Sweep Paradigm

Line
Ray
Circle

Some object sweeps through the plane.

At some finitely many position of the sweeping object, some events happen.

If we can handle these events, our geometric problem is solved.

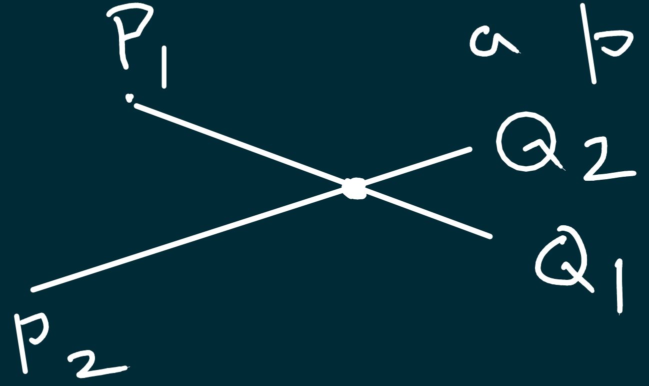
Line Segment Intersection

n line segments

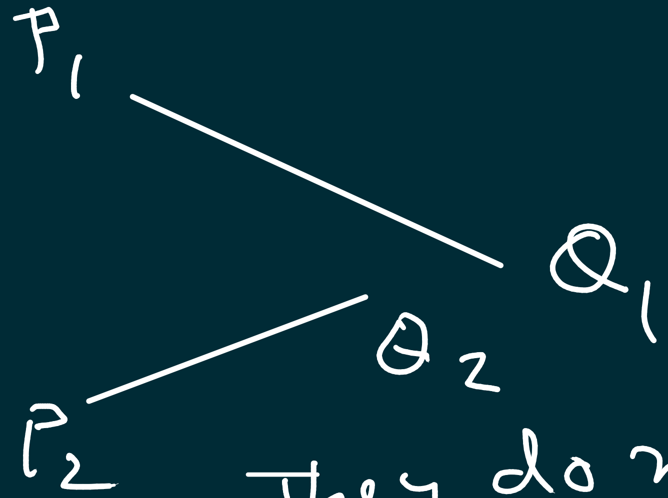
$L_1, L_2, \dots, L_n$

$L_i = (P_i, Q_i)$

a point



They intersect



They do not intersect

To compute all the intersection points.

$\binom{n}{2}$ pairs (L_i, L_j)

$\Theta(n^2)$ time, $O(1)$

Exhaustive search / naive

h - the actual no. of intersections

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

if $h = o(n^2/\log n)$, this is better.

$O(n)$ space

Line sweep algorithm

$L_1, \ldots, L_n$ should be in general position

- No two x-coordinates of the endpoints and the intersection points are the same.
- No L_i is vertical
- No two L_i, L_j are parallel.

