Arrangements and Duality

M.Ghassabi

Department of Computer Science,
Yazd University

May 10, 2011

Outline

Outline

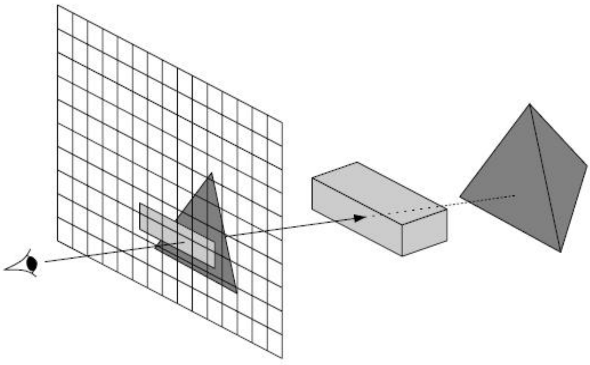
Outline

Outline

Outline

1. Supersampling in Ray Tracing
2. Computing the Discrepancy
3. Duality
4. Arrangements of Lines
5. Levels and Discrepancy

Ray Tracing:



Ray Tracing:

Rendering the scene:

Generating a 2-dimensional image of a 3-dimensional scene that amounts to:

- determining the visible object at each pixel on the screen,
- determining how bright the object is.

Ray tracing:

Determining the visible object at each pixel by shooting a ray from the view point through each pixel.

Note:

Ray tracing can also determine how bright the object is.

Ray Tracing:

Rendering the scene:

Generating a 2-dimensional image of a 3-dimensional scene that amounts to:

- determining the visible object at each pixel on the screen,
- determining how bright the object is.

Ray tracing:

Determining the visible object at each pixel by shooting a ray from the view point through each pixel.

Note:

Ray tracing can also determine how bright the object is.

Ray Tracing:

Rendering the scene:

Generating a 2-dimensional image of a 3-dimensional scene that amounts to:

- determining the visible object at each pixel on the screen,
- determining how bright the object is.

Ray tracing:

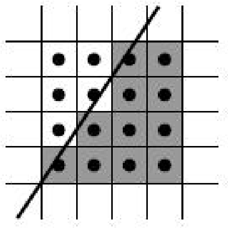
Determining the visible object at each pixel by shooting a ray from the view point through each pixel.

Note:

Ray tracing can also determine how bright the object is.

Supersampling:

- A pixel is not a point, but a small square area.
- Shooting a ray through each pixel center results in the well-known jaggies in the image.
- The solution is to shoot more than one ray per pixel.

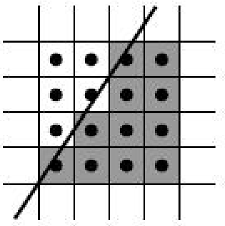


Supersampling:

instead of taking one sample point per pixel, we take many.

Supersampling:

- A pixel is not a point, but a small square area.
- Shooting a ray through each pixel center results in the well-known jaggies in the image.
- The solution is to shoot more than one ray per pixel.

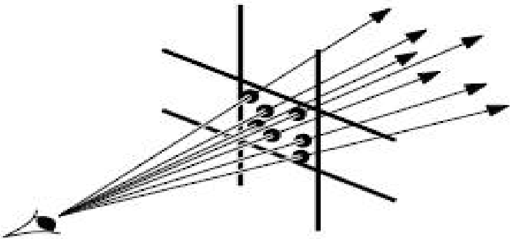


Supersampling:

instead of taking one sample point per pixel, we take many.

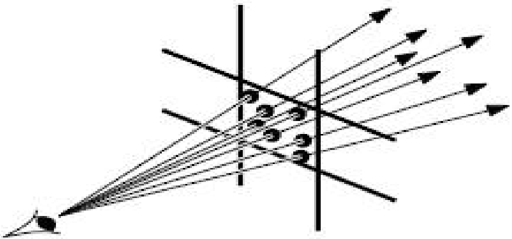
How should we distribute the rays over the pixel :

- Distributing rays regularly isn't such a good idea. Small per-pixel error, but regularity in error across rows and columns. (which triggers the human visual system.)
- It's better to choose the sample points in a somewhat random fashion.
- We want the sample points to be distributed in such a way that the number of hits is closed to the percentage of covered area.



