CS633 Lecture 11
Arrangements and Duality

Jyh-Ming Lien

Department of Computer Science
George Mason University

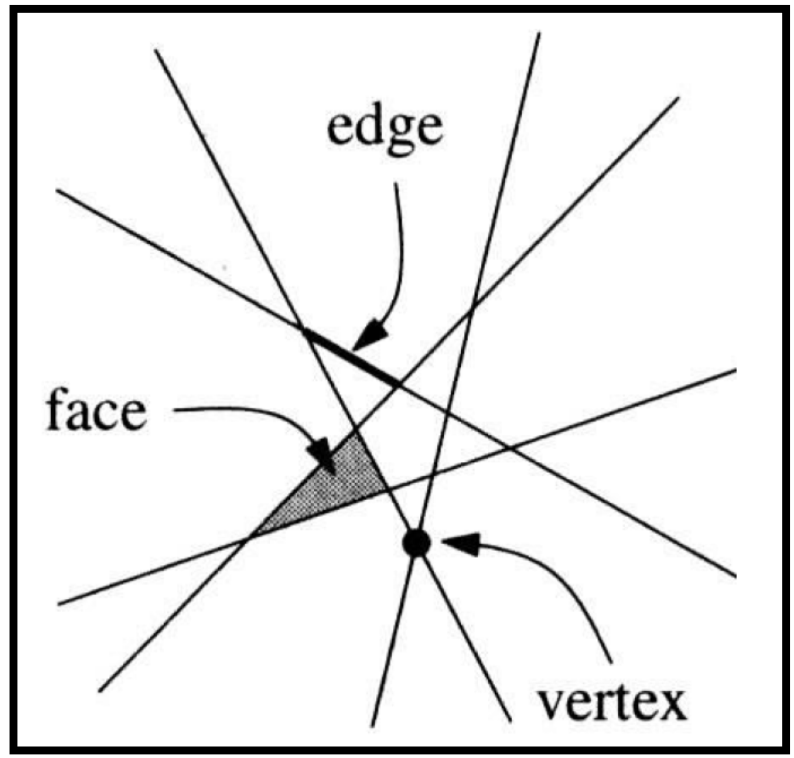
Based on the lecture notes of Sanjay Sthapit (UNC) and Darius Jazayeri (MIT)

Arrangements of Lines

- *L* is a set of *n* lines in a plane

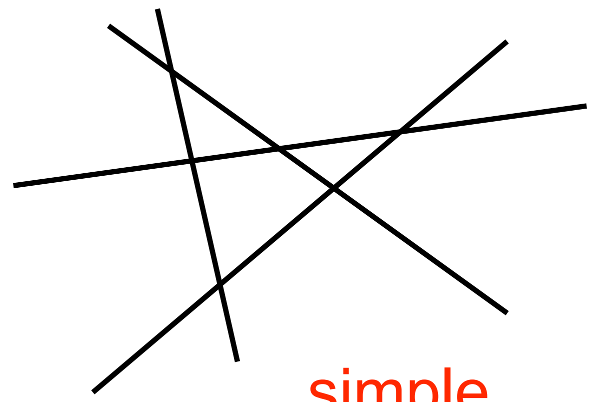
- An arrangement *A(L)* of *L* is the subdivision of a plane by *L*

- The complexity of *A(L)* is the total number of vertices, edges, and faces of the subdivision



Simple Arrangements

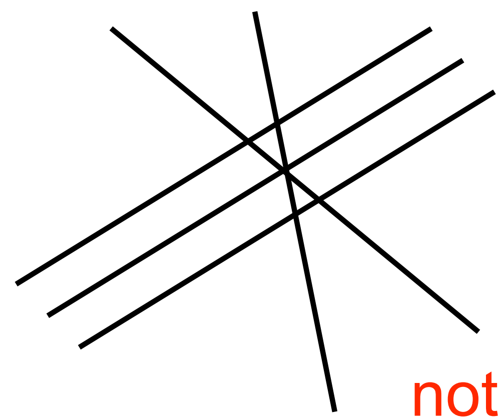
- An arrangement is simple if it does not contain
  - parallel lines
  - 3 or more lines with a common intersection point



simple

Question:

Complexity of simple arrangement
vs.
Complexity of non-simple arrangement



not simple

Complexity of Arrangements

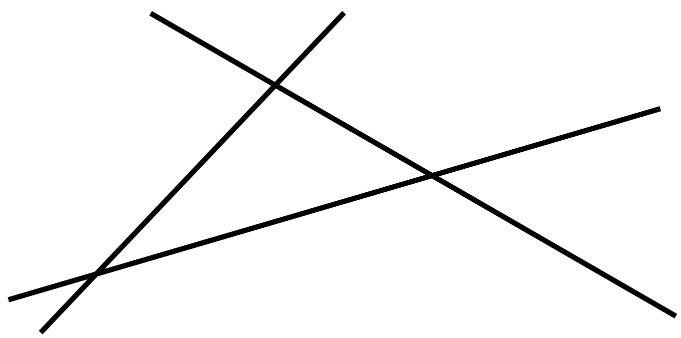
- For a set L of n lines on a plane and their arrangement $A(L)$:
  - number of vertices in $A(L) = n(n - 1)/2$
  - number of edges in $A(L) = n^2$
  - number of faces in $A(L) = n^2/2 + n/2 + 1$

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

Complexity of Arrangements

- For a set L of n lines on a plane and their arrangement $A(L)$:
  - number of vertices in $A(L) = n(n - 1)/2$
- *Proof :*

$(n-1)+(n-2)+(n-3)+\ldots+2+1= n(n-1)/2$



2+1+0=3

Complexity of Arrangements

- For a set L of n lines on a plane and their arrangement $A(L)$:
  - number of edges in $A(L) = n^2$
- *Proof : (by induction)*

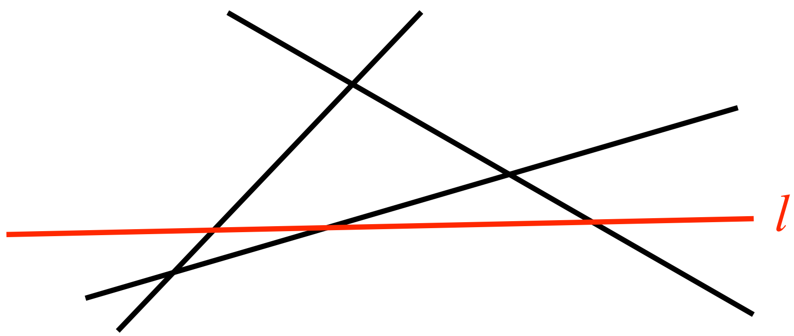
Strategy: Assume it's correct for $(n-1)$ and show that it is still correct when one more line l is added

$(n-1)^2$

$+(n-1)$ // From old $n-1$ lines

$+n$ // From l

$= n^2$

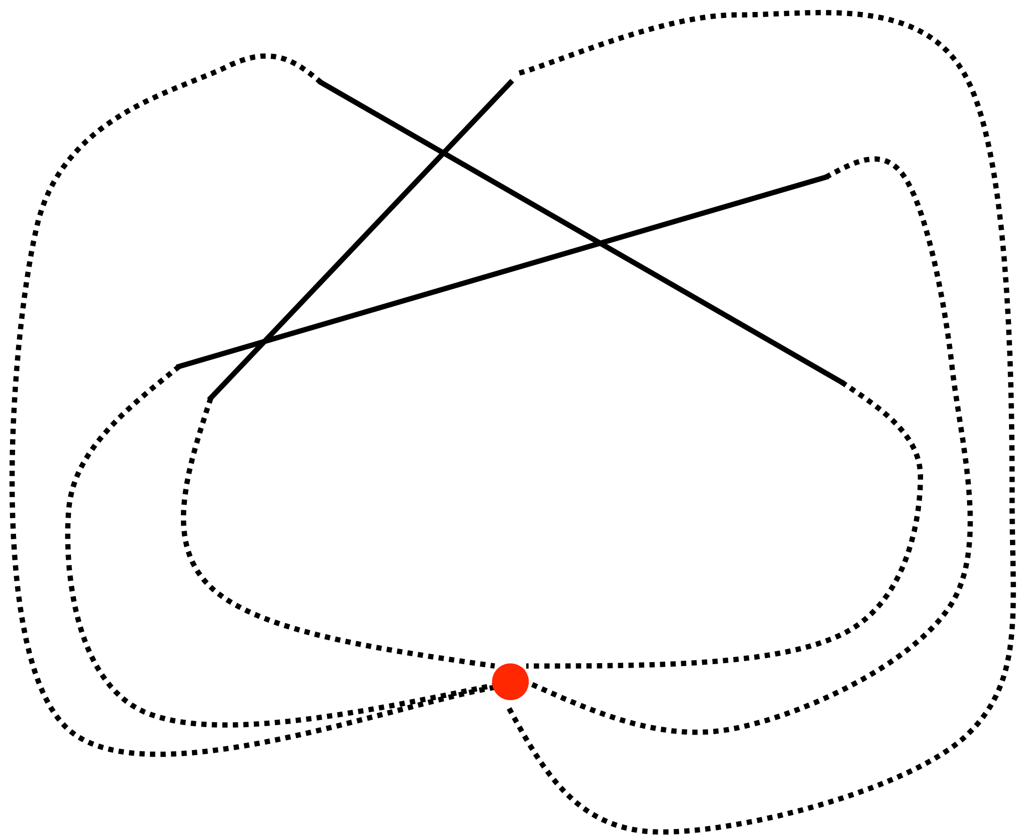


Complexity of Arrangements

- For a set L of n lines on a plane and their arrangement $A(L)$:
  - number of faces in $A(L)$ = $n^2/2 + n/2 + 1$