A vertical line sweeps from $x = -\infty$ to $x = +\infty$

L - sweeping line

Infinitely many positions

start = $\min(x(P_1), x(P_2), \ldots, x(P_n))$

end = $\max(x(Q_1), x(Q_2), \ldots, x(Q_n))$

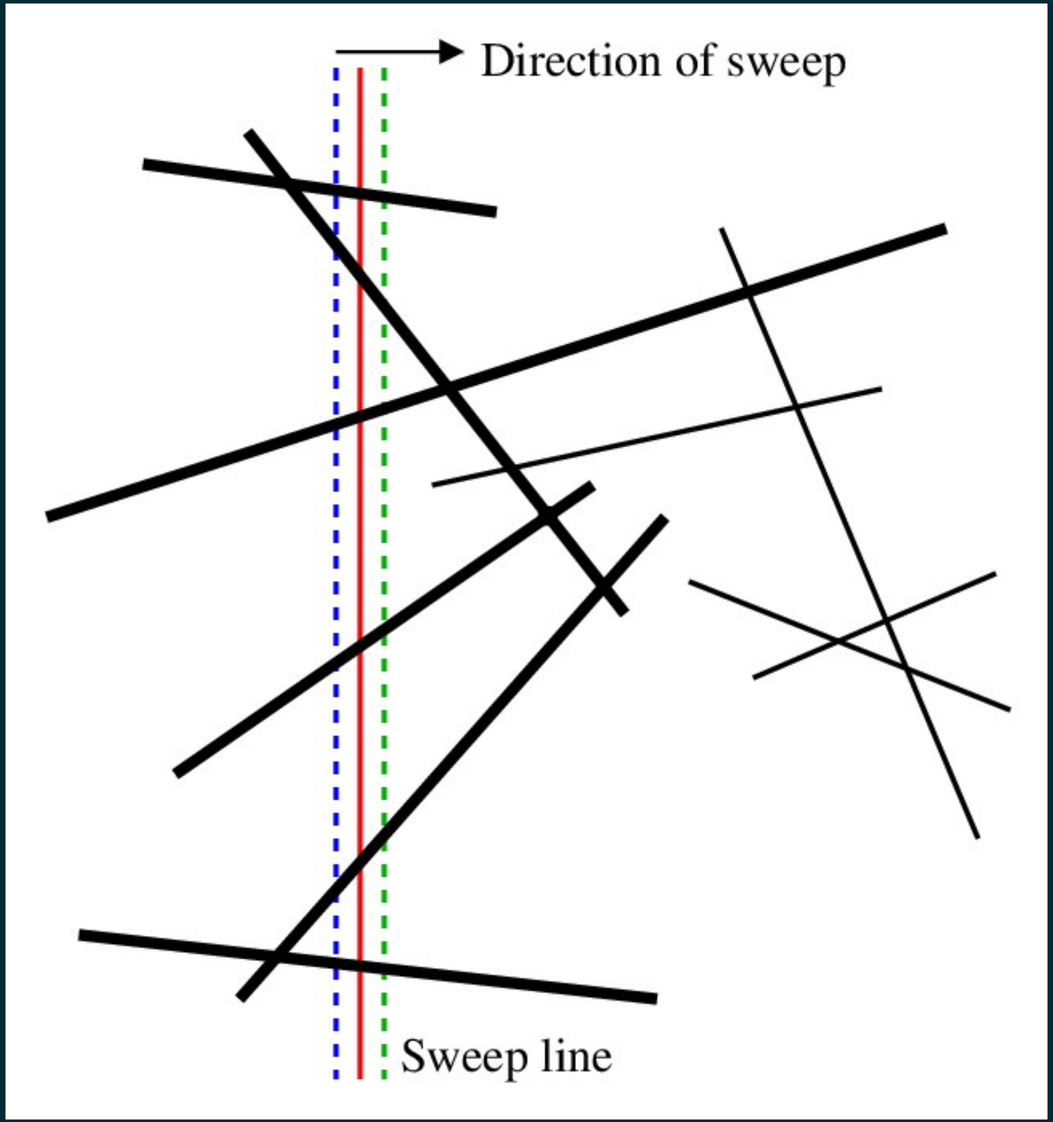
L moving from start to end.

