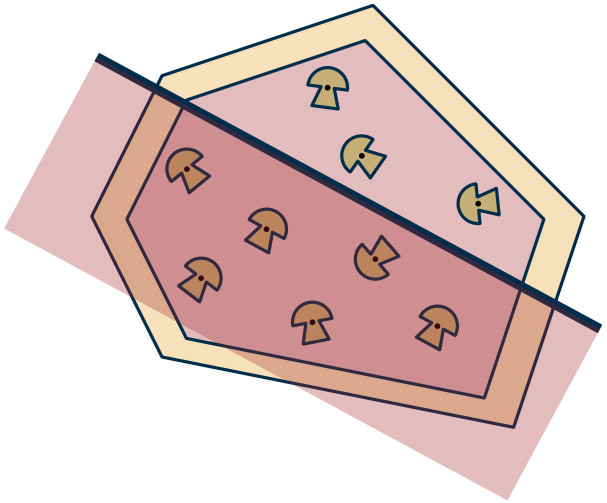
Computational Geometry

Exercise 4 *Assignment 3, 4 and Voronoi diagrams of segments*

Jean-Pierre, Marcus, Wendy

Assignment 3

Pizza-Guillotine

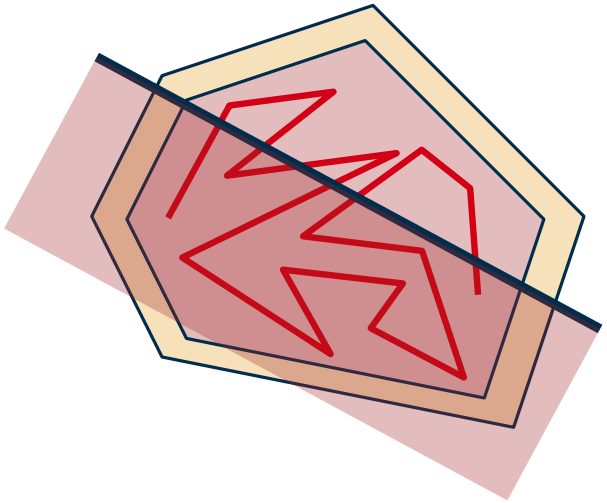


- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?

- $\mathcal{O}(n \log n)$ precompute, $\mathcal{O}(\log n)$ query

Assignment 3

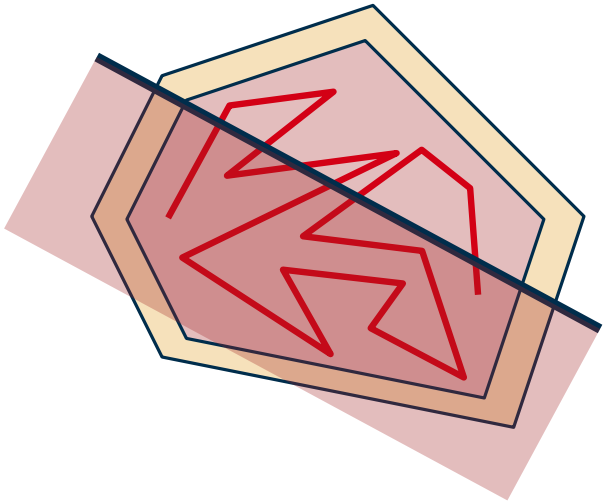
Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

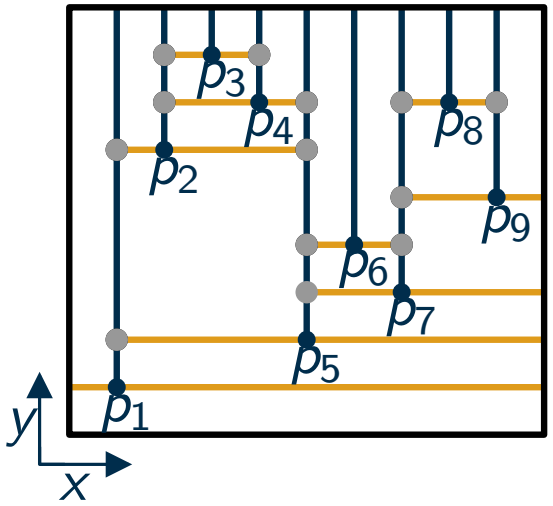
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