Proof

1. Add an extra vertex at infinity
2. join all line open-ended edges to this vertex



By Euler's formula

$V - E + F = 2$

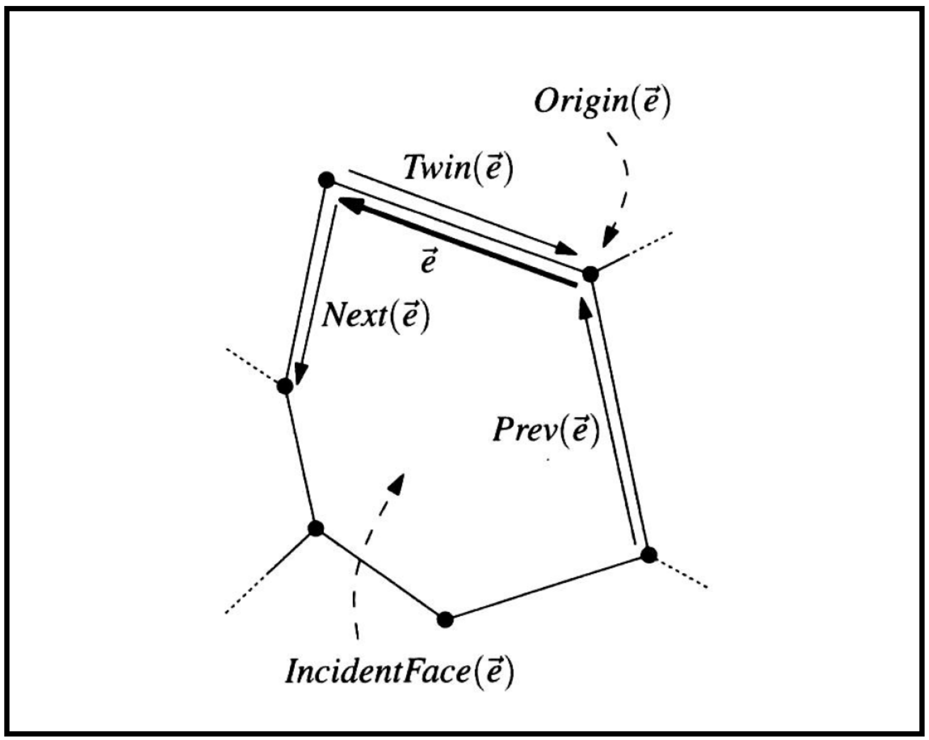
$\Rightarrow F = 2 - (V + 1) + E$

$\quad = 2 - (n(n-1)/2 + 1) + n^2$

$\quad = n^2/2 + n/2 + 1$

Data Structure for Arrangement

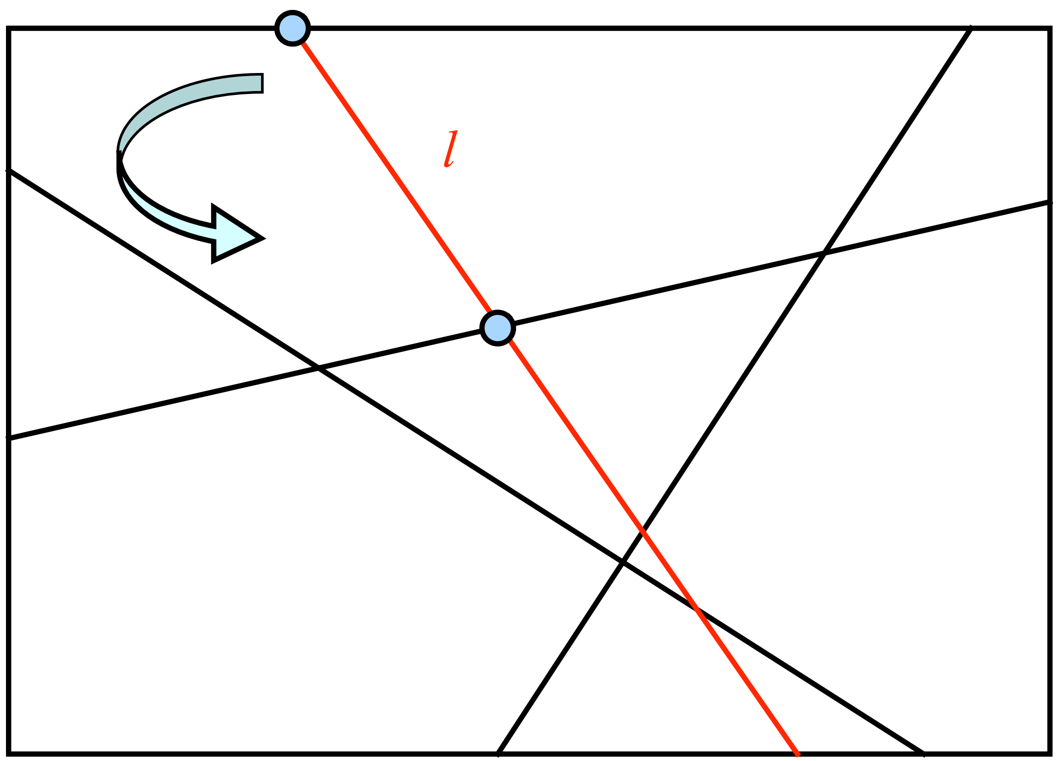
- Doubly connect edge list (Again)



Similar to Voronoi/Delaunay diagram, we can add a bounding box

Build An Arrangement

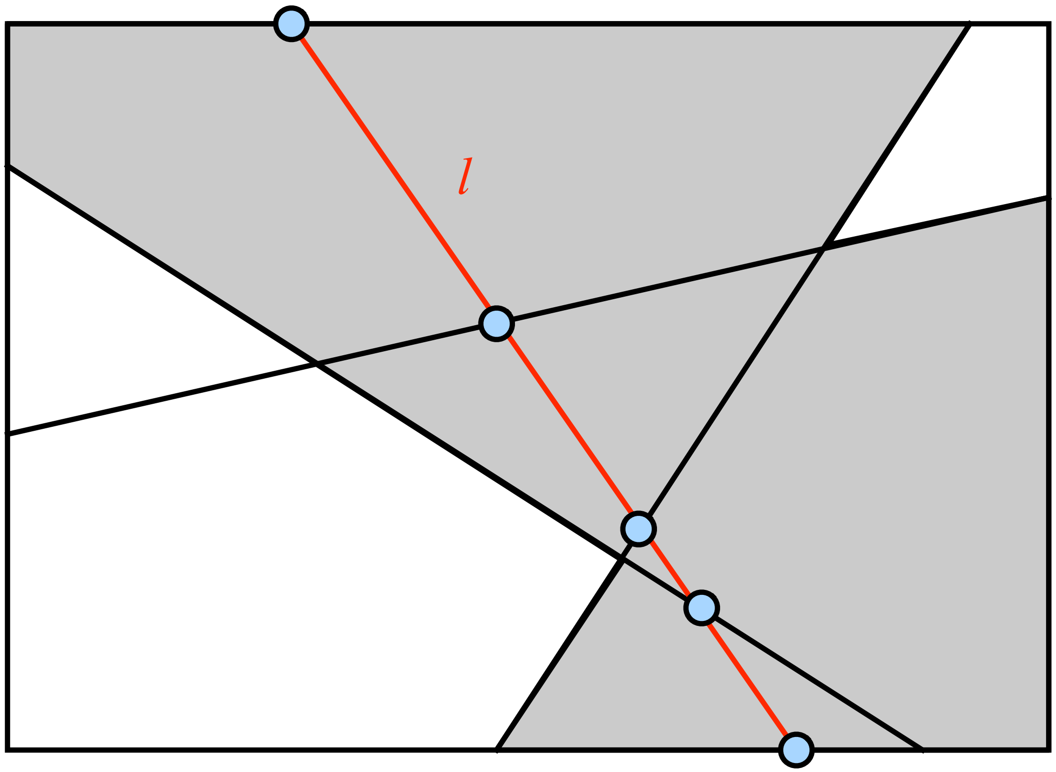
- Incremental algorithm
  - Add one line at a time



Walk around each face, to find the next intersection (new vertex)

Time Complexity of Building An Arrangement

- Depends on the complexity of all faces intersected by each new line



This is called the "Zone" of line l

Time Complexity of Building An Arrangement

- Zone complexity
  - Total number of vertices/edges/faces in the zone
- Zone theorem
  - The complexity of a zone of a line in an arrangement of n lines on the plane is $O(n)$
    - In particular, $z_n \le 6n$

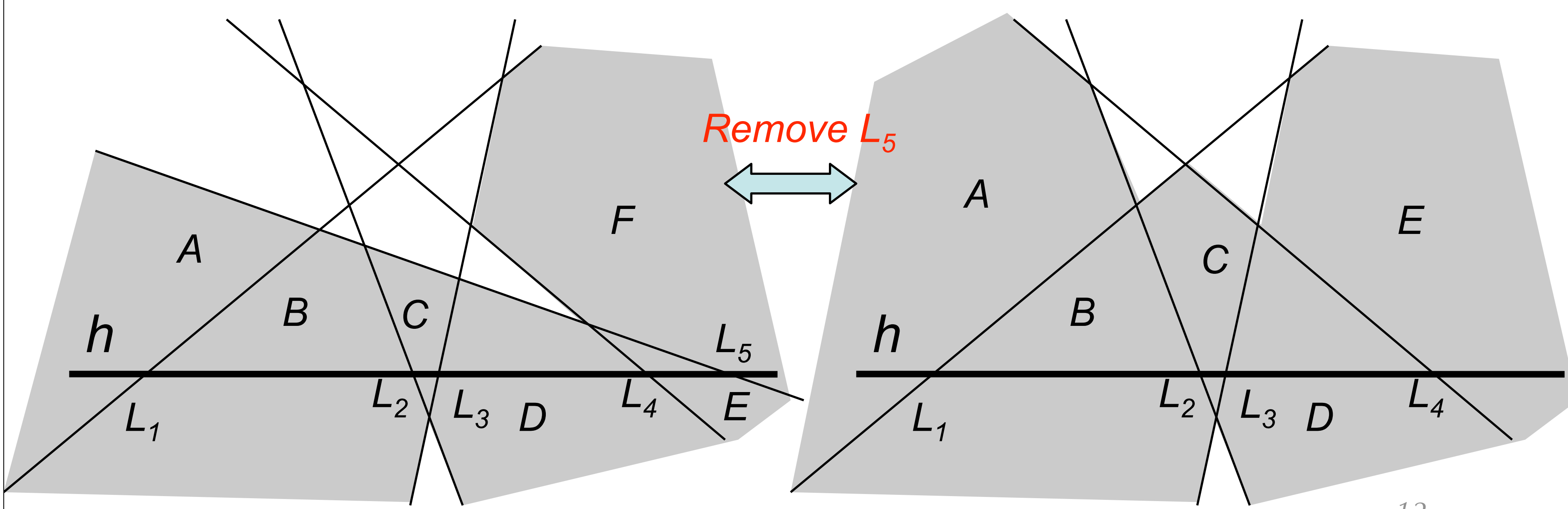
⇒We can build an arrangement in time:

$$1+2+\ldots+n=O(n^2)$$

Proof of Zone Theorem

$z_n \leq 6n$

- Given an arrangement of *n* lines
- Assume *l* is a horizontal line
- Prove by induction: take a way one line
  - So we have $z_n \leq 6n\text{-}6$
- Put is back the line we remove, show that $z_n \leq 6n$

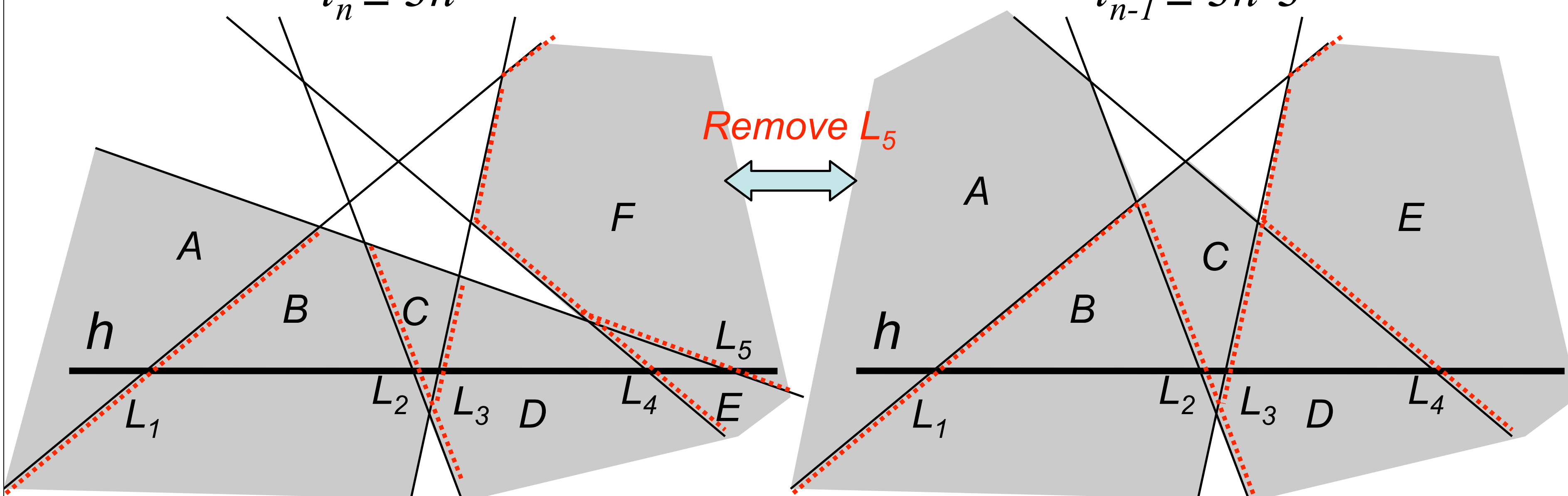


Proof of Zone Theorem

$z_n \leq 6n$

- Instead of counting total complexity, we count # of "left edges" of each face
- We prove $l_n \leq 3n$