At some finite no of position of L, events will happen

- Enter segment n
- leave segment n
- intersection h

$2n+h$

To handle each in $O(\log n)$ time



start

end

S - sweep line information

The set of all active lines sorted from top to bottom

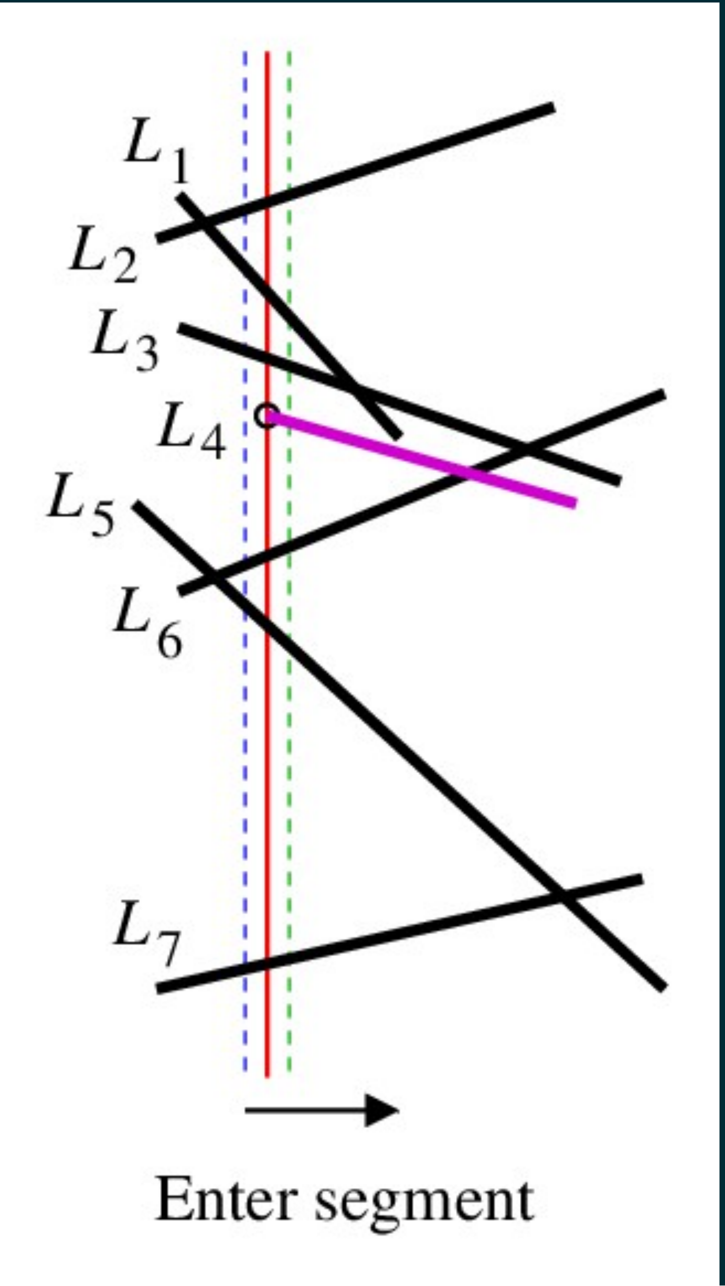
S will not store the y-coordinates of the intersection L with the active lines

An event queue E

- segment endpoints to the right of the current position of L

- L_i, L_j are consecutive in S.
  $\cap(L_i, L_j) \rightarrow$ x-coordinate of the intersection point.
  ↳ is to the right of L