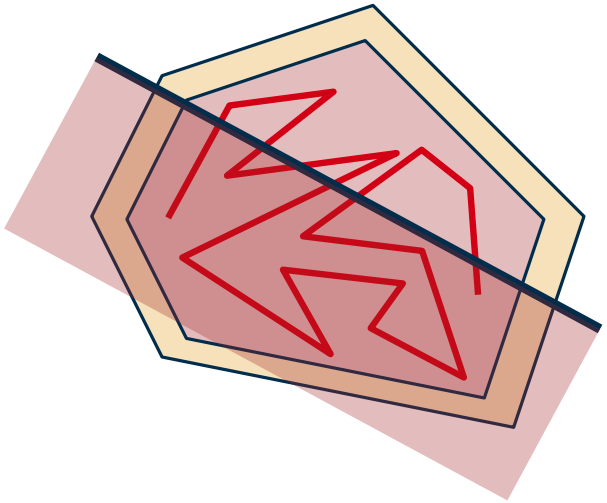
2*d* Range Queries



Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

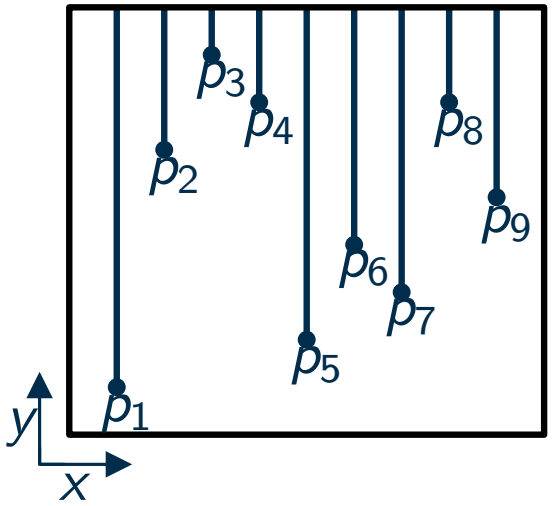
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

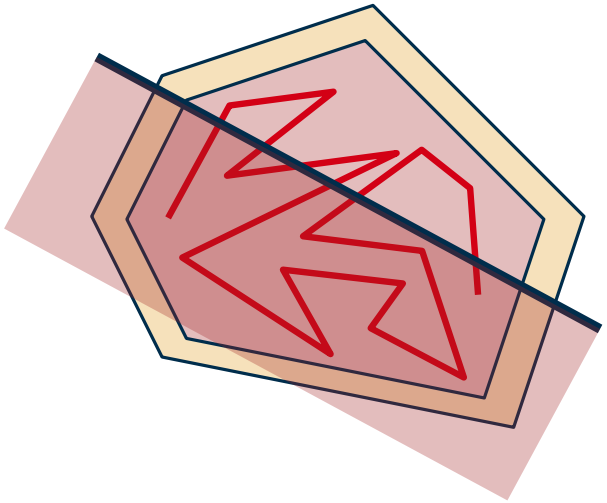


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right

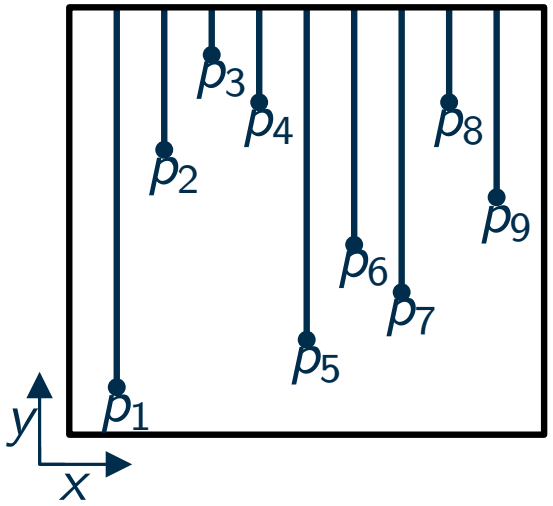
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