$l_n \leq 3n$ $\quad\quad$ $l_{n-1} \leq 3n-3$



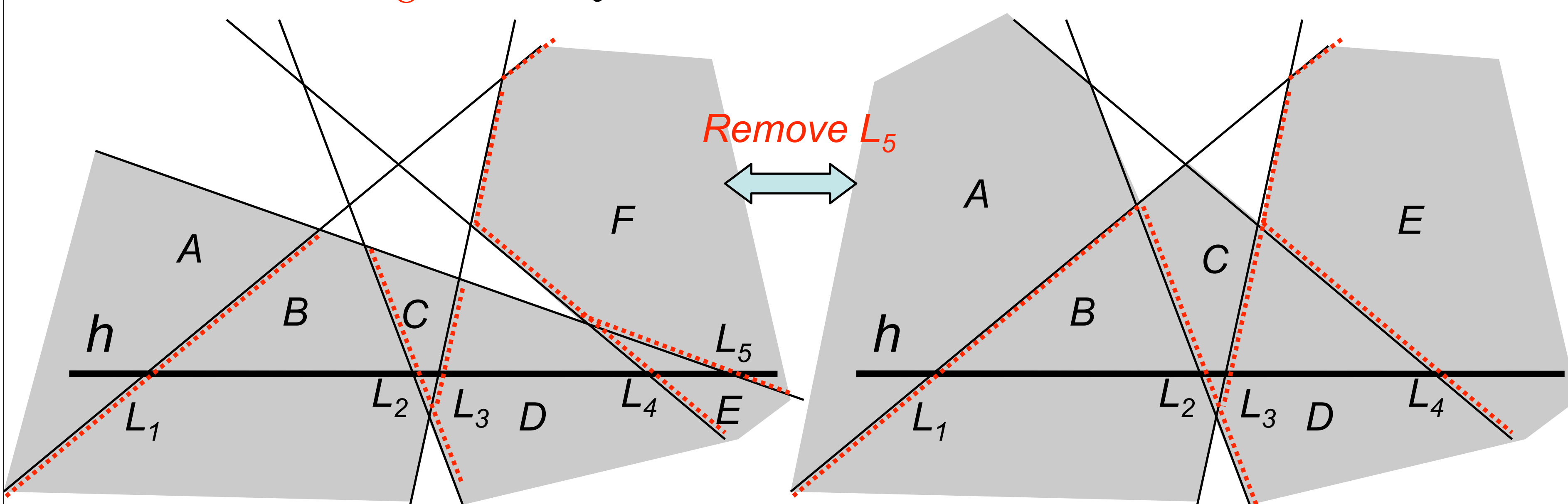
We prove that adding one line will add at most 3 left edges!

Proof of Zone Theorem

$l_n \leq 3n$

- Adding one line will add at most 3 left edges
  - One from the new line added
  - Two from the old left edge

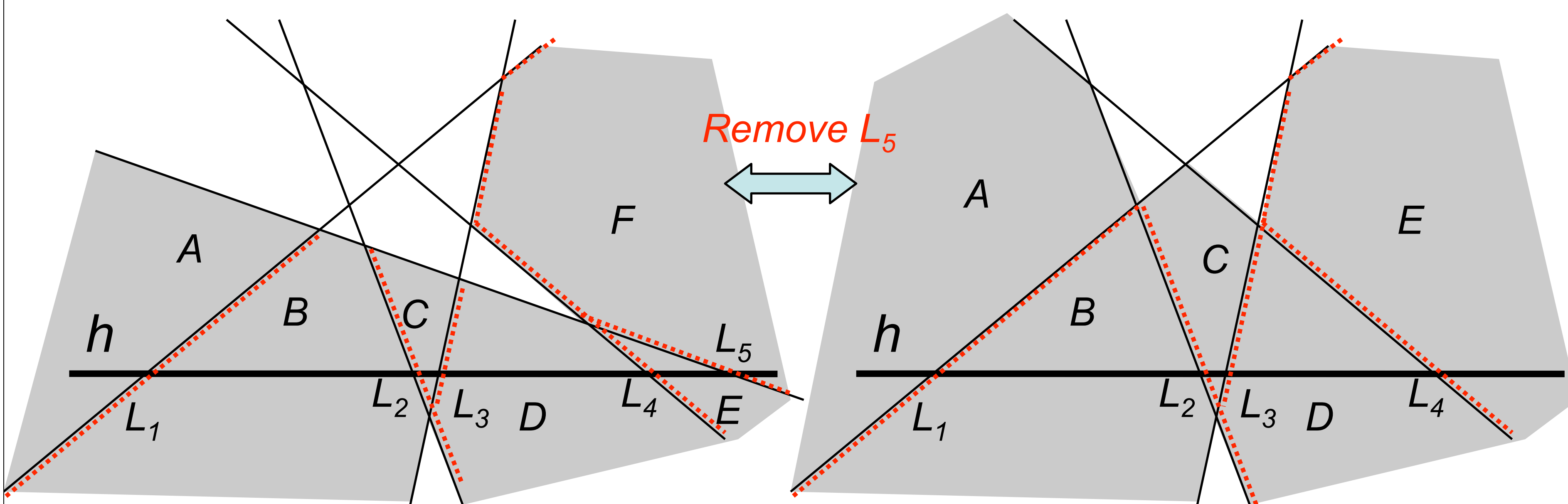
- The line r we pick to remove/add is the line whose intersection with h is rightmost, L_5 in our case



Proof of Zone Theorem

$l_n \leq 3n$

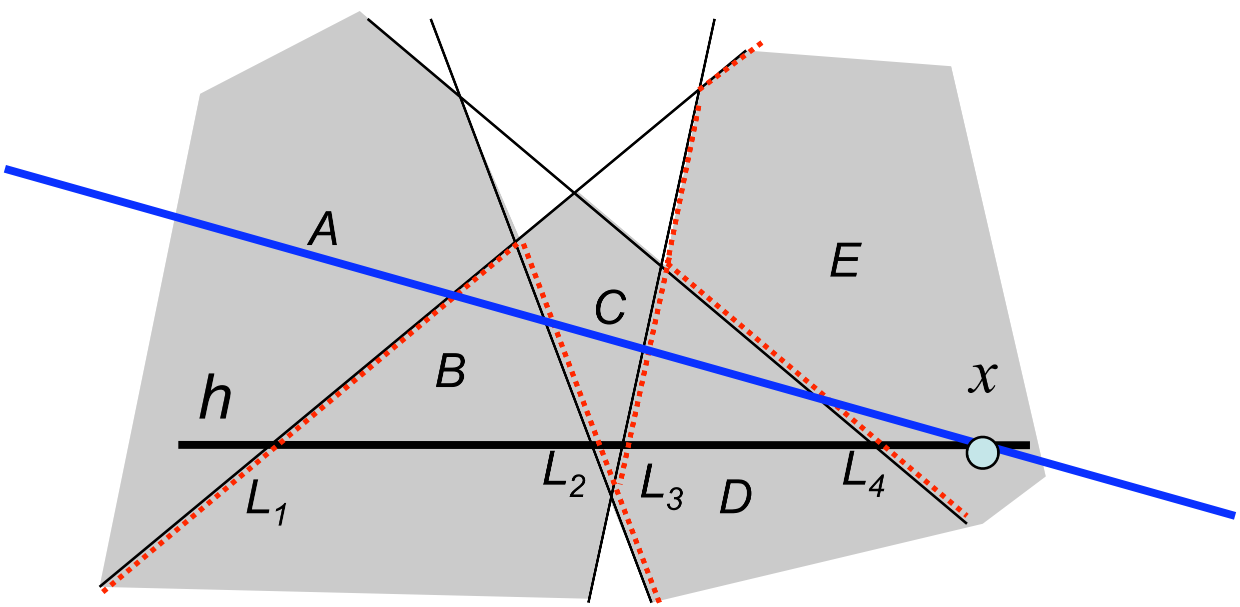
- At most one new left edge is added
  - The line *r* only contribute one left edge
  - The line *r* that contribute multiple left edges, must have another line intersecting *h* on its right
    - Example, L_2 and L_3



Proof of Zone Theorem

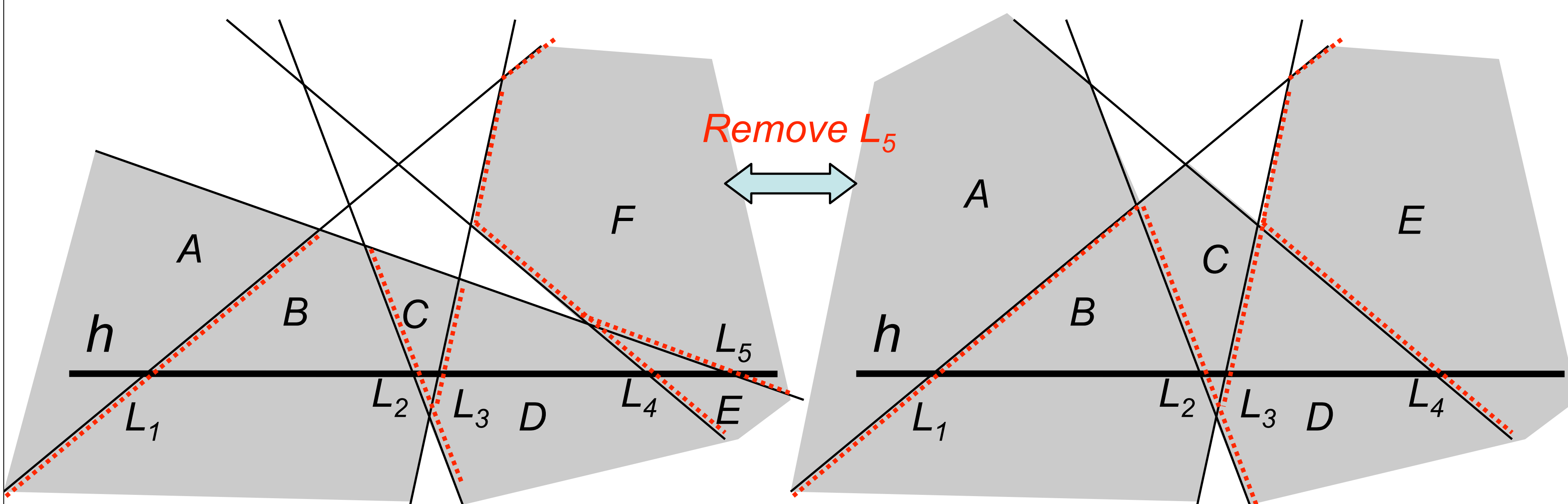
$l_n \leq 3n$

- At most two old left edges are split
  - Adding a line *r* will divide each cell in the zone to at most two cells
  - Let intersection *r* and *h* be *x*
  - Cells left to *x*, will be "clipped" (one sub cell will not be in the zone)
  - Cells contain or right to *x*, will be "split" (both sub cells will be in the zone)



