Computational Geometry Intro

15-451
10/22/20

So far we have worked in 1D.
eg sorting, searching, and priority queues
The next few lectures consider 2D.

<u>Apps</u> (even for 2D)	more dim
Graphics	3D Sci Comp
Robotics	Machine Learning
Geo Info Systems	
CAD/CAM	
Sci Comp	

Basic Approach

Build complex objects out of simple objects.

eg Image ≡ array of dots
Integrated Circuit ≡ collection of triangles

Basic Alg Design Approaches

1) Divide-and-conquer
   eg Merge Sort
   eg Quick Sort
2) Sweep-Line
3) Random-Incremental

Topics to Cover

1) Intro Primative/Atomic operations
2) Sweep-Line for line seg intersection.
3) 2D Linear Programming
4) 2D Convex Hull
5) ??

Abstract Obj	Possible Repre
Real Number	floating point, big number
Point $\in R^2$	Pair of Reals
Line	Pair of Points
Line Segment	Pair of Points
Triangle	Triple of Points

Using Points to Generate Objects

Suppose $p_1, \ldots, p_k \in \mathbb{R}^d$

Linear Combinations

Subspaces $\equiv \sum \alpha_i p_i$ for $\alpha_i \in \mathbb{R}$

Affine Combinations

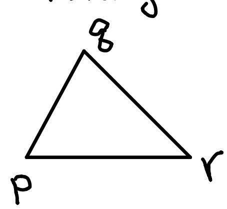
Plane, HyperPlanes $= \sum \alpha_i p_i$ st

$\alpha_i \in \mathbb{R}$ & $\sum \alpha_i = 1$

Convex Combinations

Body $= \sum \alpha_i p_i$ st $\sum \alpha_i = 1$ & $\alpha_i \geq 0$

eg Triangle



$= \{ \alpha p + \beta q + \gamma r \mid \alpha + \beta + \gamma = 1$ &

$\alpha, \beta, \gamma \geq 0 \}$

Primative Operations

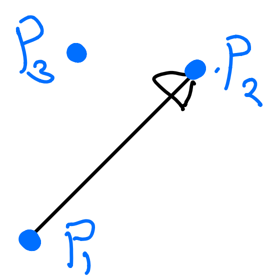
1) Equality $P = q$?

2) Line segment intersection test.

3) Line side test.

<u>Input</u>: (P_1, P_2, P_3) $P_i \in \mathbb{R}^2$

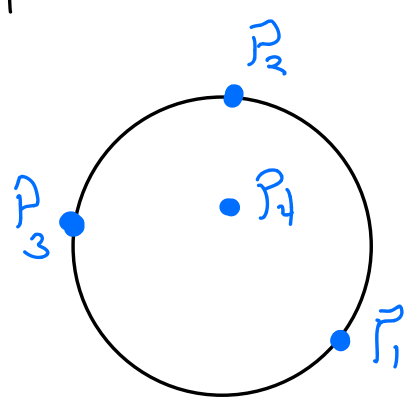
<u>Output</u>: True if P_3 is "left" of ray $P_1 \longrightarrow P_2$



4) In circle test.

<u>Input</u>: (P_1, P_2, P_3, P_4) $P_i \in \mathbb{R}^2$

<u>Output</u>: True if P_4 is "in" circle (P_1, P_2, P_3)



Line Segment Intersection

2) Consider Seg

$S_1 = [P_1, P_2]$ & $S_2 = [P_3, P_4]$

Let $P_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$

Claim $L_1 \cap L_2 \neq \emptyset$ iff

$\exists \alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \mathbb{R}$ s.t.

$$\begin{pmatrix} | & | \\ P_1 & P_2 \\ | & | \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} | & | \\ P_3 & P_4 \\ | & | \end{pmatrix} \begin{pmatrix} \alpha_3 \\ \alpha_4 \end{pmatrix}$$

$\alpha_1 + \alpha_2 = 1 \qquad \alpha_3 + \alpha_4 = 1$

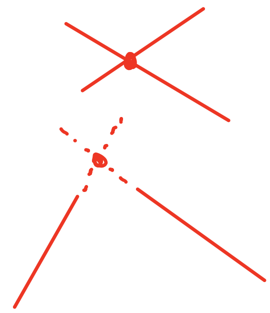
$\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq 0$

Let $A = \begin{pmatrix} x_1 & x_2 & -x_3 & -x_4 \\ y_1 & y_2 & -y_3 & -y_4 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$

<u>Case1</u> A is non singular

then solve $A \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$ $(*)$

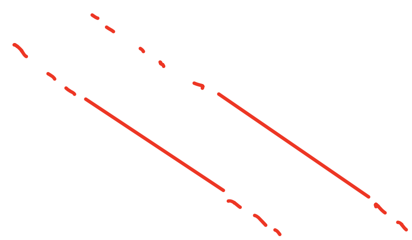
Return: true iff $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq 0$



<u>Case2</u> A is singular

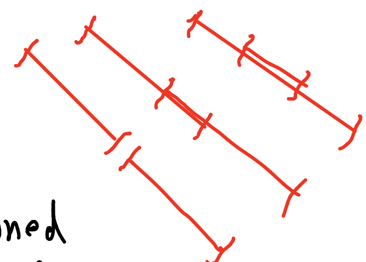
<u>Case2a</u> no solution to $(*)$.