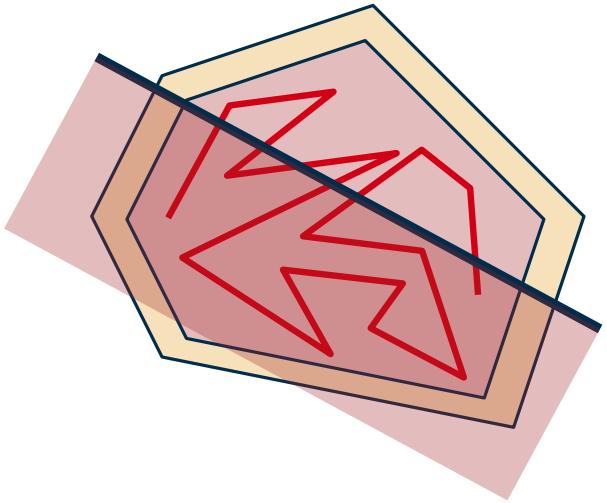


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done

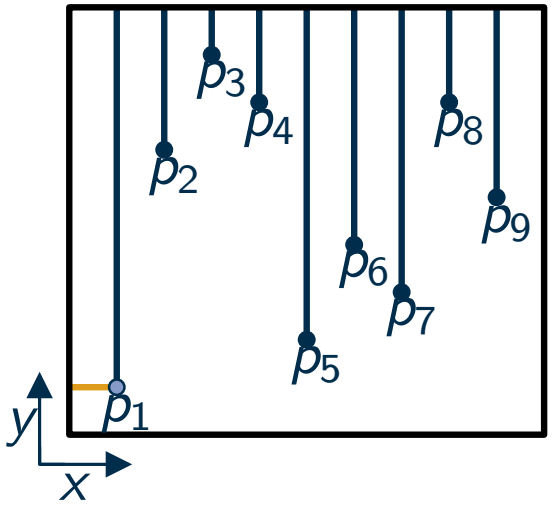
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

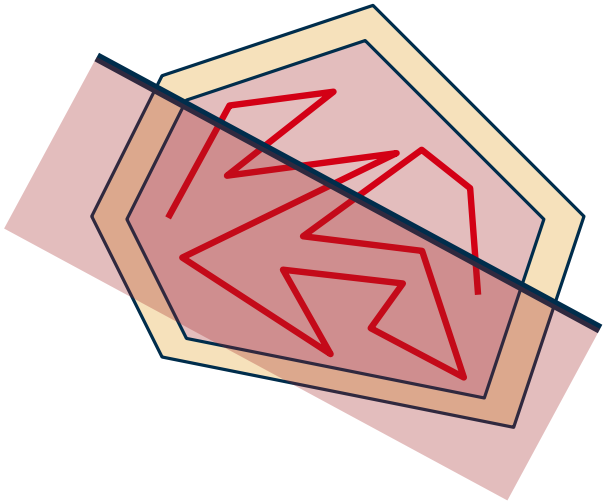


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

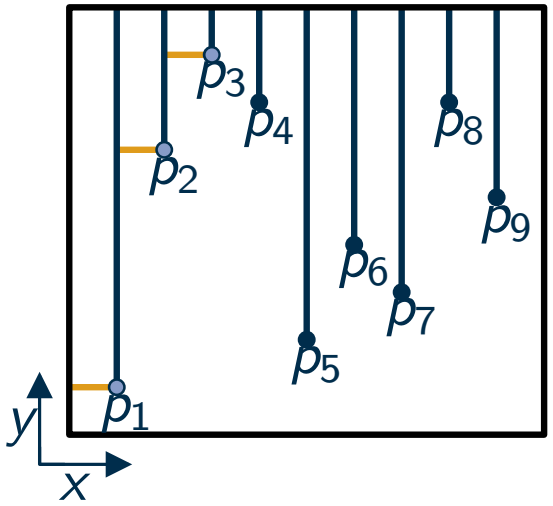
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

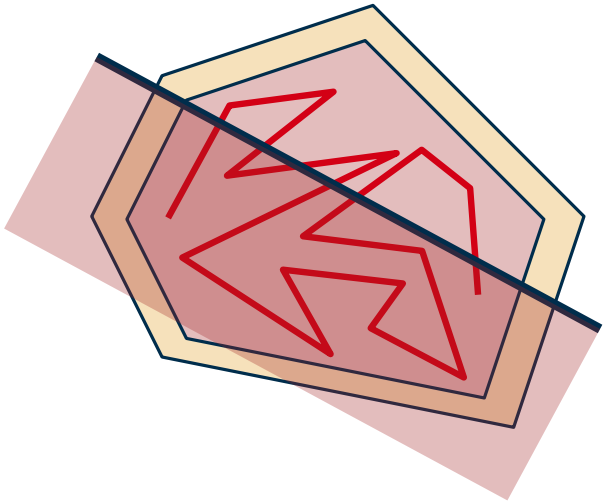


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

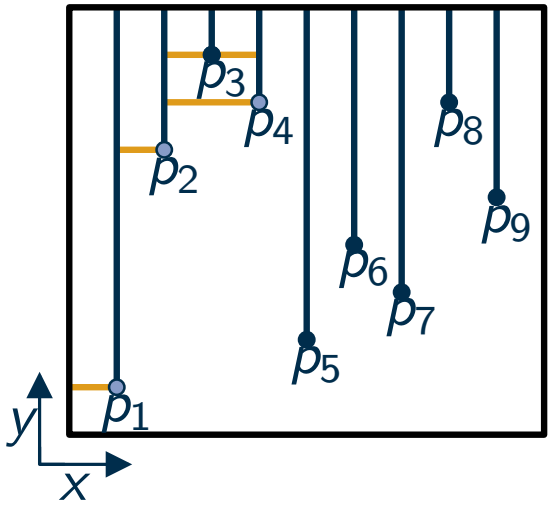
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