Proof of Zone Theorem
$l_n \leq 3n$

- At most two old left edges are split
  - Only the rightmost face will be split by *r*
    - Face must be convex so at most two edges will be split
  - The line *r* only "clips" other faces
    - Clipping does not increase the number of left edges



Summary

- Complexity of an arrangement of n lines in a plane $O(n^2)$

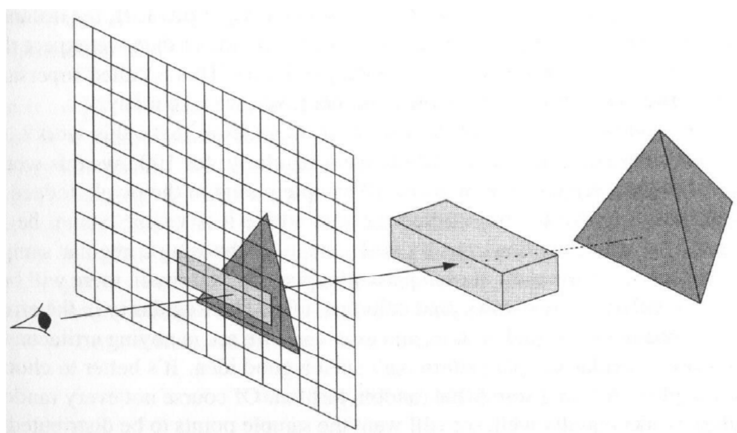
- Building an arrangement of n lines in a plane takes $O(n^2)$ time
  - Zone theorem
    - The zone of a line is a set of faces intersecting the line
    - Complexity of a zone is ***linear*** to the number of lines

Applications of Arrangements

- Ray tracing rendering
- Compute Voronoi diagram
  – K closest computation
- Visibility graph
- Hidden surface removal
- Ham (cheese) sandwich cut
- Motion planning
- …

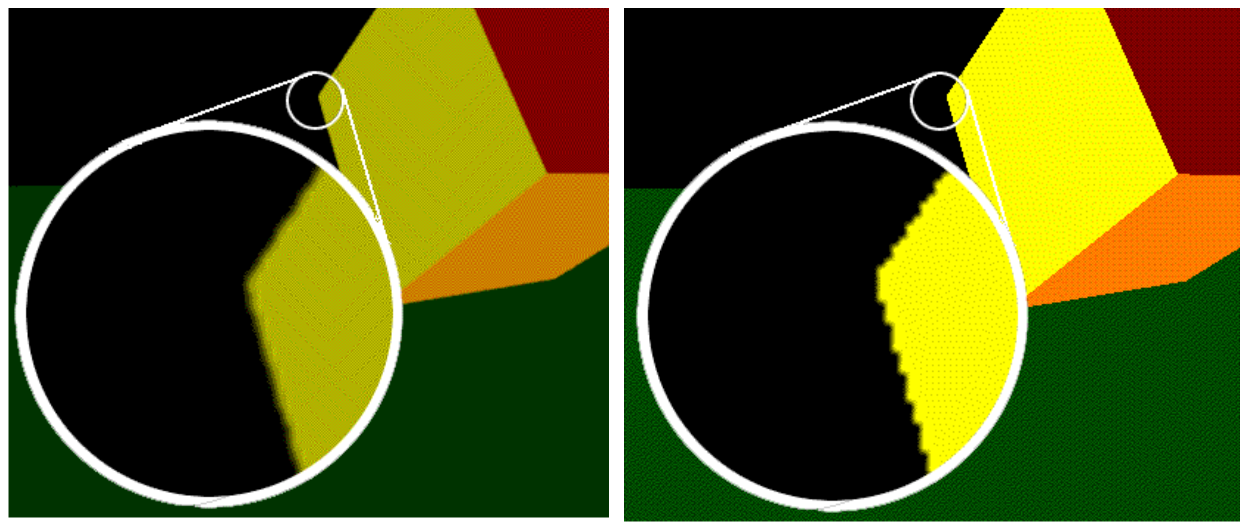
Ray-Tracing Rendering

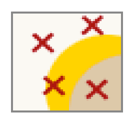
- Shooting rays from each pixel
  - Decide which object hits the rays
  - Determine the color of the pixel



Ray-Tracing Rendering

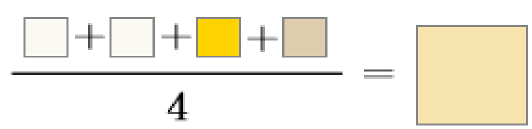
- One of the oldest problems in rendering
  - anti-aliasing





Pixel with sample postions

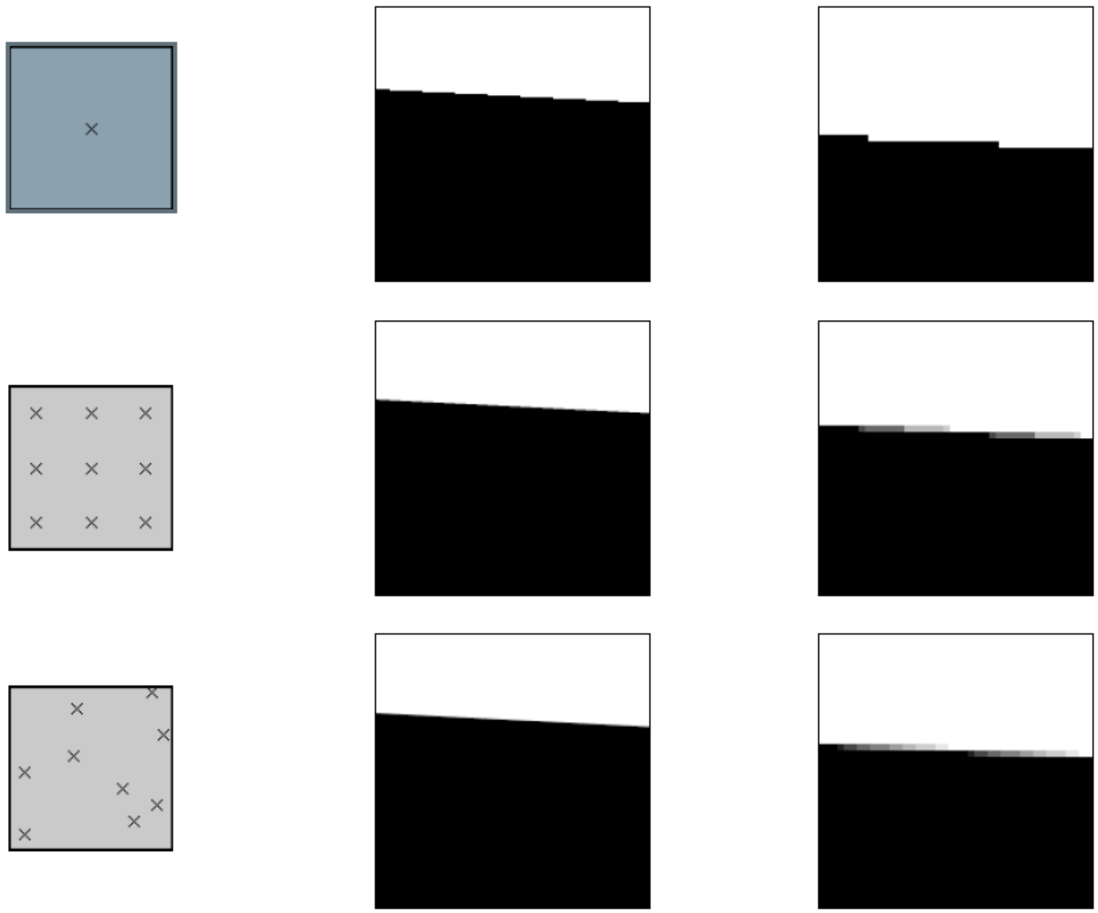
Supersampling



From wikipedia

Supersampling

- Human vision is sensitive to regularity

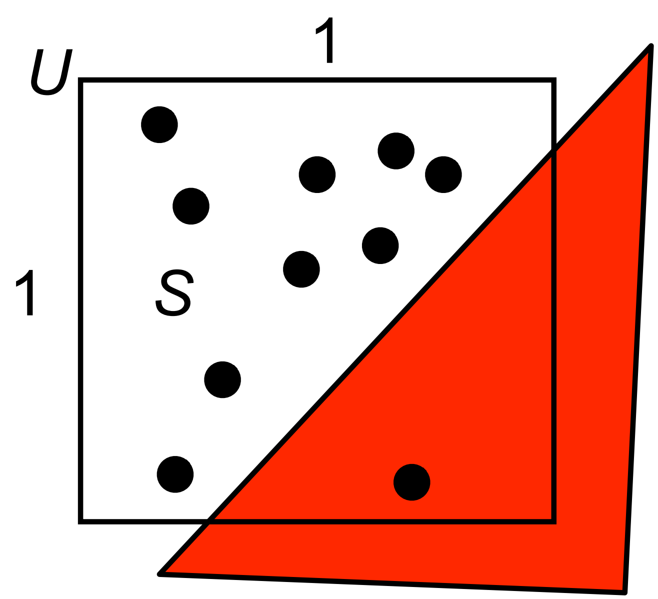


Supersampling

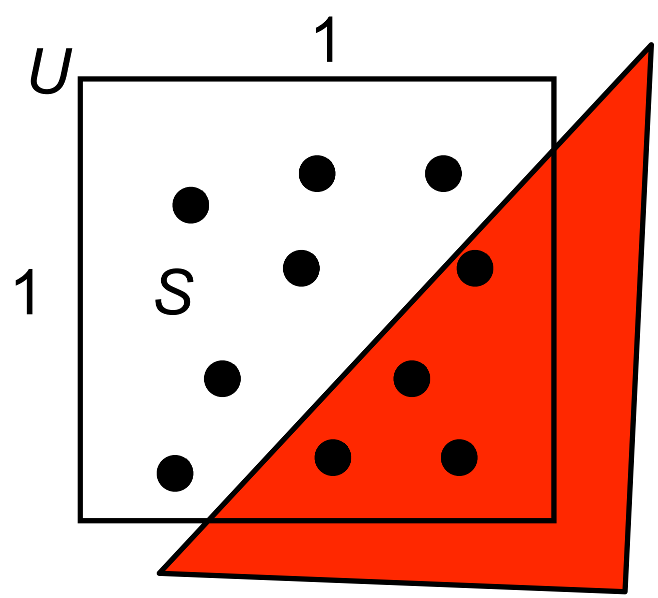
- We need generate our samples at random (regularity is BAD)
- Finding an optimal distribution depends on the objects to be rendered
- Instead, we generate a multiple random samplings and pick the one that is the best
- How do we measure the quality of a sample?