Return: false.



<u>Case2b</u> $\exists$ solution to $(*)$

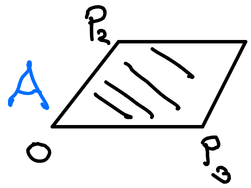
3 subcases



Check if an endpoint is contained in the other segment.

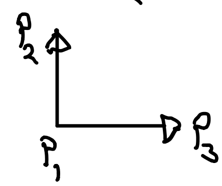
3) Line side test:

First assume that $P_1 = 0$



Claim: $\det\begin{pmatrix} x_2 & x_3 \\ y_2 & y_3 \end{pmatrix} = \pm$ Area of A

eg $\det\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = -1$ ie P_3 is right of $P_1 \to P_2$



<u>Claim</u> $LST(P_1 P_2 P_3) = \det\begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{pmatrix}$

<u>pf</u>

$$RHS = \det\begin{pmatrix} x_1 & x_2' & x_3' \\ y_1 & y_2' & y_3' \\ 1 & 0 & 0 \end{pmatrix} = \det\begin{pmatrix} x_2' & x_3' \\ y_2' & y_3' \end{pmatrix}$$

We just tranlated P_1 to origin

The Line Segment Intersection Prob

Input: n line segs
Output: All I intersections

<u>Naive</u>: $O(n^2)$ (This is worst case optimal)

<u>Goal</u>: Output sensitive alg

<u>Known</u>: $O(n \log n + |I|)$ time

<u>Today</u>: $O((n + |I|) \log n)$ time.

Application: Map Overlay

Alg today: sweepline

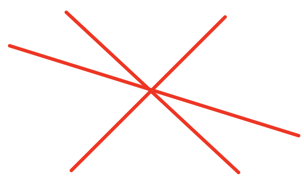
Optimal alg: Random Incremental

The Sweep Line Alg

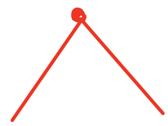
Segments: $S = \{s_1, \dots, s_n\}$

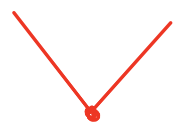
We assume: 1) no horizontal segs

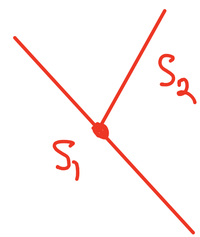
2) Cases not handled today!



≥ 3 segs at a point.







Let $\bar{P} \equiv$ Seg endpoints

$I \equiv$ Seg intersections.

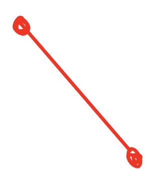
Events $\equiv \bar{P} \cup I$

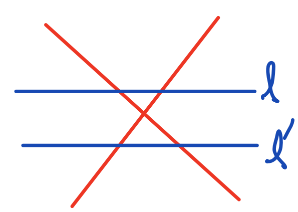
$\ell \equiv$ hori line disjoint from $\bar{P} \cup I$

Consider the linear order on $\{s \in S \mid s \cap \ell \neq \emptyset\}$

<u>Note</u> Order only changes after an event.

1) new live seg
2) seg dies
3) 2 segs swap in order





<u>Idea1</u>: Store the live ordered segs in Balanced BST, D.

<u>Idea2</u>: "Sweep" a line ℓ from top to bottom. Stopping at each event. (Just after)

<u>Problem</u>: We do not know I!

<u>Solution</u>: Compute intersections just-in-time.