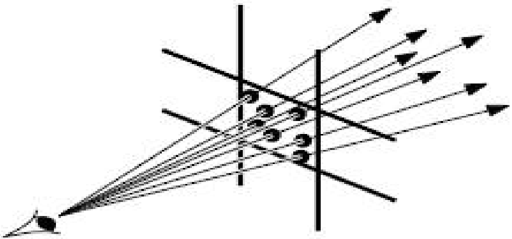
How should we distribute the rays over the pixel :

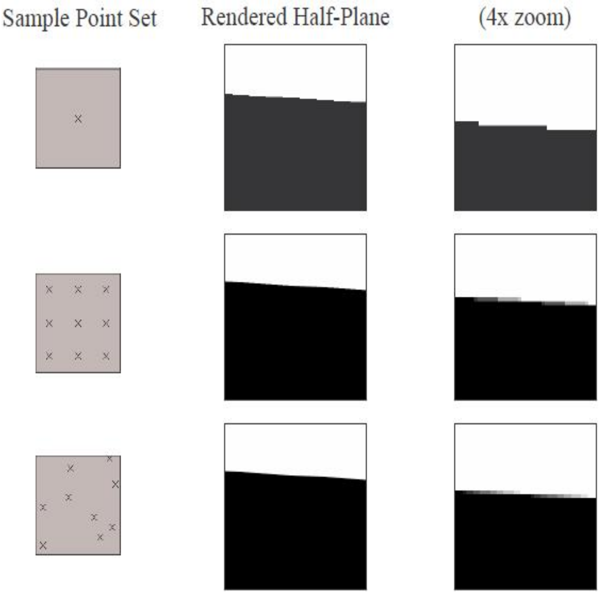
- Distributing rays regularly isn't such a good idea. Small per-pixel error, but regularity in error across rows and columns. (which triggers the human visual system.)
- It's better to choose the sample points in a somewhat random fashion.
- We want the sample points to be distributed in such a way that the number of hits is closed to the percentage of covered area.



How should we distribute the rays over the pixel :

- Distributing rays regularly isn't such a good idea. Small per-pixel error, but regularity in error across rows and columns. (which triggers the human visual system.)
- It's better to choose the sample points in a somewhat random fashion.
- We want the sample points to be distributed in such a way that the number of hits is closed to the percentage of covered area.




Sample Point Set
Rendered Half-Plane
(4x zoom)

Discrepancy:

Discrepancy of sample set with respect to object:

The difference between the percentage of hits for an object and the percentage of the pixel area where that object is visible.

Note:

we don't know in advance which objects will be visible in the pixel.

Discrepancy of the sample set:

The maximum discrepancy over all possible ways that an object can be visible inside the pixel.

Discrepancy:

Discrepancy of sample set with respect to object:

The difference between the percentage of hits for an object and the percentage of the pixel area where that object is visible.

Note:

we don't know in advance which objects will be visible in the pixel.

Discrepancy of the sample set:

The maximum discrepancy over all possible ways that an object can be visible inside the pixel.

Discrepancy:

Discrepancy of sample set with respect to object:

The difference between the percentage of hits for an object and the percentage of the pixel area where that object is visible.

Note:

we don't know in advance which objects will be visible in the pixel.

Discrepancy of the sample set:

The maximum discrepancy over all possible ways that an object can be visible inside the pixel.

How discrepancy can be useful?

Based on the discrepancy of given set of sample points we can decide if it is good enough: if the discrepancy is low enough we decide to keep it, and otherwise we generate a new random set.

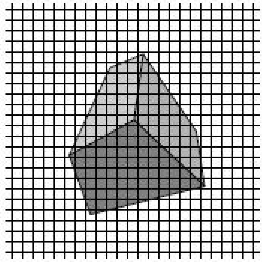
- For this we need an algorithm that computes the discrepancy of a given point set.

How discrepancy can be useful?

Based on the discrepancy of given set of sample points we can decide if it is good enough: if the discrepancy is low enough we decide to keep it, and otherwise we generate a new random set.

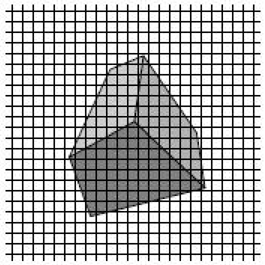
- For this we need an algorithm that computes the discrepancy of a given point set.

- Assume that curved objects are approximated using polygonal meshes.
- So the 2-dimensional objects that we must consider are the projections of the facet of polyhedra.



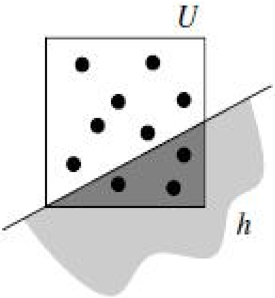
- Most likely, a single pixel intersects a single polygon side which is like intersecting a half-plane.
- Therefore we restrict our attention to half-plane discrepancy.

- Assume that curved objects are approximated using polygonal meshes.
- So the 2-dimensional objects that we must consider are the projections of the facet of polyhedra.



- Most likely, a single pixel intersects a single polygon side which is like intersecting a half-plane.
- Therefore we restrict our attention to half-plane discrepancy.