Supersampling

- Assume: our scene is made of polygons
- Most likely, one pixel will be intersected by an edge of a polygon



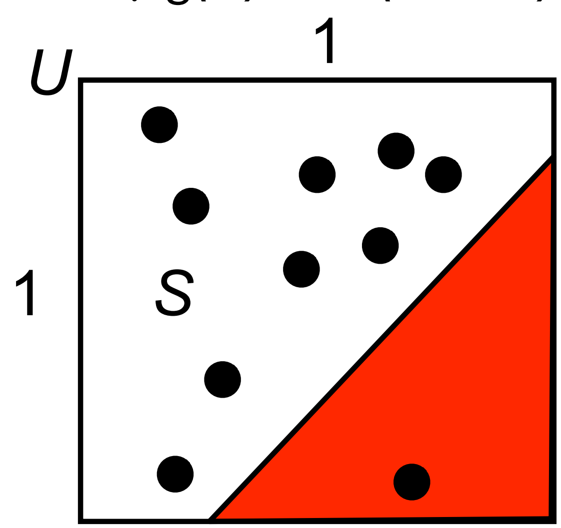
Bad samples



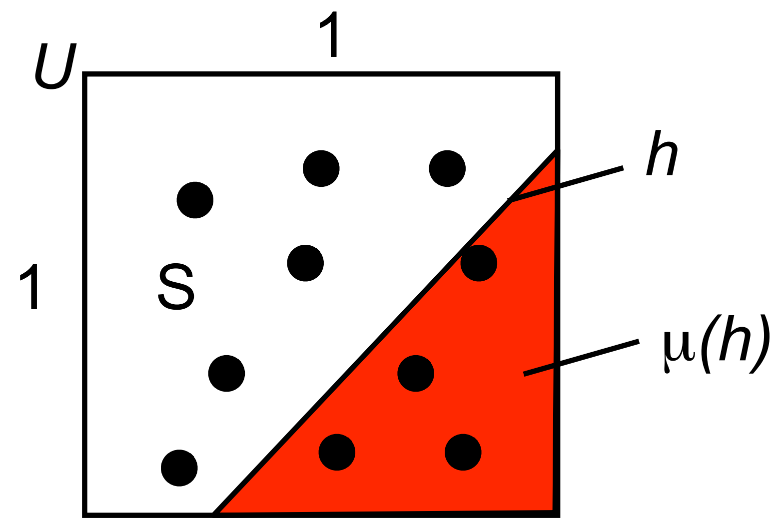
Better samples

Discrepancy

- Let's focus on the pixel
  - Pixel is a 1x1 square *U*
  - A half-plane *h* divide the square into 2 regions
  - $\mu(h)$ = area of $(U \cap h)$
  - $\mu_S(h) = \#(S \cap h)/\#(S)$



$\mu_S(h)=0.1$

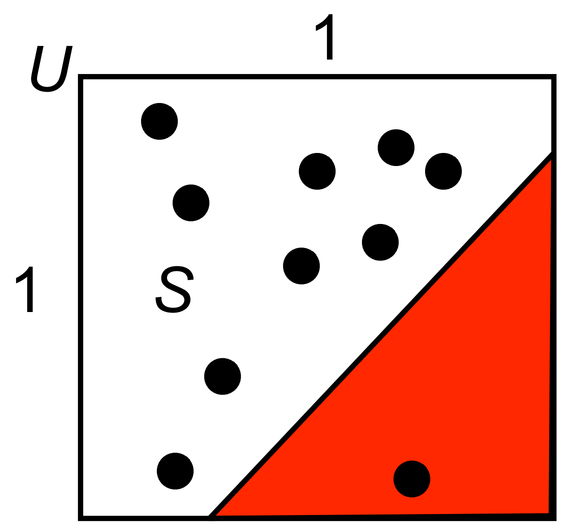


$\mu_S(h)=0.4$

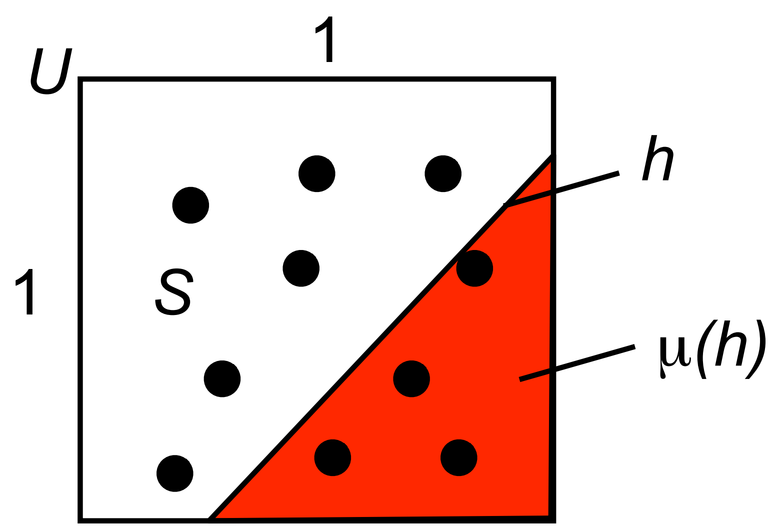
Discrepancy

- Let's focus on the pixel
  - *Discrepancy of S,* $\Delta_S(h) = | \mu_S(h) - \mu(h) |$

$\Delta_S(h)=0.35$



$\Delta_S(h)=0.05$



Assume $\mu(h)=0.45$

Discrepancy

- We want the discrepancy to be as small as possible
- Given a set of samples, what is its *worst discrepancy* for any given half-plane?

$$\Delta_S(H) = \max_{h \in H}(| \mu_S(h) - \mu(h) |) |$$

where H is a set of all possible half planes

Summary

- Given a set of samples, we can measure its quality by computing the worst discrepancy Δ_S

- We generate several sets of samples and pick the one with the best quality

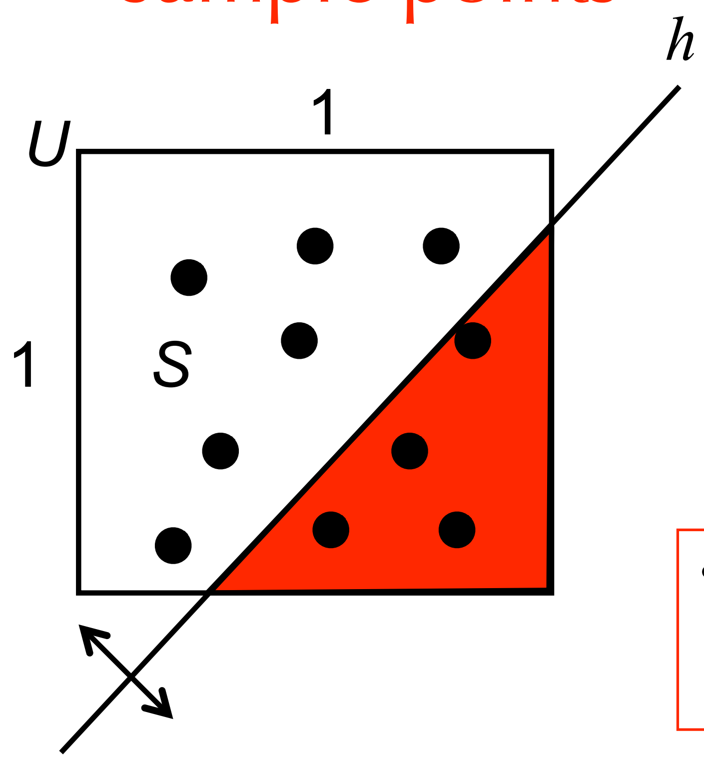
Note: A uniform distribution will have the lowest discrepancy, but a uniform distribution produces regularity.

Question: How to compute $\Delta_S(H) = \max_{h \in H}(|\, \mu_S(h) - \mu(h)\, |)$

There are an infinite number of possible half-planes…We can't just loop over all of them.

Computing Discrepancy

- The line of the half-plane of maximum discrepancy must pass through one of the sample points



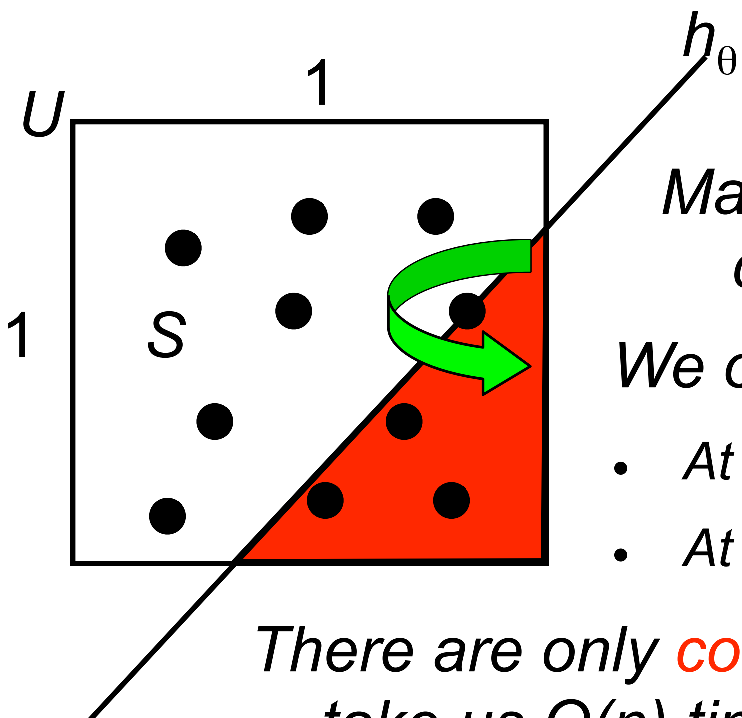
If h does not pass though any point, we can always increase the discrepancy by translating h until it touches at least one point

i.e, Same $\mu_S(h)$, but increasing/decreasing $\mu(h)$

- We only have to consider that cases
  1. When h passes through 1 point
  2. When h passes through 2 points

Computing Discrepancy

- When *h* passes through 1 point
  - There are infinite number of such *h*



Maximum discrepancy only happen at certain cases!!

We only need to find local extrema of $\mu(h_\theta)$

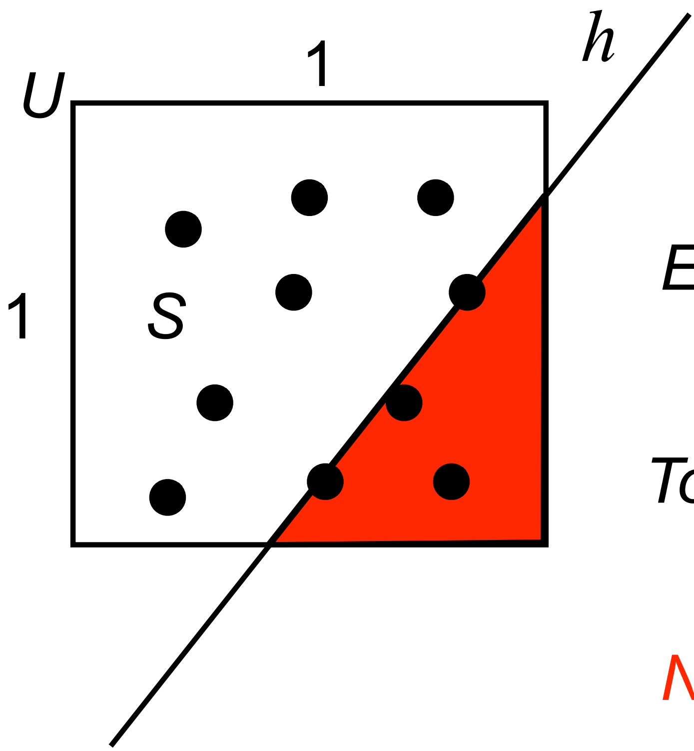
- *At local maximum of* $\mu(h_\theta)$ *and* $\mu_S(h)<\mu(h_\theta)$
- *At local minimum of* $\mu(h_\theta)$ *and* $\mu_S(h)>\mu(h_\theta)$

There are only constant number of these, each will take us O(n) time to compute $\Delta_S(h_\theta)$

