Computational Geometry

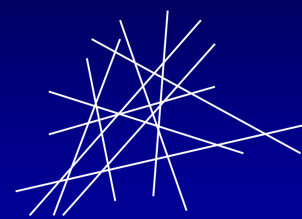
Chapter 10

Line Arrangements

10.1

On the Agenda

- Line Arrangements
- Applications



210.

Line Arrangement

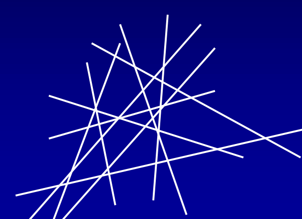
- Given a set L of n lines in the plane, their *arrangement* $A(L)$ is the plane subdivision induced by L.
- **Theorem**: The complexity of the arrangement of n lines is $\Theta(n^2)$ in the worst case.
- Proof:
  - Number of vertices $\leq \binom{n}{2} = \frac{n^2}{2} - \frac{n}{2}$ (each pair of different lines may intersect at most once).
  - Number of edges $\leq n^2$ (each line is cut into at most n pieces by at most n-1 other lines).
  - Number of faces $\leq \frac{n^2}{2} + \frac{n}{2} + 1$ (by Euler's formula and connecting all rays to a point at infinity).

  Equalities hold for lines in general position.
  (Show!)

310.

Line Arrangement

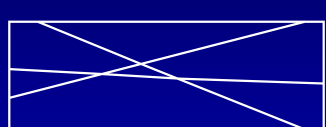
- Goal: compute this planar map (as a DCEL).
- A plane-sweep algorithm would require $\Theta(n^2 \log n)$ time: $\Theta(n^2)$ events, $\Theta(\log n)$ time each.



410.

An Incremental Algorithm

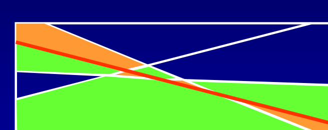
- **Input:** A set L of n lines in the plane.
- **Output:** The DCEL structure for the arrangement $A(L)$; I.e., the subdivision induced by L in a bounding box $B(L)$.
- The algorithm:
  - Compute a bounding box $B(L)$, and initialize the DCEL.
  - Insert one line after another. For each line locate the entry face, and update the arrangement, face by face, along the path of faces ("zone") traversed by the line. Move between faces by traversing edges.



510.

Line Arrangement Algorithm (cont.)

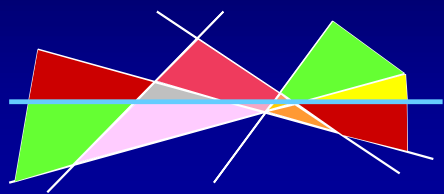
- After inserting the ith line, the complexity of the map is $O(i^2)$. ($\Theta(i^2)$ in the worst case—general position.)
- The time complexity of each insertion of a line depends on the complexity of its *zone.*



610.

Zone of a Line

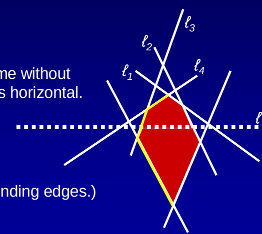
- The *zone* of a line ℓ in the arrangement $A(L)$ is the set of faces of $A(L)$ bordering on ℓ.
- The complexity of a zone is the total complexity of all its faces: the total sum of edges (or vertices) of these faces.



710.

The Zone Theorem

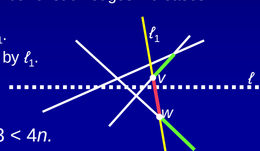
- **Theorem:** In an arrangement of n lines, the complexity of the zone of a line is $O(n)$.
- **Idea of Proof:**
  - Consider a line ℓ. Assume without loss of generality that ℓ is horizontal.
  - Count the number of *left bounding edges* in the zone, and prove that this is at most $4n$. (Same idea for right bounding edges.)



810.

Zone Complexity: Proof

- By induction on n.
- For n=1: Trivial.
- For n>1:
  - Let ℓ_1 be the rightmost line intersecting ℓ.
  - By the induction hypothesis, the zone of ℓ in $A(L \setminus \{\ell_1\})$ has at most $4(n-1)$ left bounding edges.
  - When adding ℓ_1, the number of such edges increases:
    - One new edge on ℓ_1.
    - Two old edges split by ℓ_1.



Hence, the new zone complexity is: $4(n-1)+3 < 4n$.

910.

Constructing the Arrangement

- The time required to insert a line ℓ_i is linear in the complexity of its zone, which is linear in the number of the existing lines. So the total time is

$$O(n^2) \quad + \quad \sum_{i=1}^{n} O(i) \quad = \quad O(n^2).$$

finding a bounding box (can be done faster!) — according to the zone theorem

- Note: This time does not depend on the order of insertion!