- $U = [0:1] \times [0:1]$: The unit square (pixel)
- $H =$ The (infinite) set of all possible half-planes (scene)
- $S =$ A set of n sample points in U
- Continuous measure : $\mu(h) =$ area of $h \cap U$
- Discrete measure : $\mu_S(h) = card(S \cap h)/card(S)$
- Discrepancy of h wrt S : $\Delta_S(h) = |\mu(h) - \mu_S(h)|$
- Half-plane discrepancy of S : $\Delta_H(S) = \sup_{h \in H} \Delta_S(h)$



- We first identify a finit set of condidate half-planes.
- The half-plane of maximum discrepancy must pass through at least one sample point.
- Let it pass through exactly one point.
- The maximum discrepancy must be at a local extremum of the continuous measure.
- There are an infinite number of h through each point p, but only $O(1)$ of them are local extrema.

- We first identify a finit set of condidate half-planes.
- The half-plane of maximum discrepancy must pass through at least one sample point.
- Let it pass through exactly one point.
- The maximum discrepancy must be at a local extremum of the continuous measure.
- There are an infinite number of h through each point p, but only $O(1)$ of them are local extrema.

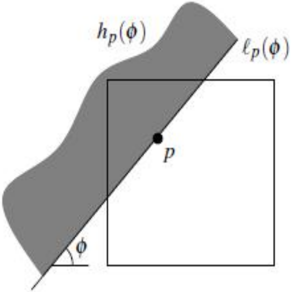
- We first identify a finit set of condidate half-planes.
- The half-plane of maximum discrepancy must pass through at least one sample point.
- Let it pass through exactly one point.
- The maximum discrepancy must be at a local extremum of the continuous measure.
- There are an infinite number of h through each point p, but only $O(1)$ of them are local extrema.

- We first identify a finit set of condidate half-planes.
- The half-plane of maximum discrepancy must pass through at least one sample point.
- Let it pass through exactly one point.
- The maximum discrepancy must be at a local extremum of the continuous measure.
- There are an infinite number of h through each point p, but only $O(1)$ of them are local extrema.

- We first identify a finit set of condidate half-planes.
- The half-plane of maximum discrepancy must pass through at least one sample point.
- Let it pass through exactly one point.
- The maximum discrepancy must be at a local extremum of the continuous measure.
- There are an infinite number of h through each point p, but only $O(1)$ of them are local extrema.

Let:

- $p := (p_x, p_y)$ be a point in S,
- $l_p(\phi)$ be the line through p that makes an angle ϕ with the positive x-axis for $0 \leq \phi < 2\pi$,
- $h_p(\phi)$ be the half-plane initially lying above $l_p(\phi)$.



- We are interested in the local extrema of the function $\phi \to \mu(h_p(\phi))$.

- There is a constant number of local extrema per point $p \in S$.
- Thus the total number of condidate half-planes with one point on their boundary is $O(n)$.
- Moreover, we can find the extrema and the corresponding half-planes in $O(1)$ time per point.

- There is a constant number of local extrema per point $p \in S$.
- Thus the total number of condidate half-planes with one point on their boundary is $O(n)$.
- Moreover, we can find the extrema and the corresponding half-planes in $O(1)$ time per point.

- There is a constant number of local extrema per point $p \in S$.
- Thus the total number of condidate half-planes with one point on their boundary is $O(n)$.
- Moreover, we can find the extrema and the corresponding half-planes in $O(1)$ time per point.

Lemma 8.1

Let S be a set of n points in the unit square U. A half-plane h that achieves the maximum discrepancy with respect to S is of one of the following types:

(i) h contains one point $p \in S$ on its boundary,

(ii) h contains two or more points of S on its boundary.

The number of type (i) condicates is $O(n)$, and they can be found in $O(n)$ time.

- The number of type (*ii*) condidates is quadratic.
- Because the number of type (*i*) condidates is linear, we treat them in a brute-force way: for each of the $O(n)$ half-planes we compute their continuous measure in constant time, and their discrete measure in $O(n)$ time. This way the muximum of the discrepancies of this half-planes can be computed in $O(n^2)$ time.
- For the type (*ii*) candidates we need some new techniques.

Theorem 8.2

The half-plane discrepancy of a set S of n points in the unit square can be computed in $O(n^2)$ time.

Duality:

- A point in the plane has two parameters: its x-coordinate and its y-coordinate.
- A (non-vertical) line in the plane also has two parameters: its slope and its intersection with the y-axis.
- Therefore we can map a set of points to a set of lines, and vice versa, in a one-to-one manner.

Duality transform:

One-to-one mapping of a set of points to a set of lines such that certain properties are preserved.

- The image of an object under a duality transform is called the dual of the object.

Duality:

- A point in the plane has two parameters: its x-coordinate and its y-coordinate.
- A (non-vertical) line in the plane also has two parameters: its slope and its intersection with the y-axis.
- Therefore we can map a set of points to a set of lines, and vice versa, in a one-to-one manner.

Duality transform:

One-to-one mapping of a set of points to a set of lines such that certain properties are preserved.

- The image of an object under a duality transform is called the dual of the object.