Total time complexity is $O(n^2)$

Computing Discrepancy

- When *h* passes through 2 points
  - There are $O(n^2)$ of such h



Q: Do we have to consider h passes through 3 or more points

Each will take us O(n) time to compute $\Delta_S(h)$

Total time complexity is $O(n^3)$

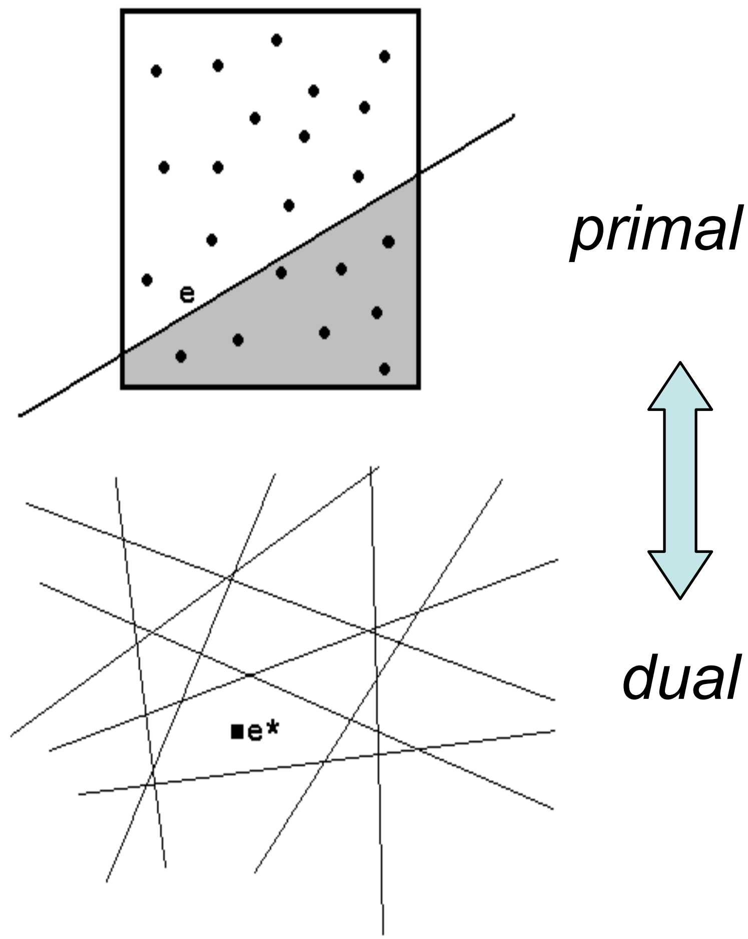
Need a faster algorithm for this!

Computing Discrepancy

Use arrangement!

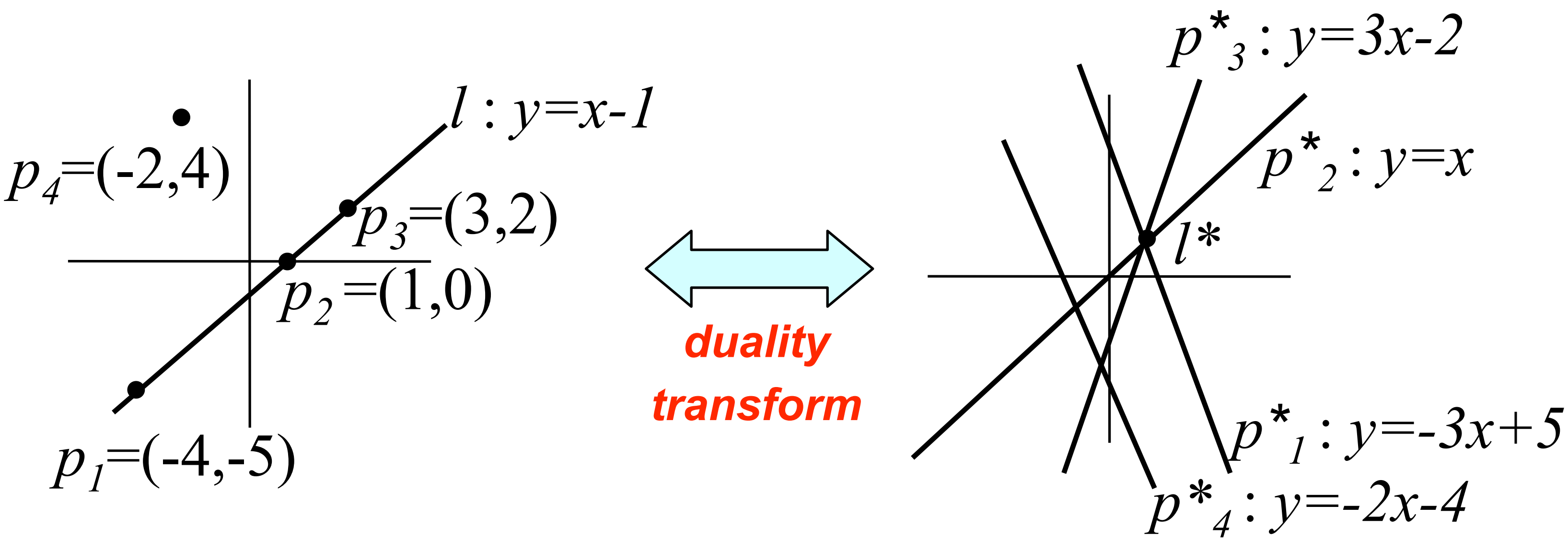
Construct an arrangement A of the *duals of the sample points* and h

Count the number of lines above and below h^*



Duality

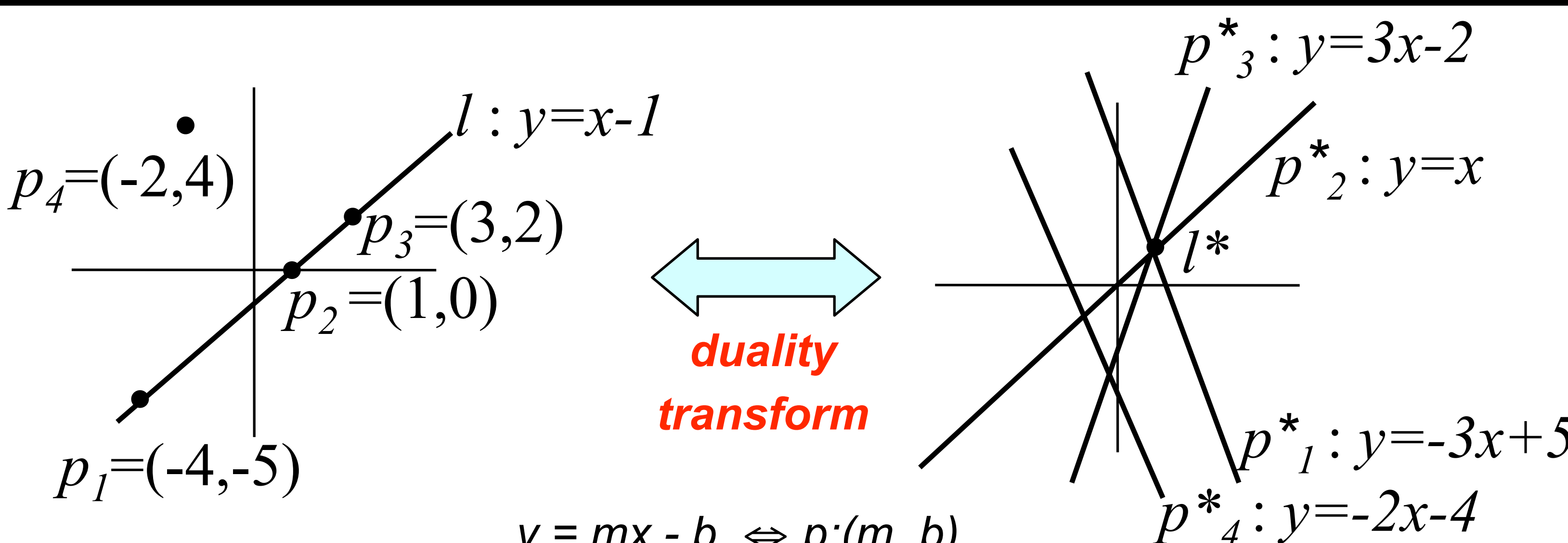
- We can map between different ways of interpreting 2D values
- Points (x,y) can be mapped in a one-to-one manner to lines (slope,intercept) in a different space



Duality Transforms

- Some duality transforms :-
  - Slope: $y = mx - b \Leftrightarrow p:(m, b)$
  - Polar: $ax + by = 1 \Leftrightarrow p:(a, b)$
  - Parabolic: $y = 2ax - b \Leftrightarrow p:(a, b)$

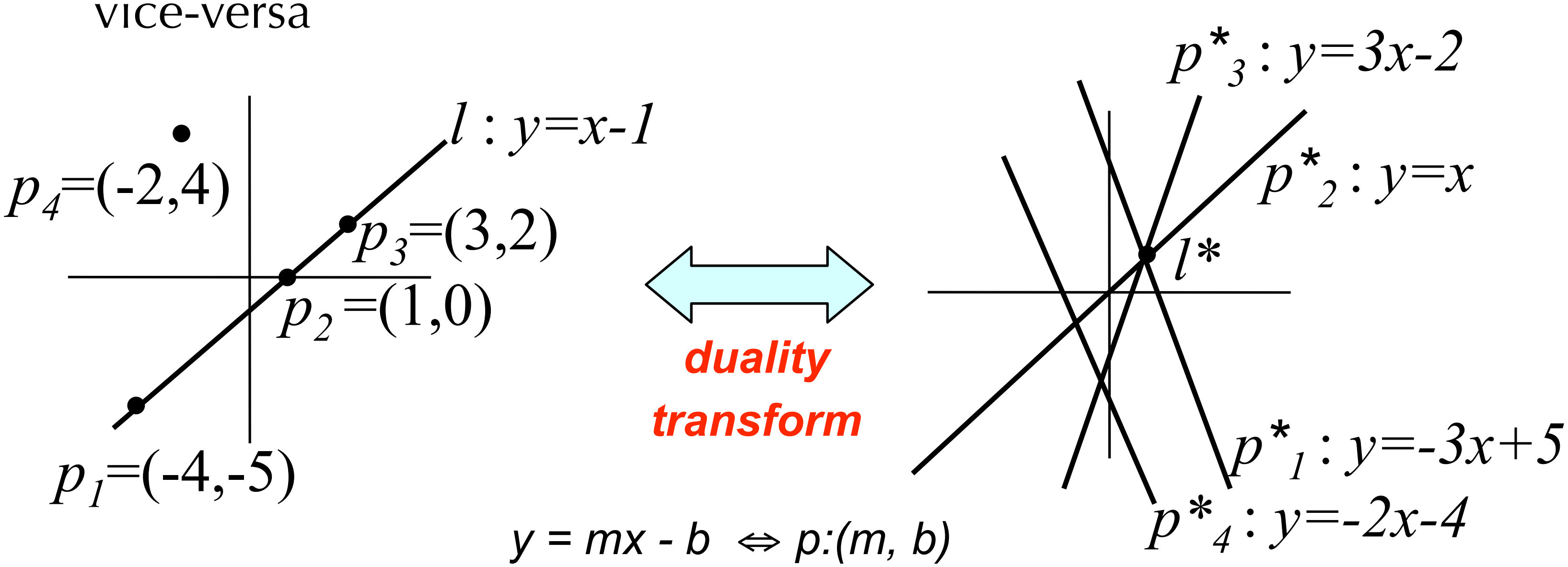
Q: When you move a point from left to right in primary space what will happen in dual space