E - store x-coordinates only



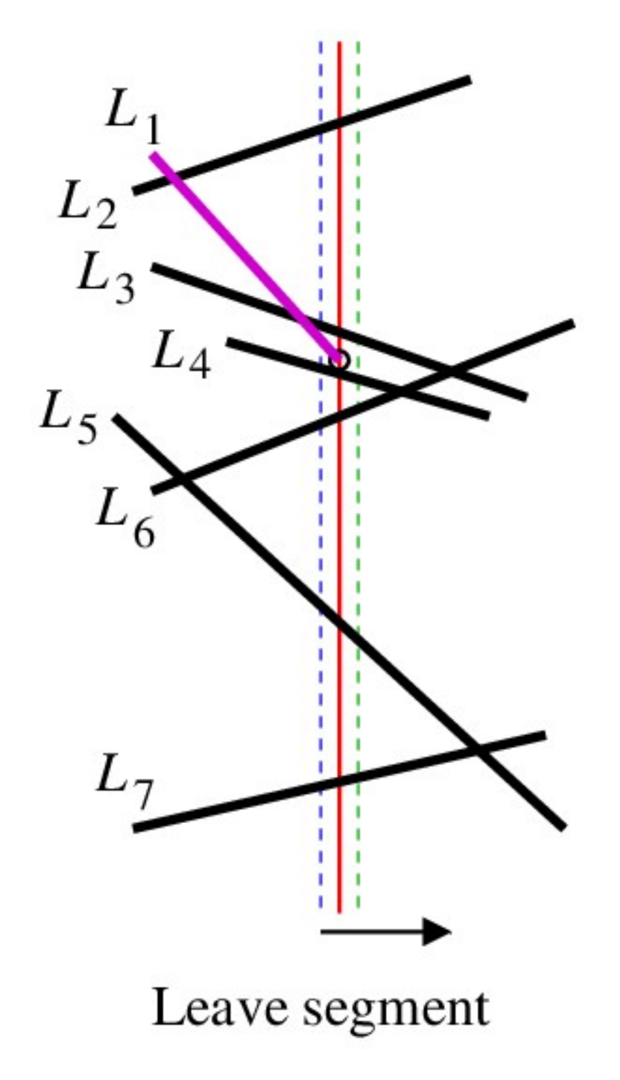
S : $L_2, L_1, L_3, L_6, L_5, L_7$

L_4

E : Delete $\cap(L_3, L_6)$

L_3, L_4 do not intersect

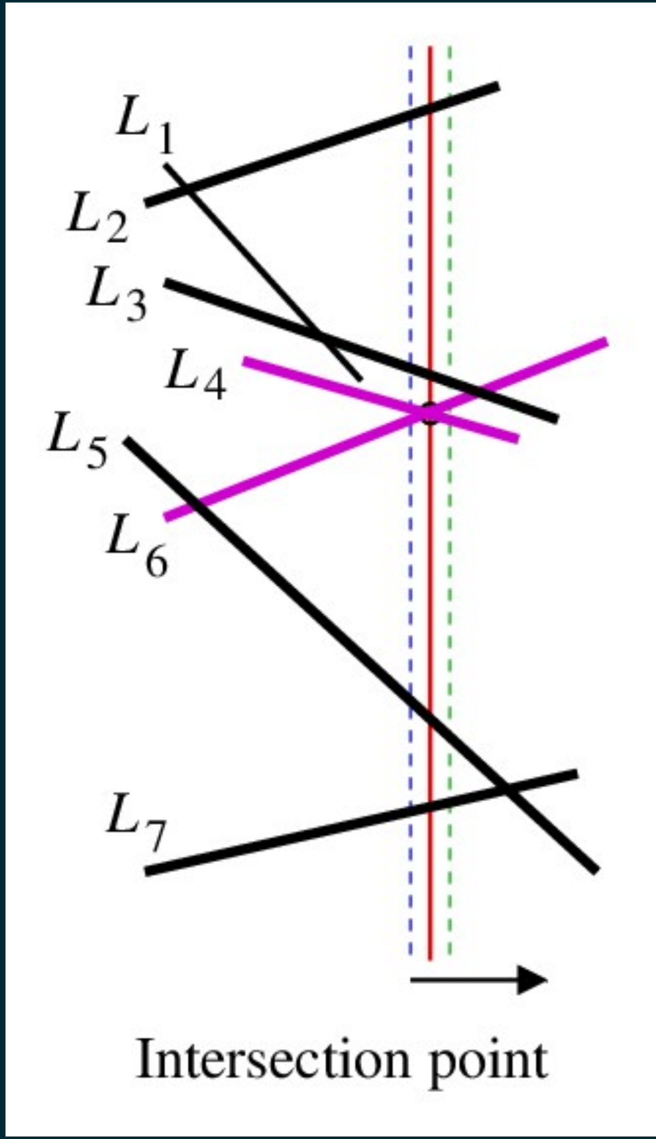
Insert $\cap(L_4, L_6)$



S : $L_2, L_3, \cancel{L_1}, L_4, L_6, L_5, L_7$

E : $L_3, L_4 \rightarrow$ now are adjacent

They do not intersect.



L_6, L_4

S: $L_2, L_3, L_4, L_6, L_5, L_7$

Report $\cap(L_4, L_6)$

E: $\cap(L_3, L_4)$
$\cap(L_6, L_5)$ } to be deleted if at all in E

$\cap(L_3, L_6)$
$\cap(L_4, L_5)$ } insert in E if appropriate

Initialize E (the event queue) to $P_1,\ldots,P_n,Q_1,\ldots,Q_n$ in sorted order;

Initialize S (the sweep line information) to empty;

while (E is not empty) {

  Pick up the next event (the event with smallest x-coordinate) from E;

  if it is an *enter segment* event {

    Suppose that the left end point of L_i has triggered the event;