Duality:

- A point in the plane has two parameters: its x-coordinate and its y-coordinate.
- A (non-vertical) line in the plane also has two parameters: its slope and its intersection with the y-axis.
- Therefore we can map a set of points to a set of lines, and vice versa, in a one-to-one manner.

Duality transform:

One-to-one mapping of a set of points to a set of lines such that certain properties are preserved.

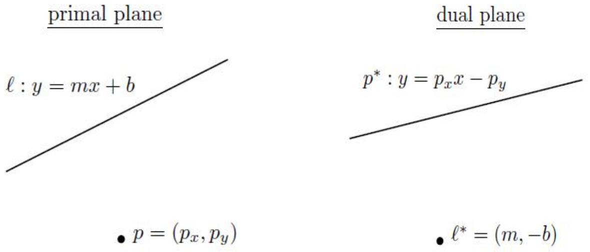
- The image of an object under a duality transform is called the dual of the object.

One possible and simple duality tranform:

- point $p : (p_x, p_y) \iff$ line $p^* : y = p_x x - p_y$
- line $l : y = mx + b \iff$ point $l^* : (m, -b)$

Note:

The duality transform is not defined for vertical lines.

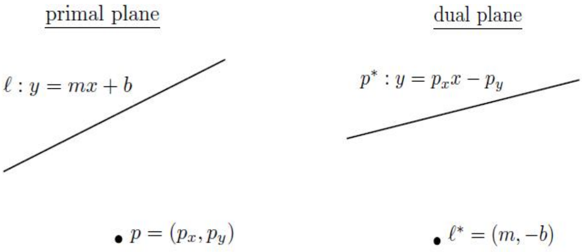


One possible and simple duality tranform:

- point $p : (p_x, p_y) \iff$ line $p^* : y = p_x x - p_y$
- line $l : y = mx + b \iff$ point $l^* : (m, -b)$

Note:

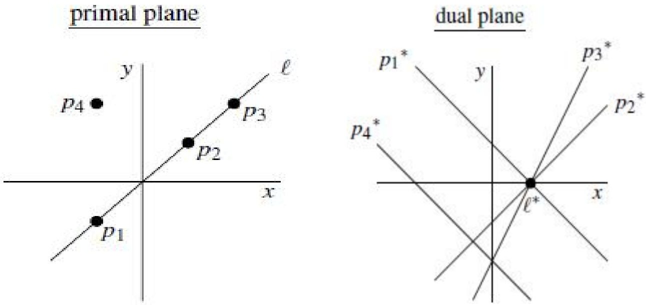
The duality transform is not defined for vertical lines.



Observation 8.3

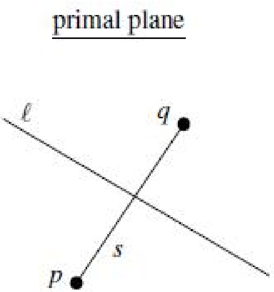
Let p be a point in the plane and let l be a non-vertical line in the plane. The duality transform $o \mapsto o^*$ has the following properties.

- It is incidence preserving: $p \in l$ if and only if $l^* \in p^*$.
- It is order preserving: p lies above l if and only if l^* lies above p^*.



Duality can be applied to other objects, e.g. segments:

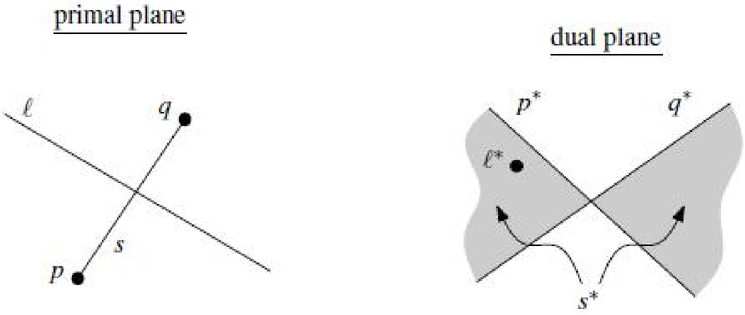
- Let $s := \bar{pq}$ be a line segment



- Dual of a segment is a double wedge.

Duality can be applied to other objects, e.g. segments:

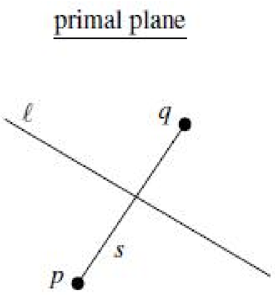
- Let $s := \bar{pq}$ be a line segment

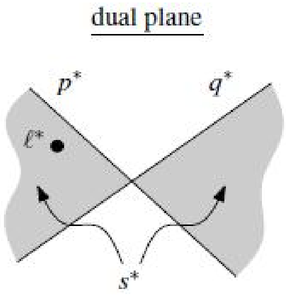


- Dual of a segment is a double wedge.