$y = mx - b \Leftrightarrow p:(m, b)$

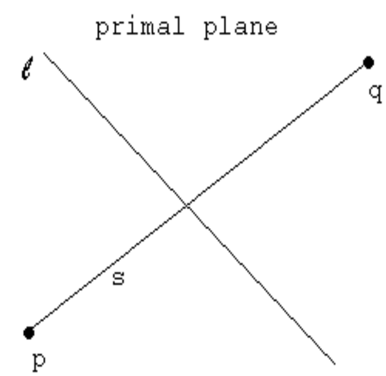
Duality Properties

- $(x^*)^* = x$
- Point p lies on line l iff point l^* lies on line p*
- Lines L_1 and L_2 intersect at a point p iff line p* passes thru L_1^* and L_2^*
- If point p lies above line L, then line p* lies below point L* and vice-versa

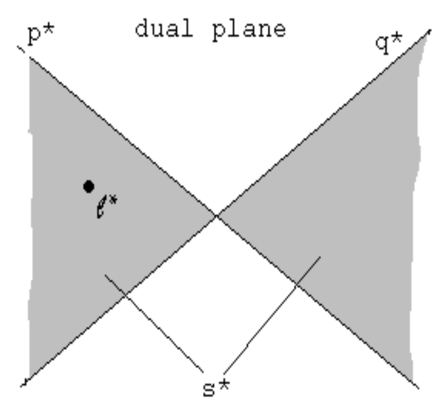


Additional Duality Properties

- This duality transform takes
  - points to lines, lines to points

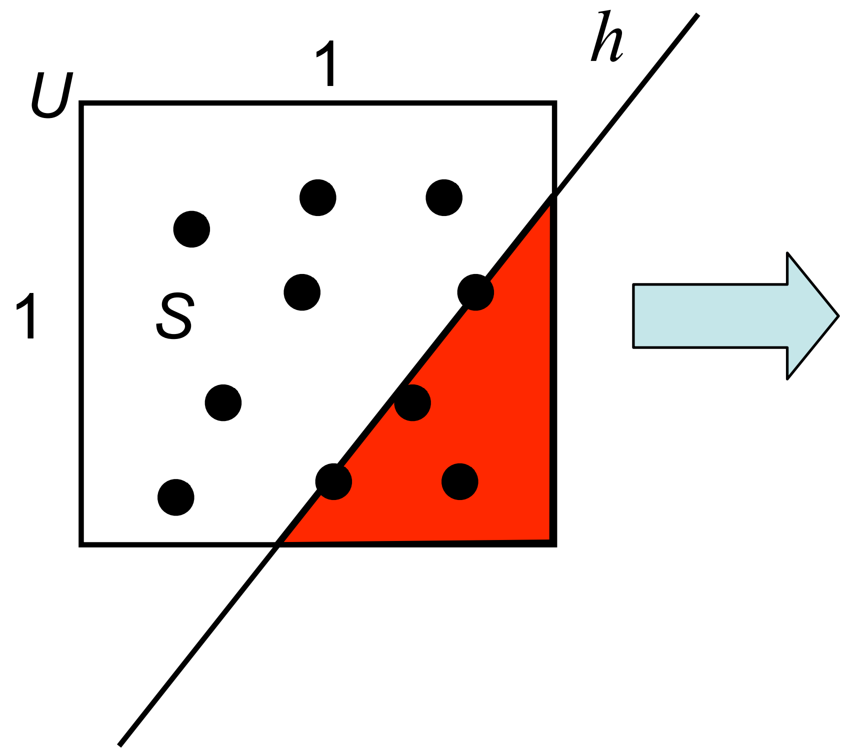


- For line segments, the dual of a line segment s between points p and q is the double wedge between lines $p*$ and $q*$ on the dual plane

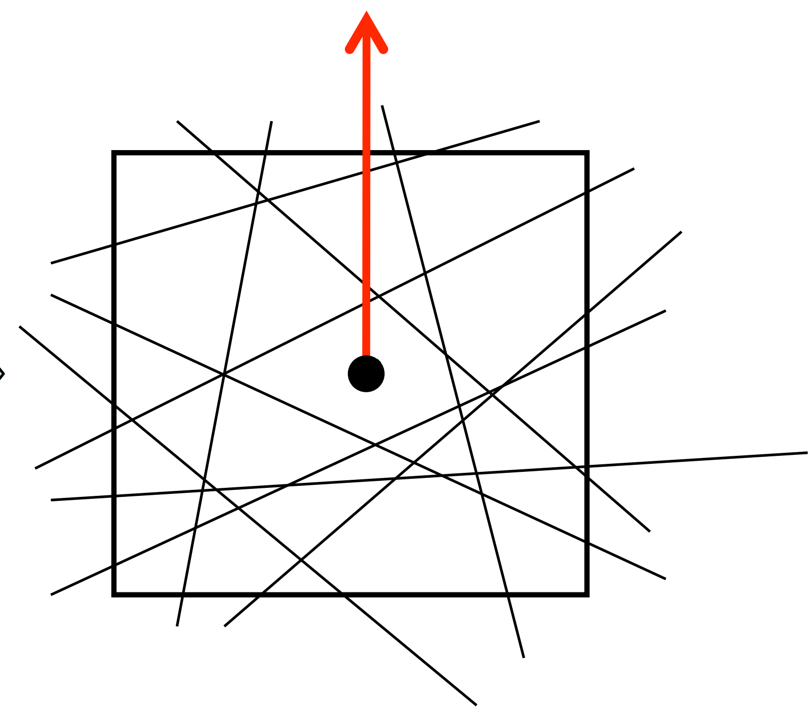


Computing Discrepancy

Determine how many sample points lie below a given line



Determine how many lines lie above a given vertex

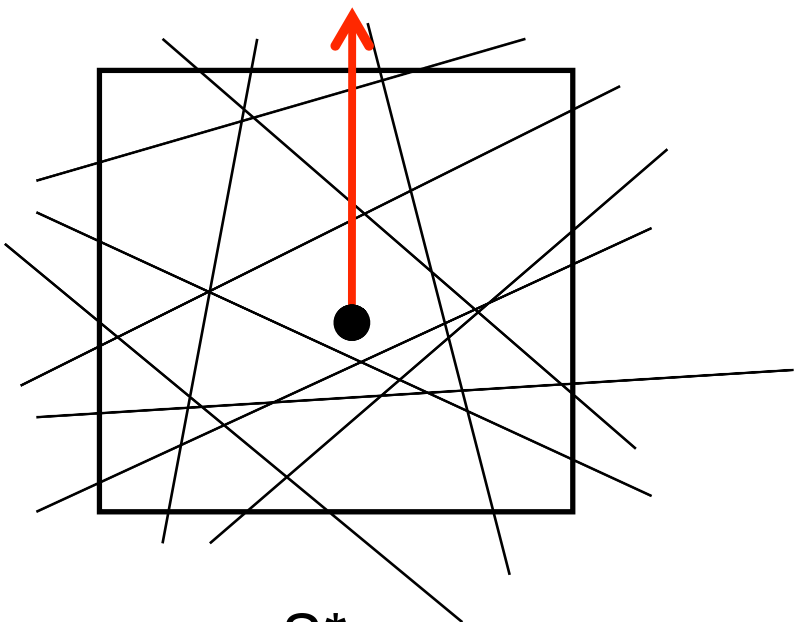


Why Duality?

- Looking at things on the dual plane provides new perspectives
  - It does makes problem harder or easier
- For problems dealing with points, their structure is more apparent
  - arrangement of lines

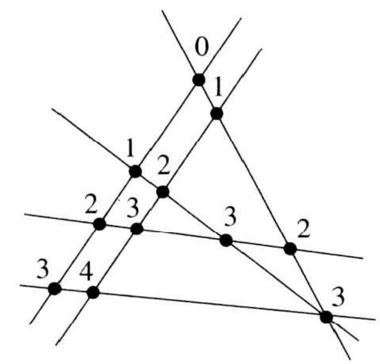
Computing Discrepancy

Determine how many lines lie **above** a given point



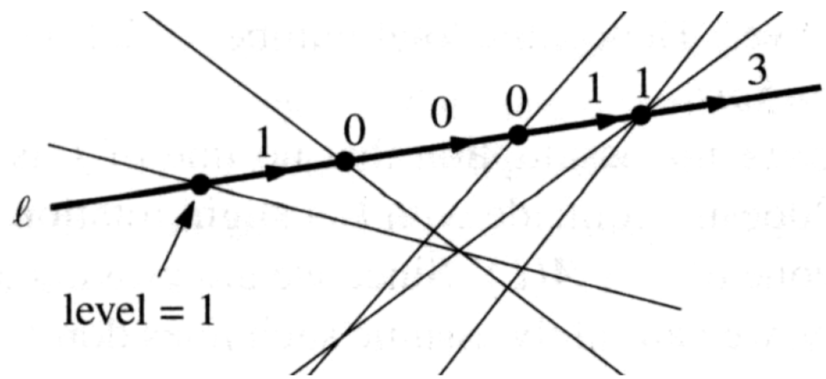
S^*

1. Compute arrangement of S^*
2. For each vertex we compute # of lines above the vertex



Computing Discrepancy

- For each line l in S^*
  - Compute the level of the leftmost vertex. $O(n)$
    - Check, for all other lines l_i, whether l_i is above that vertex
  - Walk along l from left to right to visit the other vertices on l, using the DCEL.
    - Walk along l, maintaining the level as we go (by inspecting the edges incident to each vertex we encounter).
  - $O(n)$ per line



Summary

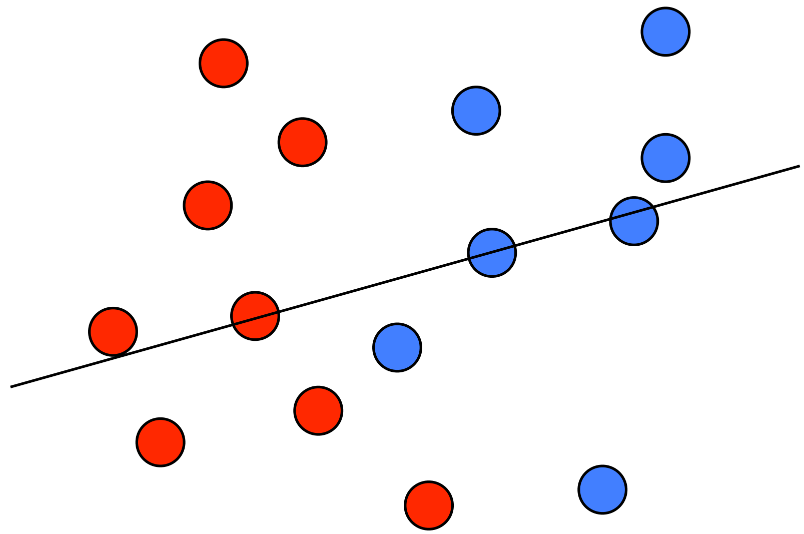
- Given the level of a vertex in the (dualized) arrangement, we can compute the discrete measure of *S wrt* the *h* that vertex corresponds to in $O(1)$ time.

- **We can compute all the interesting discrete measures in $O(n^2)$ time.**

- Thus we can compute all $\Delta_{S(h)}$ and hence Δ_S, in $O(n^2)$ time.