1010.

Duality

Primal ⇔ Dual relationship

A **point** $p=(x,y)$ (in the primal XY-plane) is mapped into the **line** p^* in the dual AB-plane:

$p^*=\{ (a,b) \;\; b=ax+y\}$

A **line** $l=\{(x,y) \mid y=ax+b\}$ (in the primal XY-plane) is mapped into the **point** l^* (in the dual AB-plane)

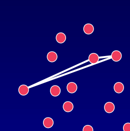
$l^*= (a,-b)$

Claim: p is above l iff p^* is above l^*
l contains p iff p^* contains l^*

1110.

Minimum-Area Triangle

- Given a set of n points, determine the three points that form the triangle of minimum area.
- Easy to solve in $\Theta(n^3)$ time.
- May be solved in $\Theta(n^2)$ time using the line arrangement in the dual plane.



Area of triangle= $hd/2$.

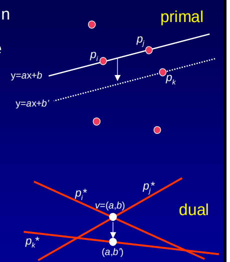
Base- d

Idea – check all bases of triangles. For each find the third vertex.

1210.

An $\Theta(n^2)$-Time Algorithm

- Construct the arrangement of dual lines in $\Theta(n^2)$ time.
- For each pair of points p_i and p_j (assume p_ip_j is the triangle base):
  - Identify the vertex v of the arrangement corresponding to the line through these points.
  - Find the line of the arrangement that is closest vertically to v.
  - Remember the best line so far.
- Output point corresponding to the best dual line.
- Questions:
  Why is it easier to find p_k* than p_k?
  Why do we look vertically?
  Why is the total runtime only $\Theta(n^2)$?



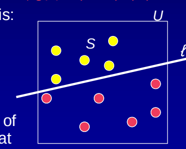
Discrepancy

- Given a set S of n points in the unit square $U=[0,1]^2$.
- For a given line ℓ, how many points lie below ℓ ?
  - Let h be halfplane below ℓ.
  - If the points are well distributed, this number should be close to $\mu(h)\cdot n$, where $\mu(h) = Area(U \cap h)$. Define $\mu_S(h) = |S \cap h| / |S|$.
  - The *discrepancy* of S with respect to h is:

    $\Delta_S(h) = |\mu(h)-\mu_S(h)|$

    and the *halfplane discrepancy* of S is

    $\Delta(S) = \sup_h \Delta_S(h)$

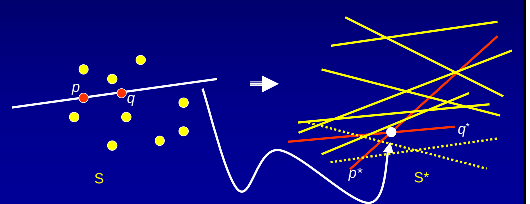


Observation: To compute the discrepancy of S, it suffices to consider those halfplanes that pass through pairs of points.

Naive algorithm: $\Theta(n^3)$ time.

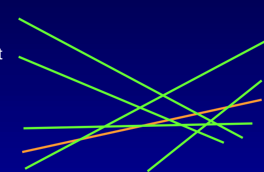
Computing Discrepancy

- In the dual plane, this is equivalent to counting the number of dual lines *above* the dual point.



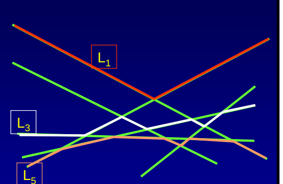
Computing Discrepancy

- For every vertex in $A(S^*)$, compute the number of lines above it, passing through it (two if in general position), or lying below it.
- These three numbers sum to n, so it suffices to compute only two of them.
- From the DCEL structure we know how many lines pass through each vertex.



Levels of an Arrangement

- A point is at *level k* in an arrangement of n lines if there are at most k-1 lines above this point and at most n-k lines below this point.
- There are n levels in an arrangement of n lines.
- A vertex can be on multiple levels, depending on the number of lines passing through it.



An $\Theta(n^2)$-Time Algorithm

- Construct the dual arrangement.
- For each line, compute the levels of all its vertices:
  - Compute the levels of the left infinite rays by sorting slopes. O(n log n) time.
  - Traverse all the lines from left to right, incrementing or decrementing the level, depending on the direction (slope) of the crossing line. $\Theta(n)$ time for each line.
- Total: $\Theta(n^2)$ time.