Duality can be applied to other objects, e.g. segments:

- Let $s := \bar{pq}$ be a line segment

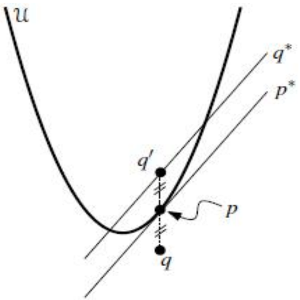




- Dual of a segment is a double wedge.

Duality can be applied to other objects, e.g. parabola:

- parabola $\mathcal{U} : y = x^2/2$
- point $p = (p_x, p_y)$ on $\mathcal{U}$
- derivative of $\mathcal{U}$ at p is p_x, i.e., p^* has same slope as tangent line
- tangent line intersects y-axis at $(0, -p_x^2/2)$
- $\Rightarrow p^*$ is tangent line at p
- if q lies directly above or below p, then q^* is the line parallel to p^*



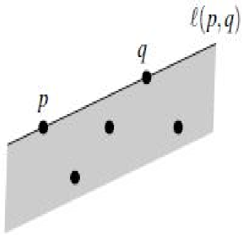
How duality can be useful?

- If you can solve a problem in the dual plane, you could solved it in the primal plane as well by mimicking the solution to the dual problem in the primal problem.
- Looking at things on the dual plane provides new perspectives.

Back to the discrepancy problem:

To determine our discrete measure, we need to:

- Determine how many sample points lie below a given line(in the primal plane).



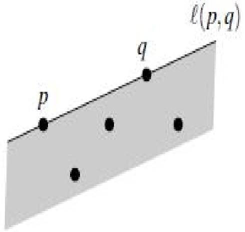
Dualizes to:

- Given a point in the dual plane we want to determine how many sample lines lie above it.

Back to the discrepancy problem:

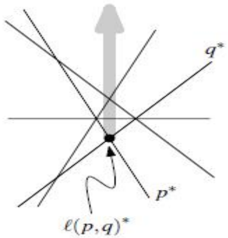
To determine our discrete measure, we need to:

- Determine how many sample points lie below a given line(in the primal plane).



Dualizes to:

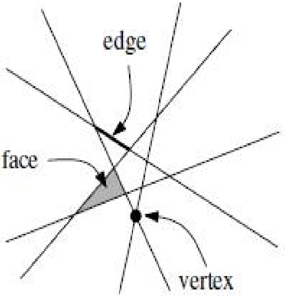
- Given a point in the dual plane we want to determine how many sample lines lie above it.



Arrangements:

Arrangement $A(L)$:

Let L be a set of n lines in the plane. L induces a subdivision of the plane that consists of vertices,edges, and faces.This is called the arrangement induced by L, denoted $A(L)$.



Simple arrangment:

An arrangement is called simple if no three lines pass through the same point and no two lines are parallel.

Complexity:

The complexity of an arrangement is defined as the total number of vertices, edges, and faces of the arrangement.

Simple arrangment:

An arrangement is called simple if no three lines pass through the same point and no two lines are parallel.

Complexity:

The complexity of an arrangement is defined as the total number of vertices, edges, and faces of the arrangement.

Theorem 8.4

Let L be a set of n lines in the plane, and let $A(L)$ be the arrangement induced by L.

(i) The number of vertices of $A(L)$ is at most $n(n-1)/2$.

(ii) The number of edges of $A(L)$ is at most n^2.

(iii) The number of faces of $A(L)$ is at most $n^2/2 + n/2 + 1$.

Equality holds in these three statements if and only if $A(L)$ is simple.

- Total complexity of an arrangement is $O(n^2)$.

Theorem 8.4

Let L be a set of n lines in the plane, and let $A(L)$ be the arrangement induced by L.

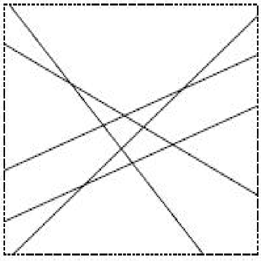
- (i) The number of vertices of $A(L)$ is at most $n(n-1)/2$.
- (ii) The number of edges of $A(L)$ is at most n^2.
- (iii) The number of faces of $A(L)$ is at most $n^2/2 + n/2 + 1$.

Equality holds in these three statements if and only if $A(L)$ is simple.

- Total complexity of an arrangement is $O(n^2)$.

Constructing Arrangements:

- We place a bounding box $B(L)$ that contains all the vertices of $A(L)$ in its interior.



- The subdivision defined by the bounding box plus the part of the arrangement inside it has bounded edges only and can be stored in a doubly-connected edge list.