Ham (cheese) sandwich cut

- Given a sandwich, can you cut it so that each half has the same amount of ham, cheese and bread



Ham sandwich theorem

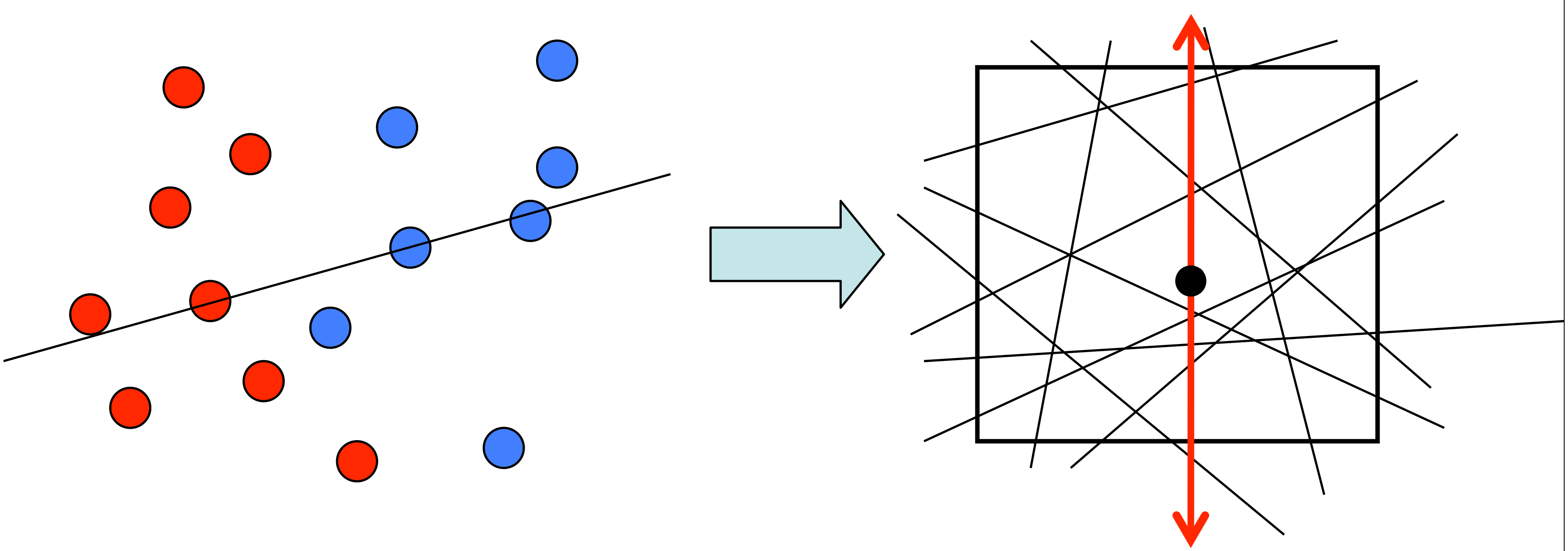
You can always do this

But how do you compute the cut?

Ham (cheese) sandwich cut

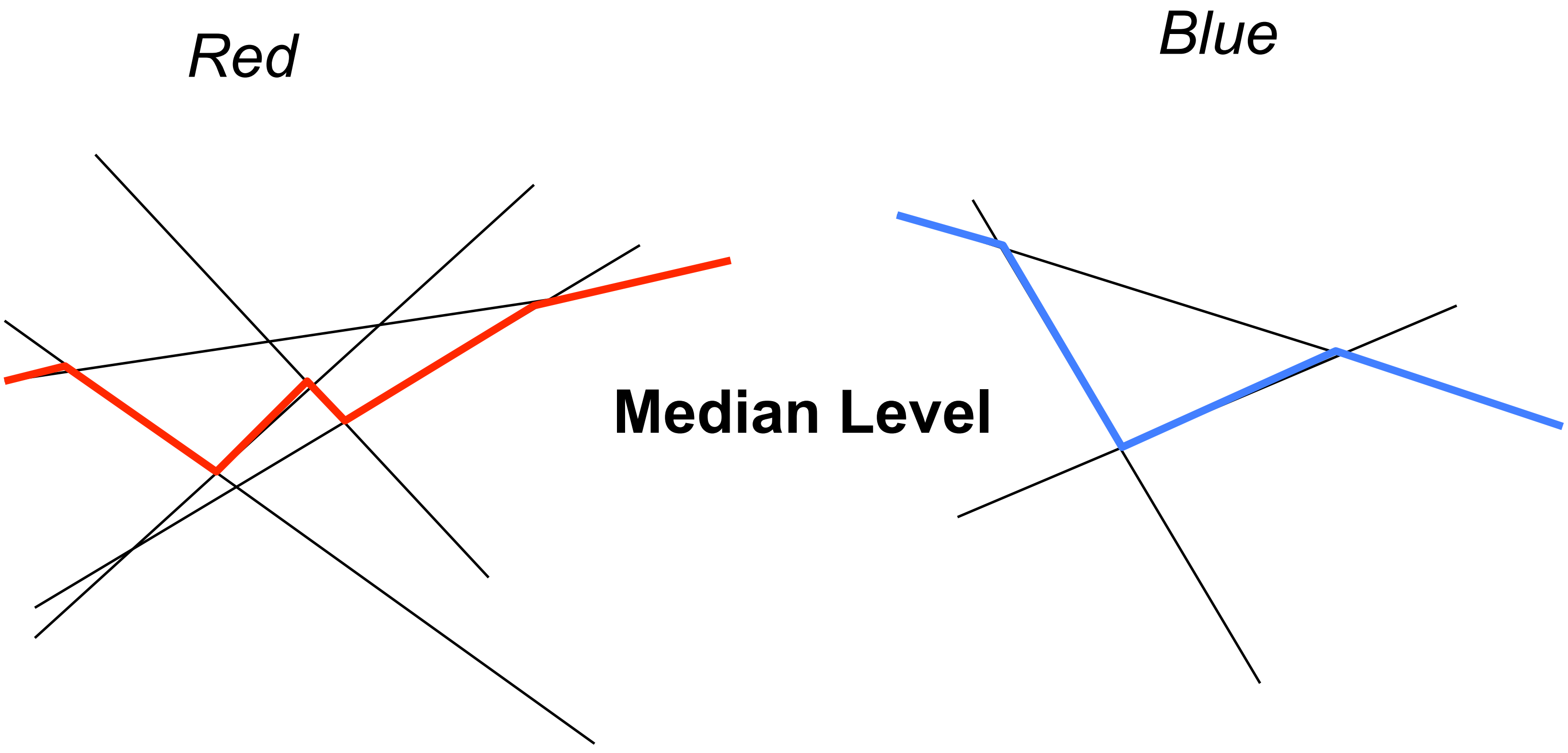
Find lines, such that the number of red/blue points above the line is the same as that below the lines

Find points, such that the number of red/blue lines above the points is the same as that below the points



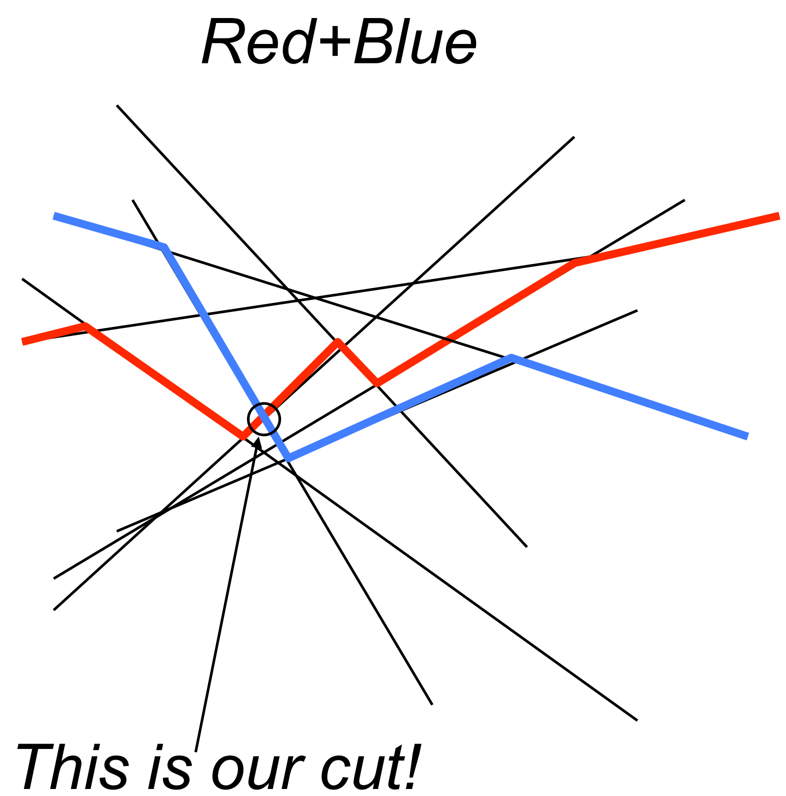
Ham (cheese) sandwich cut

- Consider blue/red (dual) points separately



Ham (cheese) sandwich cut

- Consider blue/red (dual) points separately



Constructing Vornoi Diagrams

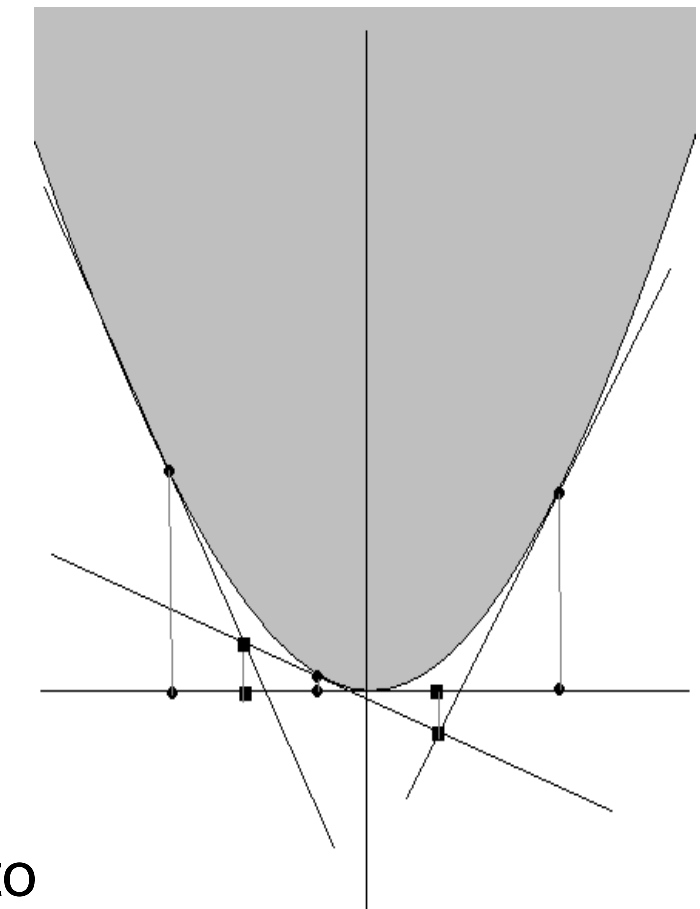
1. Lift each 1D point a to a 2D point (a, a^2)

- For each 2D point (a, a^2), find its dual $y=2ax-a^2$

1. Compute intersection of these lines in the dual space

2. Project the intersections of these duals onto the x-axis.

3. This is the Vornoi diagram.



The process generalizes to

- Higher order diagrams (by checking vertex levels!)
- Higher dimensional space