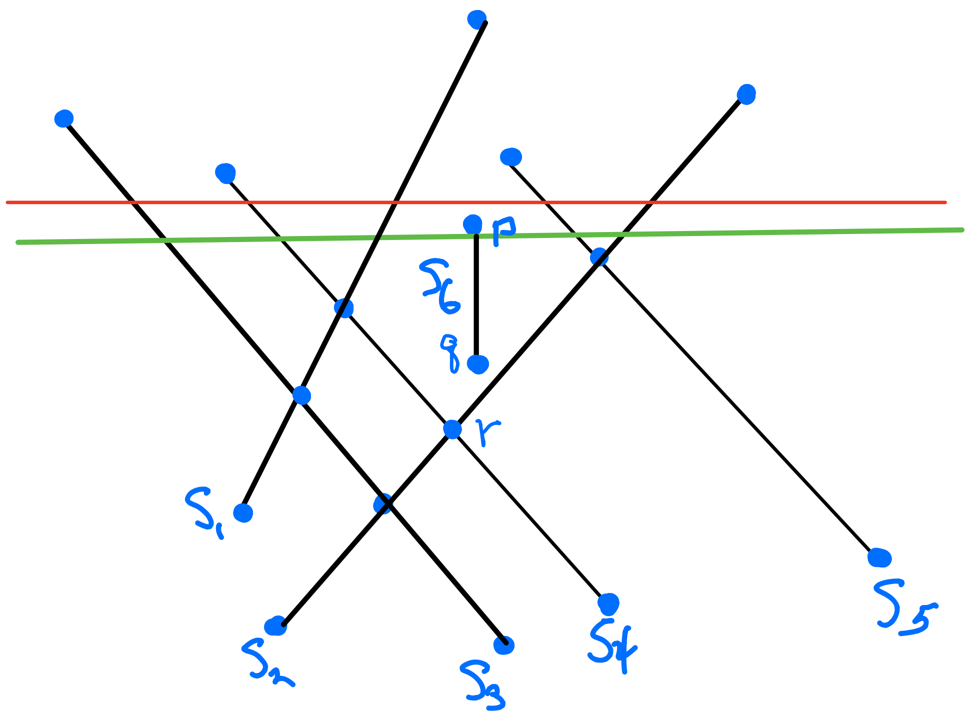
Claim: If next event is $S \cap S'$ then S & S' are consecutive "neig".

Keep a priority queue Q_ℓ of known events.

Inductively: Q_ℓ contains

1) Endpoint of $\vec{P}$ below ℓ.
2) Neig inter below ℓ.

Case Upper Endpoint



Just before P:

$(S_3, S_4, S_1, S_5, S_2)$

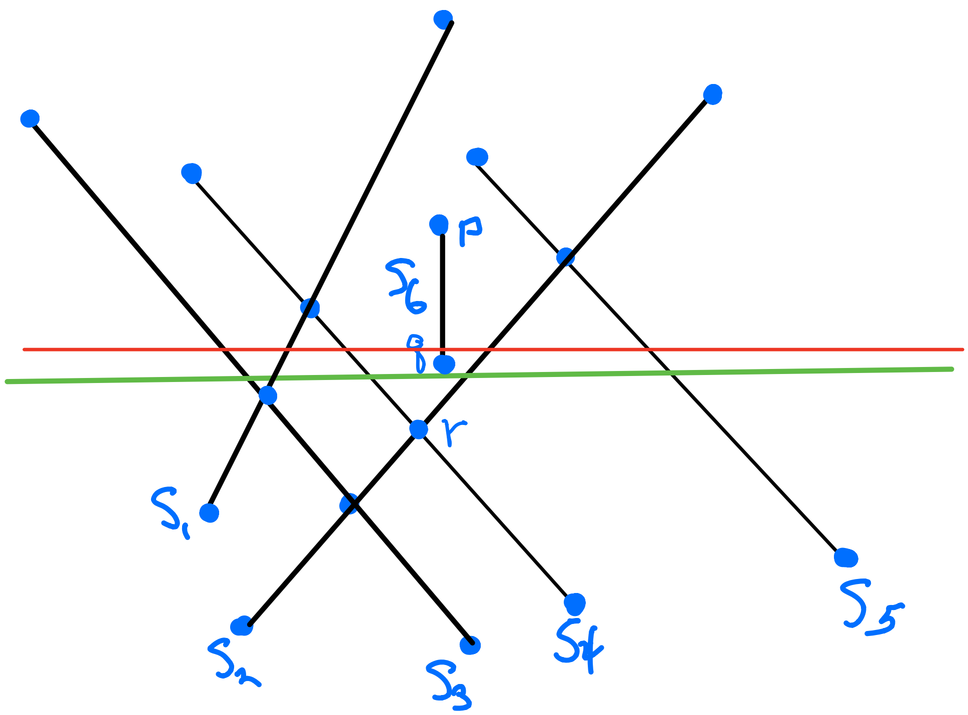
Just after P:

$(S_3, S_4, S_1, S_6, S_5, S_2)$

Intersections checked

$S_1 \cap S_6$ & $S_6 \cap S_5$

Case Lower Endpoint



Just before q:

$(s_3, s_1, s_4, s_6, s_2, s_5)$

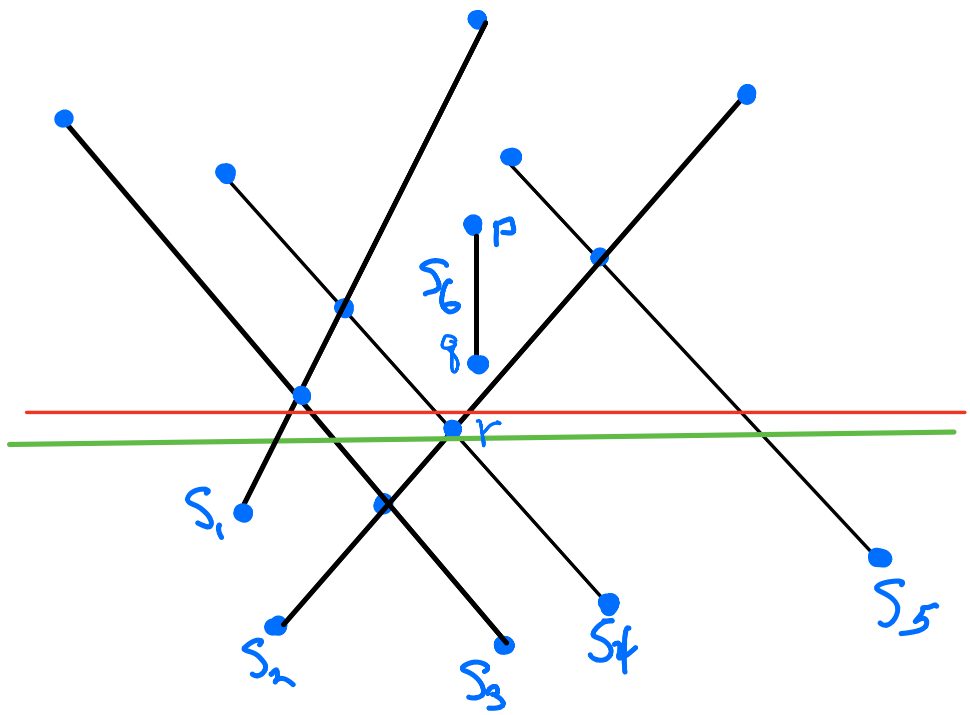
Just after q:

$(s_3, s_1, s_4, s_2, s_5)$

Intersection checked

$s_2 \cap s_4$

Case Intersection Point



Just before r:

$(S_1, S_3, S_4, S_2, S_5)$

Just after r:

$(S_1, S_3, S_2, S_4, S_5)$

Intersections checked

$S_3 \cap S_2$ & $S_4 \cap S_5$

Sweepline Algorithm

Keep a priority queue Q_ℓ of known events.

Inductively: Q_ℓ contains

1) Endpoint of $\vec{P}$ below ℓ.
2) Neig inter below ℓ.

Alg Sweep Line$(S_1, \ldots, S_n)$

Insert $\vec{P}$ into Q

While $Q \neq \emptyset$

a) $P = \text{ExtractMax}(Q)$

b) HandleEvent(P)

Handle Events End-Points

Proc **HandleEvent**(P)

1) If P is upperend of S then
   a) insert(S, D)
   b) add-inter(left(S), S, Q)
   c) add-inter(S, right(S), Q)
2) If P is lowerend of S then
   a) add-inter(left(S), right(S), Q)
   b) delete(S, D)

Handle Events Intersection

Proc HandleEvent(p)

3) if $p = s \cap s'$ and $s < s'$

   a) swap(s, s', D)

   b) add-inter$(\text{left}(s'), s', Q)$

   c) add-inter$(s, \text{right}(s), Q)$

   d) Report p.

Sweepline Timing

Proc Handle Event(P)

1) If P is upperend of S then
   a) insert(S, D)
   b) add-inter(left(S), S, Q)
   c) add-inter(S, right(S), Q)
2) If P is lowerend of S then
   a) add-inter(left(S), right(S), Q)
   b) delete(S, D)
3) if $P = S \cap S'$ and $S < S'$
   a) swap(S, S', D)
   b) add-inter(left(S'), S', Q)
   c) add-inter(S, right(S), Q)
   d) Report P.

Step	Cost	#
1a	$\log n$	n
1b	$\log n$	n
1c	$\log n$	n
2a	$\log n$	n
2b	$\log n$	n
3a	$\log n$	I
3b	$\log n$	I
3c	$\log n$	I

$$O((n+I)\log n)$$

Timing Question?

Why do we have an $O(I \log n)$ term?

Answer: We returned the intersections in sorted order!

To get an output output optimal algorithm

The simplest is

1) Random Incremental
2) Beckwards Analysis

Topics on next lecture.