Constructing Arrangements:

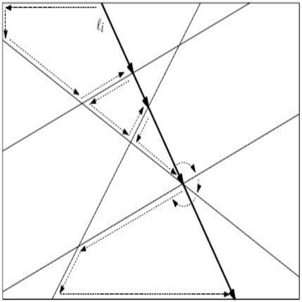
Goal:

Compute A(L) in bounding box in DCEL representation

- A plane sweep algorithm would run in $O(n^2 \log n)$ time.
- faster: Incremental algorithm ($O(n^2)$)

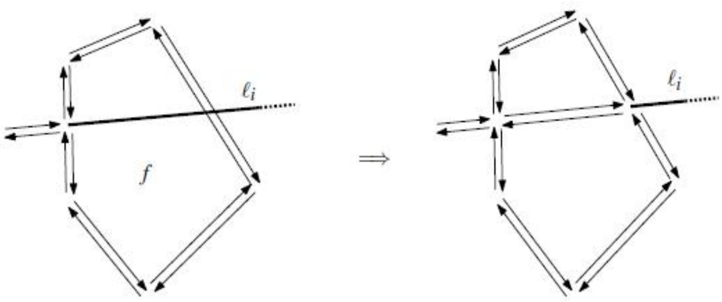
Incremental Algorithm:

- Compute a bounding box $B(L)$ that contains all vertices of $A(L)$ in its iterior and initialize the DCEL.
- Incrementally add each line l_i to A_{i-1} and update DCEL.
  - Find the edge e on $B(L)$ that contains the leftmost intersection point of l_i and A_i
  - Split face bounded by e
  - Move on to next intersected face



Incremental Algorithm:

- Splitting a face f intersected by l_i:



Incremental Algorithm:

- Splitting a face f intersected by l_i:
  - Assume that the face intersected by l_i to the left of f has already been split.
  - Find the edge e' where l_i leaves f and its twin.
  - Create two new records for new faces f' and f'' created by l_i.
  - Create a new vertex record for vertex v' where l_i intersects e' ($l_i \cap e'$).
  - Create two new records for half-edges created by v'.
  - Create half-edge record for the edge $l_i \cap f$.
  - Delete records for e' and f.
- Move to face bounded by twin(e').

Incremental Algorithm:

Algorithm CONSTRUCTARRANGEMENT(L)
Input. A set L of n lines in the plane.
Output. The doubly-connected edge list for the subdivision induced by $\mathcal{B}(L)$ and the part of $\mathcal{A}(L)$ inside $\mathcal{B}(L)$, where $\mathcal{B}(L)$ is a bounding box containing all vertices of $\mathcal{A}(L)$ in its interior.

1. Compute a bounding box $\mathcal{B}(L)$ that contains all vertices of $\mathcal{A}(L)$ in its interior.
2. Construct the doubly-connected edge list for the subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** Find the edge e on $\mathcal{B}(L)$ that contains the leftmost intersection point of ℓ_i and $\mathcal{A}_i$.
5. $f \leftarrow$ the bounded face incident to e
6. **while** f is not the unbounded face, that is, the face outside $\mathcal{B}(L)$
7. **do** Split f, and set f to be the next intersected face.

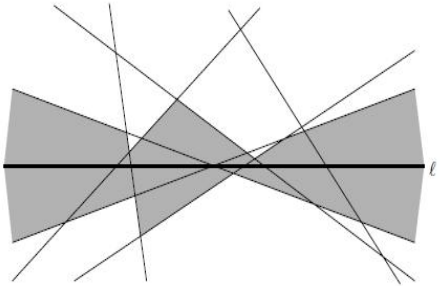
Running time analysis:

- Step 1, computing $B(L)$, can be done in $O(n^2)$ time.
- Step 2, constructing DCEL for $B(L)$, takes only constant time.
- Step 4, Finding the first face split by l_i takes $O(n)$ time.
- We now bound the time it takes to split the faces intersected by l_i (step 7).
- The edges we encounter are on the boundary of faces whose closure is intersected by l_i. This leads us to the concept of zones.

Zones:

Zone of a line l in an arrangement:

The zone of a line l in an arrangement $A(L)$ is the set of faces of $A(L)$ whose closure intersects l.



Zones:

Complexity:

The complexity of a zone is defined as the total complexity of all the faces it consists of, i.e. the sum of the number of edges and vertices of these faces.

- The time we need to insert line l_i is linear in the complexity of the zone of l_i in $A(l_1, \ldots, l_i)$.
- The Zone Theorem tells us that his quantity is linear.

Zones:

Complexity:

The complexity of a zone is defined as the total complexity of all the faces it consists of, i.e. the sum of the number of edges and vertices of these faces.

- The time we need to insert line l_i is linear in the complexity of the zone of l_i in $A(l_1, \ldots, l_i)$.
- The Zone Theorem tells us that his quantity is linear.

Zone Theorem:

Theorem 8.5 (Zone Theorem)

The complexity of the zone of a line in an arrangement of m lines in the plane is $O(m)$.

