable (single byte))

Tables A.3 and A.4 gives the values taken in each type. It is important to mention that the compiler checks for consistency in the types of all variables used in the program to prevent mistakes. The following example shows some variables:

```
int i,j,k;
float x,y;
unsigned char var1;
unsigned char var2[10] = "welcome";
```

Once these variables and constants are defined what we can do with? The answer is that we can do many things like:

- arithmetic operations (act on one or multi variables)

- logic operations (act on one or multi variables)

- relational operations (act on one or multi variables)

- etc.

Tables A.5 and A.6 give an idea on the different operations we can do either on the constants or the variables.

Table A.5 Arithmetic operations

Symbol	Meaning	example
+	addition	$u[k] = u_1[k] + u_2[k];$
-	substrate	$u[k] = u_1[k] - u_2[k];$
/	division	$u[k] = u_1[k]/u_2[k];$ ($u_2[k]$ must be different from zero)
*	multiplication	$u[k] = K * x[k];$
%	modulo	$u[k] = u_1[k]\%u_2[k];$

Table A.6 Logic operations

Symbol	Meaning	example
&	ET	$u[k] = u_1[k]\&u_2[k];$
|	OR	$u[k] = u_1[k] \mid u_2[k];$
$\wedge$	XOR	$u[k] = u_1[k] \wedge u_2[k];$
$\sim$	inversion	$u[k] =\sim x[k];$
$\ll$	shift left	$u[k] = u_1[k] \ll u_2[k];$
$\ll$	shift right	$u[k] = u_1[k] \gg u_2[k];$

More often we need to print data wither on the screen or an LCD. For this purpose the *print* function can be used. This function can be instructed to print integers, floats and strings properly. The general syntax is

```
printf( "format", variables );
```

where "format" specifies the conversion specification and variables is a list of quantities to be printed.
The useful formats are:

```
%.nd   integer (optional n = number of columns; if 0, pad with zeroes)
%m.nf  float or double (optional m = number of columns,
n = number of decimal places)
%ns    string (optional n = number of columns)
```

```
%c        character
\n \t     to introduce new line or tab
\g    ring the bell (''beep'') on the terminal
```

In some programs, the concepts of loops are preferable to perform calculations that require repetitive actions on a stream of data or a region of memory. There are several ways to loop in the standard C and the following ways the most common used loops:

```
 // while loop
while (expression)
{
        block of statements to be executed
}

 // for loop
for (expression_1; expression_2; expression_3)
{
        block of statements to be executed
}
```

In some cases we need to take an action based on the realization of some condition or dependent on the value of a given variable. In this case the word *if* or the word *if else* and the switch can be used. The following structures are used:

```
if (conditional_1)
{
        block of statements to be executed when conditional_1 is true
}
else if (conditional_2)
{
        block of statements to be executed when conditional_2 is true
}
else
{
        block of statements to be executed otherwise
}
```

Tables A.7 and A.8 show the most used conditional operators that we may use in the expressions.

Table A.7 Logic operations

Symbol	Meaning	example
&&	and	*x&&y*
||	or	*x||y*
!	not	*x!y*

Table A.8 Logic operations

Symbol	Meaning	example
$<$	smaller than	$x < y$
$<=$	smaller than or equal to	$x <= y$
$==$	equal to	$x == y$
$! =$	not equal to	$x! = y$
$>=$	greater than or equal to	$x >= y$
$>$	greater than	$x > y$

```
switch (expression)
{
    case const_expression_1:
{
    block of statements to be executed
        break;
}
case const_expression_2:
{
    block of statements to be executed
        break;
}
default:
{
     block of statements
}
}
```

The C language allows the programmer to access directly the memory locations using the pointers. To show how this works, let us consider a variable Xposition defined as follows:

```
float Xposition;
Xposition = 1.5;
```

If for example we want to get the address of the variable Xposition, the following can be used:

```
float* pXposition;
float Xposition;

Xposition = 1.5;
pXposition = &Xposition;
```

In this code we define a pointer to objects of type float, Xposition and set it equal to the address of the variable Xposition.

To get the content of the memory location referenced by a pointer is obtained using the "*" operator (this is called dereferencing the pointer). Thus, *pXposition refers to the value of Xposition.

Arrays of any type play an important role in C. The syntax of the declaration of the arrays is simple and the following gives that:

```
type varname[dim];
```

where dim is the dimension we want to give to the array varname.

In standard C, the array begins with the position 0 and all its elements occupy adjacent locations in memory. Thus, if *matrix* is an array, $*matrix$ is the same thing as *matrix*[0], while $*(matrix + 1)$ is the same thing as *matrix*[1], and so on.

References

[1] Boukas, E.K.: Systèmes Asservis. Editions de l'Ecole Polytechnique de Montréal, Montréal (1995)
[2] Boukas, E.K.: Stochastic Switching Systems: Analysis and Design. Birkhauser, Boston (2005)
[3] Bryson Jr, A.E., Ho, Y.: Applied optimal control: optimization, estimation, and control. Blaisdell, Waltham (1969)
[4] Powell, J., Franklin, G., Workman, M.: Digital Control of Dynamic Systems. Addison Weley, New York (1998)
[5] Kailath, T.: Linear Systems. Prentice-Hall, New York (1980)
[6] Lozeau, M., Boukas, E.K.: Design and control of a balancing robot (2009)
[7] Rugh, W.J.: Linear System Theory. Prentice Hall, New Jersey (1996)

Index