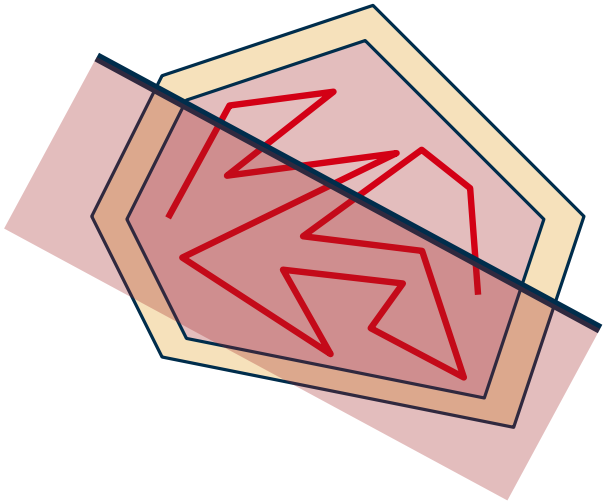


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

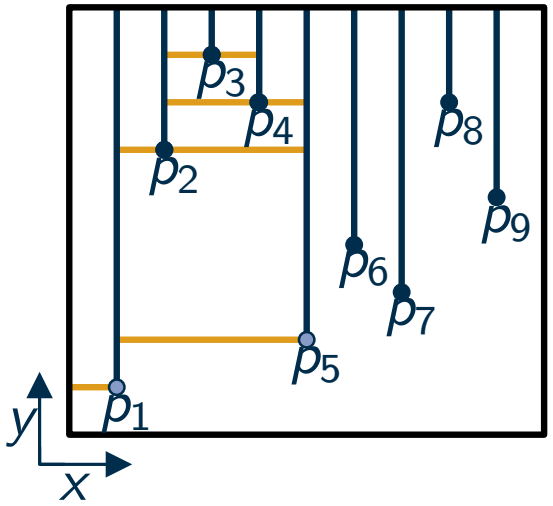
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

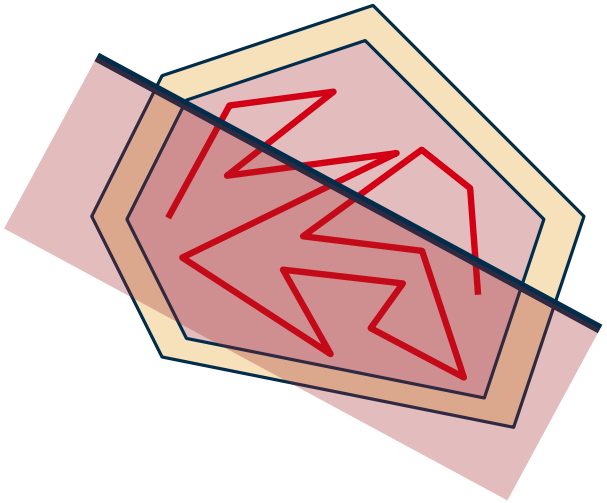


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

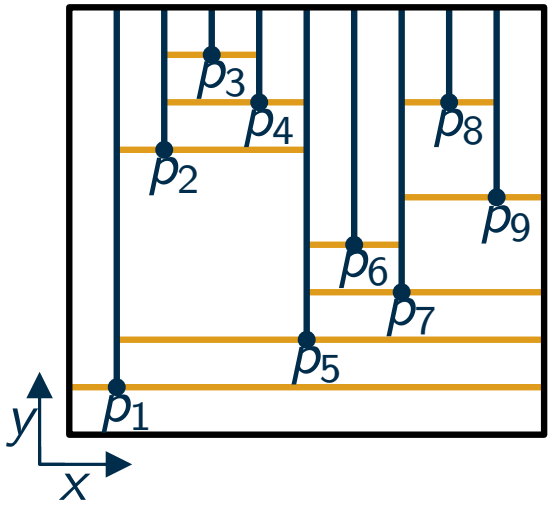
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries

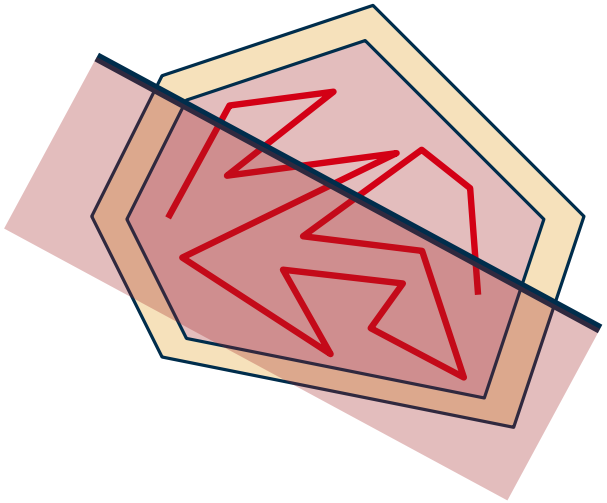


Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

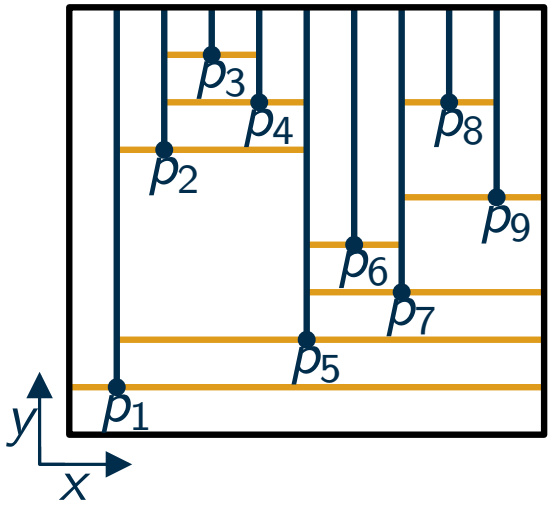
Assignment 3

Pizza-Guillotine



- At which points does the blade cut the pizza polygon?
- Which ingredients are cooked?
- how many times is the tomato sauce hit?
- $\mathcal{O}(n \log n)$ precompute,
  $\mathcal{O}((k+1)\log(n/(k+1)))$ query

2*d* Range Queries



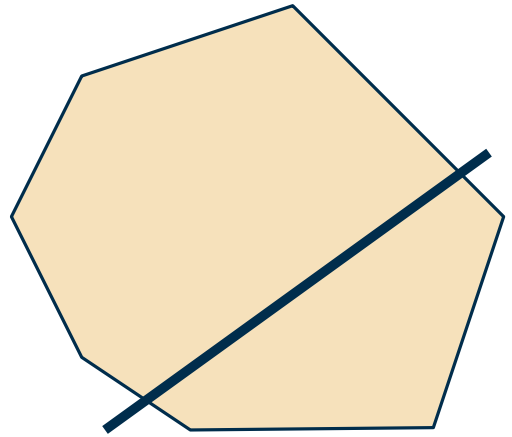
Given the points sorted by x, calculate the geometric graph in $\mathcal{O}(n)$

- process points from left to right
- use stack to store points that are not done
- add horizontal lines

Invariant? Correctness? Runtime?

Pizza Guillotine I

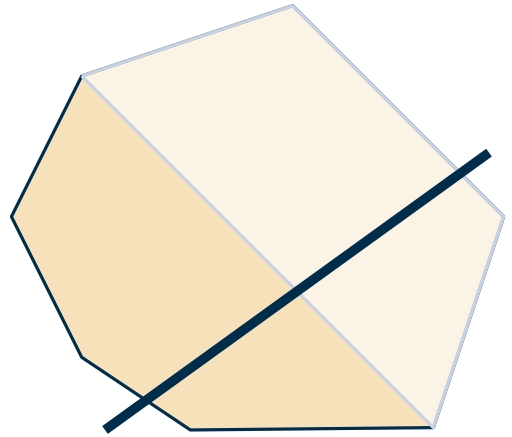
a) Find intersection of line with convex polygon



Pizza Guillotine I

a) Find intersection of line with convex polygon

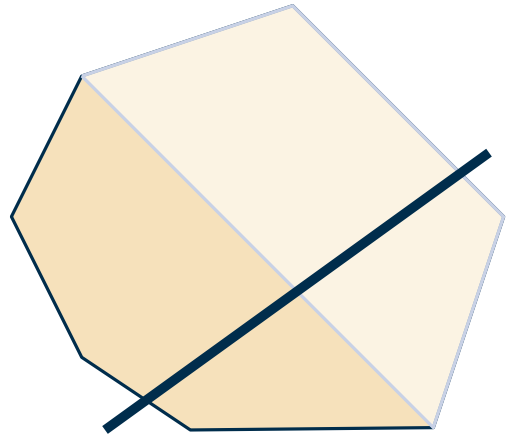
Step 1: split polygon such that each part contains one intersection



Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

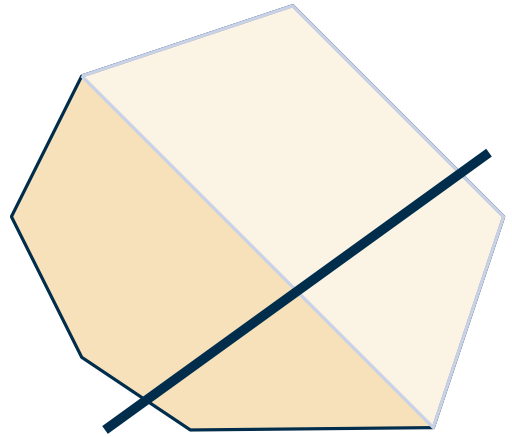


Step 2: find intersection for each part

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



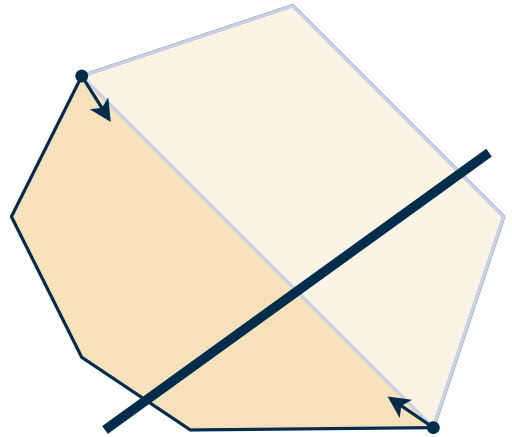
Step 2: find intersection for each part

- use binary search over vertices

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



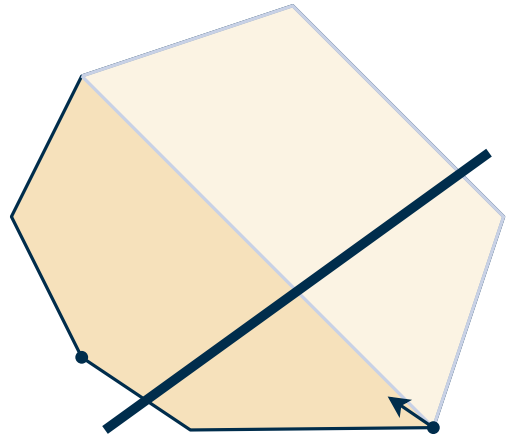
Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



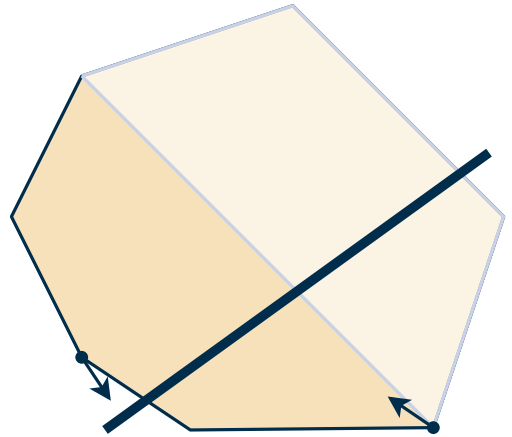
Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



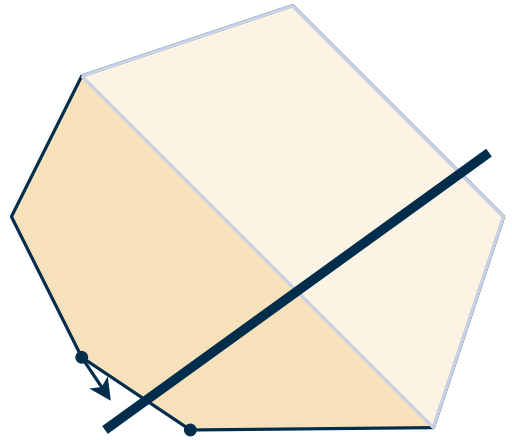
Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



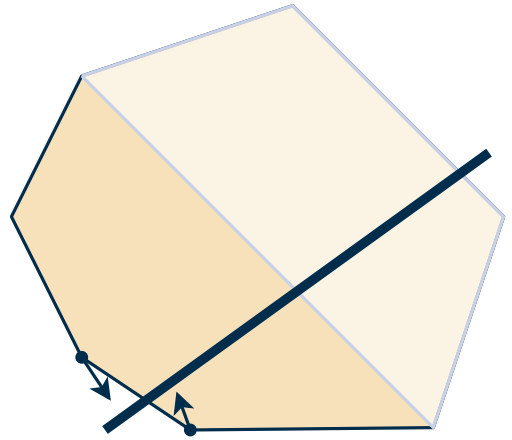
Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection



Step 2: find intersection for each part

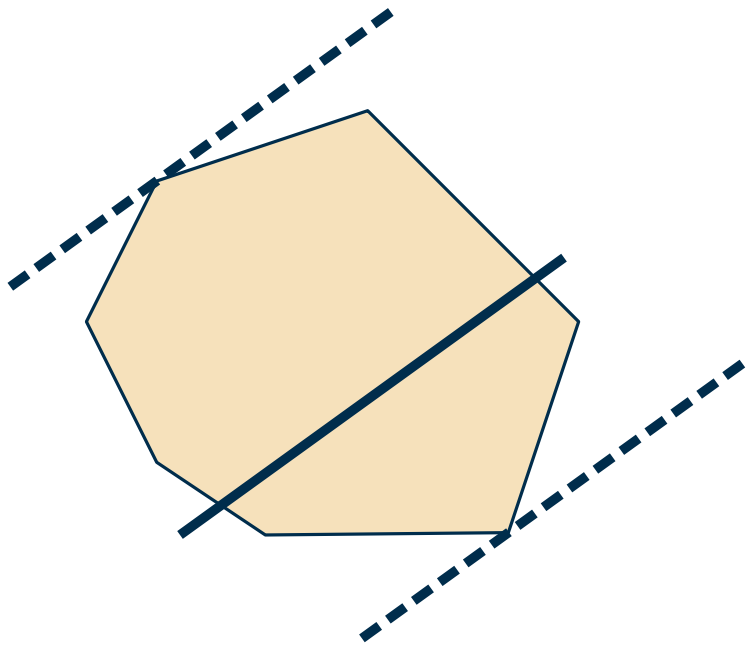
- use binary search over vertices
- monotone property: above/below line

Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

FIND TANGENTS PARALLEL TO LINE



Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

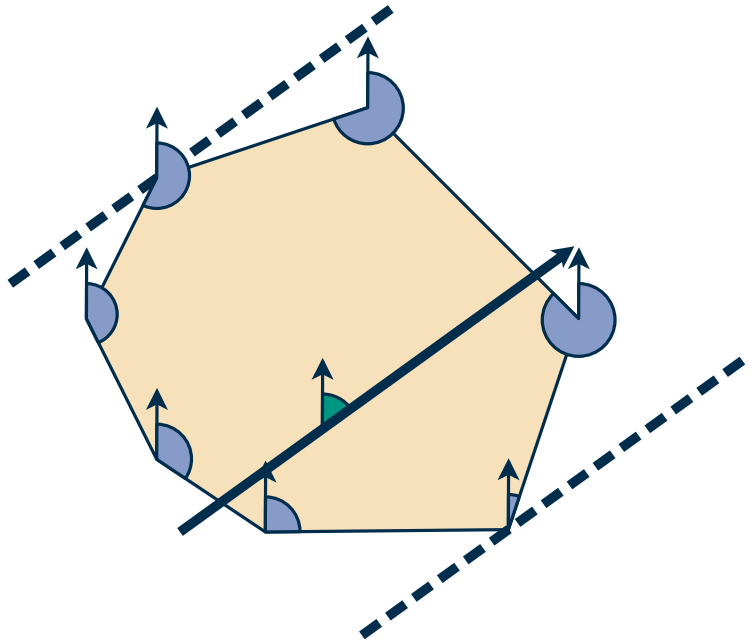
Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