Proof of Zone Theorem:

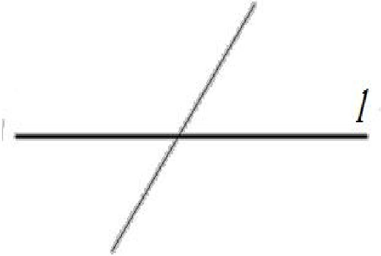
- Given an arrangement of m lines, $A(L)$, and a line l.
- Without loss of generality we assume that l coincides with the x-axis.
- An edge is a left bounding edge for the face lying to the right of it and a right bounding edge for the face lying to the left of it.
- Claim: the number of left bounding edges of the faces in the zone of l is at most $5m$.(Same for number of right bounding edges)

Proof of Zone Theorem:

(i) Assume first that no line of L is horizontal.

Claim: the number of left bounding edges of the faces in the zone of l is at most $5m$.(Same for number of right bounding edges)

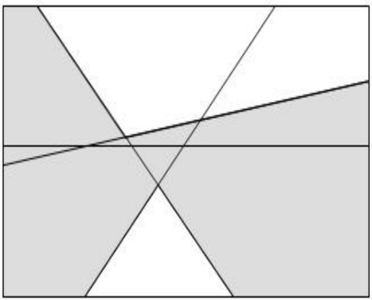
- By induction on m.
- For $m = 1$: Trivial.
  (1 left bounding edge ≤ 5)
- For $m > 1$:



Proof of Zone Theorem:

(1) Let l_1 be the rightmost line intersecting l (assume it's unique).

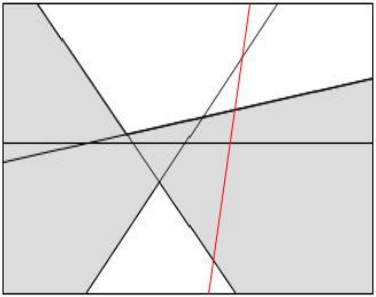
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - One new left bounding edge on l_1.
  - Two old left bounding edges split by l_1.
- Hence, the total number of left bounding edged in this case is at most $5(m-1)+3 < 5m$.



Proof of Zone Theorem:

(1) Let l_1 be the rightmost line intersecting l (assume it's unique).

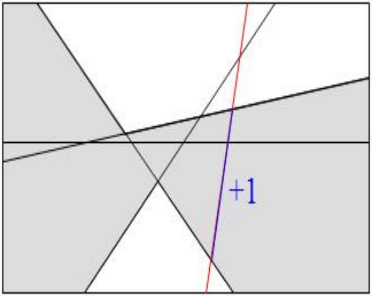
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - One new left bounding edge on l_1.
  - Two old left bounding edges split by l_1.
- Hence, the total number of left bounding edged in this case is at most $5(m-1)+3 < 5m$.



Proof of Zone Theorem:

(1) Let l_1 be the rightmost line intersecting l (assume it's unique).

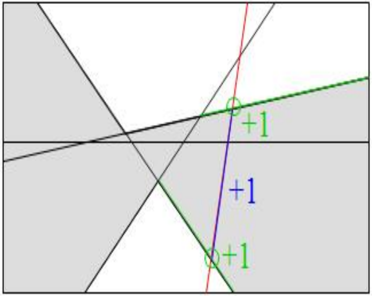
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - One new left bounding edge on l_1.
  - Two old left bounding edges split by l_1.
- Hence, the total number of left bounding edged in this case is at most $5(m-1)+3 < 5m$.



Proof of Zone Theorem:

(1) Let l_1 be the rightmost line intersecting l (assume it's unique).

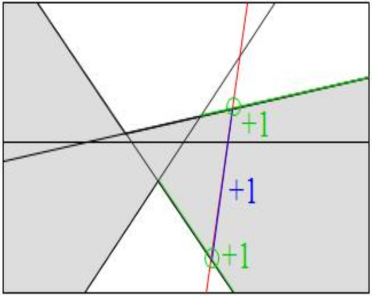
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - One new left bounding edge on l_1.
  - Two old left bounding edges split by l_1.
- Hence, the total number of left bounding edged in this case is at most $5(m-1)+3<5m$.



Proof of Zone Theorem:

(1) Let l_1 be the rightmost line intersecting l (assume it's unique).

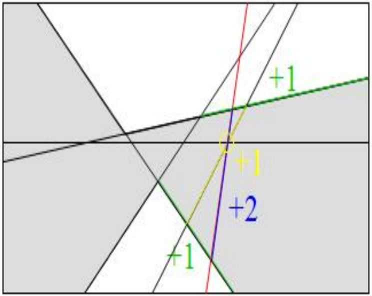
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - One new left bounding edge on l_1.
  - Two old left bounding edges split by l_1.
- Hence, the total number of left bounding edged in this case is at most $5(m-1)+3 < 5m$.