    Insert L_i in the appropriate position of S (kept sorted);

    Let L_s be the immediate predecessor of L_i in S;

    Let L_t be the immediate successor of L_i in S;

    If $\bigcap(L_s,L_t)$ exists and lies to the right of L, delete $\bigcap(L_s,L_t)$ from E;

    If $\bigcap(L_s,L_i)$ exists and lies to the right of L, insert $\bigcap(L_s,L_i)$ in E;

    If $\bigcap(L_i,L_t)$ exists and lies to the right of L, insert $\bigcap(L_i,L_t)$ in E;

  } else if it is a *leave segment* event {

    Suppose that the right end point of L_i has triggered the event;

    Let L_s be the immediate predecessor of L_i in S;

    Let L_t be the immediate successor of L_i in S;

    Delete L_i from S;

    If $\bigcap(L_s,L_t)$ exists and lies to the right of L, insert $\bigcap(L_s,L_t)$ in E;

  } else if it is an *intersection point* event {

    Suppose that $\bigcap(L_i,L_j)$ has triggered the event;

    Print $\bigcap(L_i,L_j)$;

    Let L_s be the immediate predecessor of L_i in S;

    Let L_t be the immediate successor of L_j in S;

    Interchange L_i and L_j in S;

    If $\bigcap(L_s,L_i)$ exists and lies to the right of L, delete $\bigcap(L_s,L_i)$ from E;

    If $\bigcap(L_t,L_j)$ exists and lies to the right of L, delete $\bigcap(L_t,L_j)$ from E;

    If $\bigcap(L_s,L_j)$ exists and lies to the right of L, insert $\bigcap(L_s,L_j)$ in E;

    If $\bigcap(L_i,L_t)$ exists and lies to the right of L, insert $\bigcap(L_i,L_t)$ in E;

  }

}

S - arbitrary insert
arbitrary deletion
swapping two consecutive lines

E - Deletemin
Arbitrary insert
Arbitrary delete

S and E are implemented as height-balanced binary search trees (AVL / RB trees)

$$|S| \leq n$$

$$|E| \leq 2n + (n-1) = 3n-1$$

$$O(n) \quad \text{space}$$

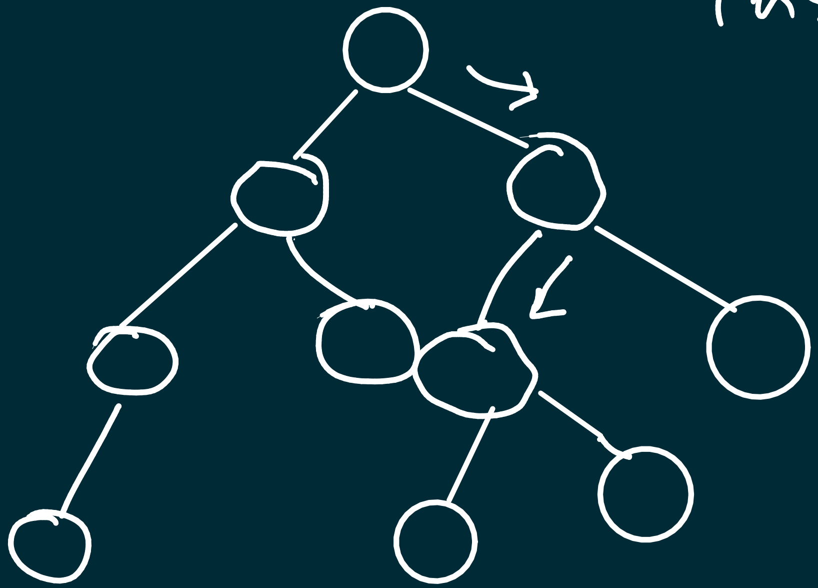
$$O\left((2n + 2n + h)\log n\right) \quad \text{time}$$

$$= O((n+h)\log n)$$

How to insert in absence of stored y-coordinates.

insert P_i / delete Q_j

S



O(log n) overhead

Compute only the y-coordinates compared in the tree

E stores x-coordinates only. All comparisons are based on x-coordinates only.