Proof of Zone Theorem:

(2) If exactly two lines intersect l in the rightmost intersection point:

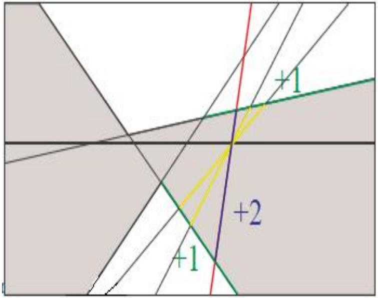
- Denote these lines by l_1 ,l_2.
- The zone of l in $A(L \setminus l_1)$ has at most 5(m-1) left bounding edges.
- l_1 has two left bounding edges
- l_2 is split into two left bounding edges
- l_1 splits two other left bounding edges
- Hence, the new zone complexity is at most $5(m-1) + 5 = 5m$.



Proof of Zone Theorem:

(3) If several lines (> 2) intersect l in the rightmost intersection point:

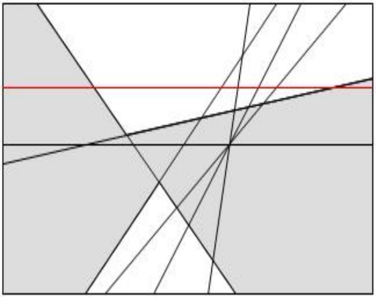
- Pick l_1 randomly out of these lines.
- The zone of l in $A(L \setminus l_1)$ has at most $5(m-1)$ left bounding edges.
- When adding l_1, the number of such edges increases:
  - Two new edges on l_1.
  - Two old edges split by l_1.
- Hence, the new zone complexity is at most $5(m-1) + 4 < 5m$.



Proof of Zone Theorem:

(ii) And what if there are horizontal lines?

- A horizontal line that does'nt coincide with l, introduces less complexity into A(L) than a non-horizontal line.
- If L contains a line l_i that coincide with l, the addition of l_i to $A(L \setminus l_i)$ increases the number of left bounding edges by at most $4m - 2$
- This concludes the proof of the Zone Theorem.



Theorem 8.6

A doubly-connected edge list for the arrangement induced by a set of n lines in the plane can be constructed in $O(n^2)$ time.

Run time analysis:

1. $O(n^2)$
2. constant
3. $\sum_{i=1}^{n} O(i) = O(n^2)$

in total $O(n^2)$

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Back to Discrepancy (Again):

- For every line between two sample points, we want to determine how many sample points lie below that line.
- For every vertex in the dual plane, we want to determine how many sample lines lie above it.
- We build the arrangement $A(S^*)$ and use that to determine, for each vertex, how many lines lie above it.

Levels and Discrepancy:

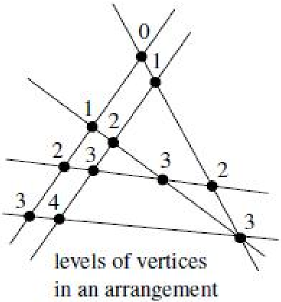
level of a point:

The level of a point in an arrangement of lines is the number of lines strictly above it.

Levels and Discrepancy:

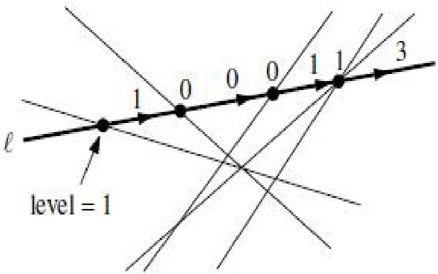
level of a point:

The level of a point in an arrangement of lines is the number of lines strictly above it.



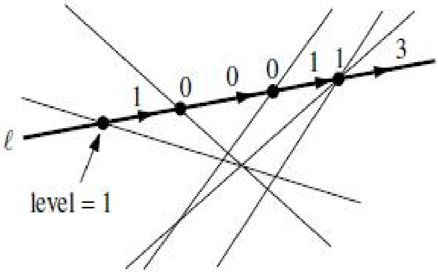
Computing the Levels:

- For each line l in S^*:
  - Compute the level of the leftmost vertex. $O(n)$
  - Walk along l from left to right to visit the other vertices on l, using the DCEL. The level only changes at a vertex, and the change can be computed by inspecting the edges incident to the vertex that is encountered. $O(1)$
- The levels of all vertices of $A(S^*)$ can be computed in $O(n^2)$ time.



Computing the Levels:

- For each line l in S^*:
  - Compute the level of the leftmost vertex. $O(n)$
  - Walk along l from left to right to visit the other vertices on l, using the DCEL. The level only changes at a vertex, and the change can be computed by inspecting the edges incident to the vertex that is encountered. $O(1)$
- The levels of all vertices of $A(S^*)$ can be computed in $O(n^2)$ time.



Summary:

- Problem regarding points S in ray-tracing
- Dualize to a problem of lines L.
- Compute arrangement of lines $A(L)$.
- Compute level of each vertex in $A(L)$.
- Use this to compute discrete measures in primal space.
- We can determine how good a distribution of sample points is in $O(n^2)$ time.

END.