FIND TANGENTS PARALLEL TO LINE

- find edge with closest y-axis-angle to oriented line



Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

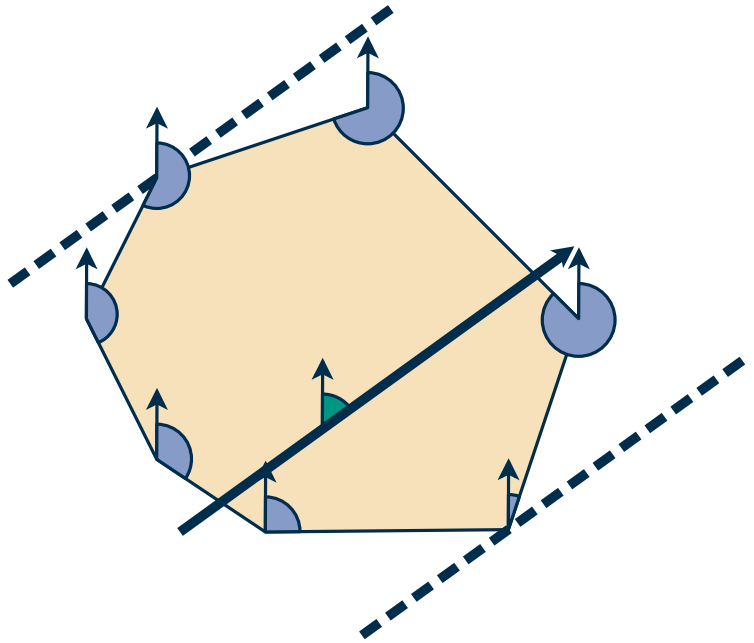
Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

FIND TANGENTS PARALLEL TO LINE

- find edge with closest y-axis-angle to oriented line
- binary search, monotone property: y-axis-angle sorted along polygon



Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

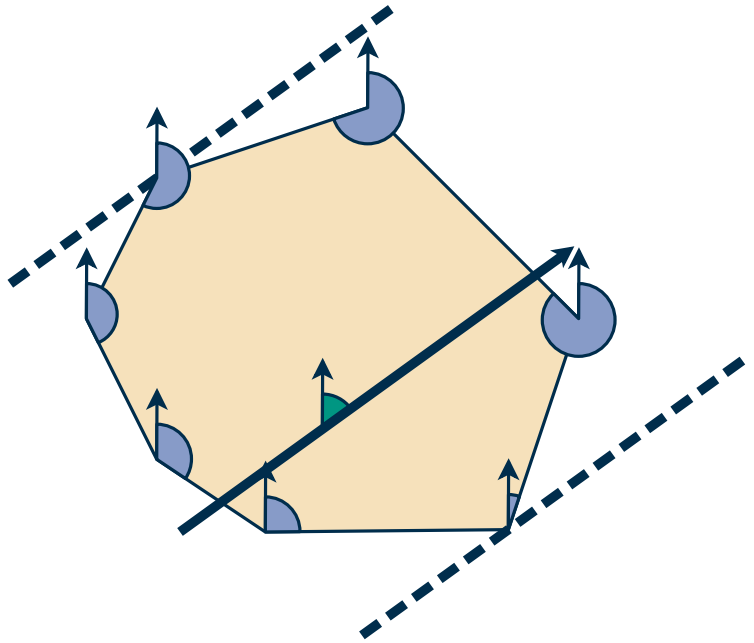
Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

FIND TANGENTS PARALLEL TO LINE

- find edge with closest y-axis-angle to oriented line
- binary search, monotone property: y-axis-angle sorted along polygon
- try both orientations



Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

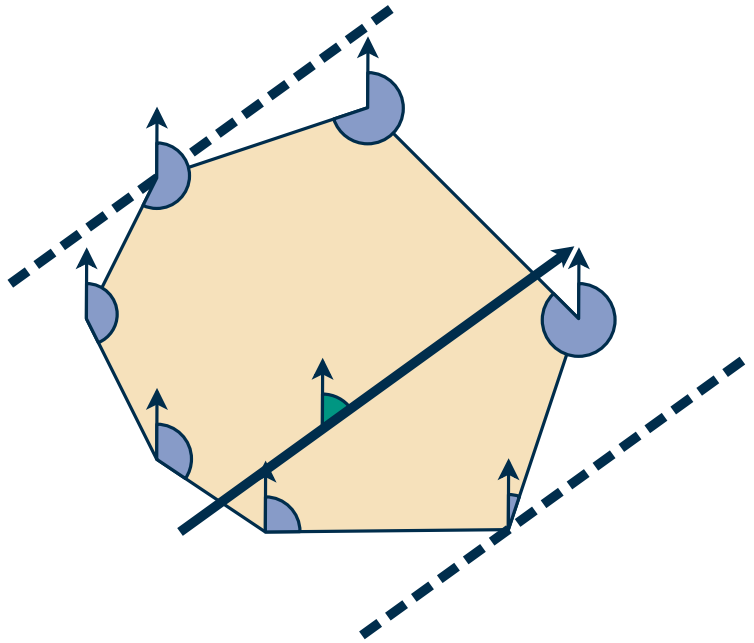
Pizza Guillotine I

a) Find intersection of line with convex polygon

Step 1: split polygon such that each part contains one intersection

FIND TANGENTS PARALLEL TO LINE

- find edge with closest y-axis-angle to oriented line
- binary search, monotone property: y-axis-angle sorted along polygon
- try both orientations



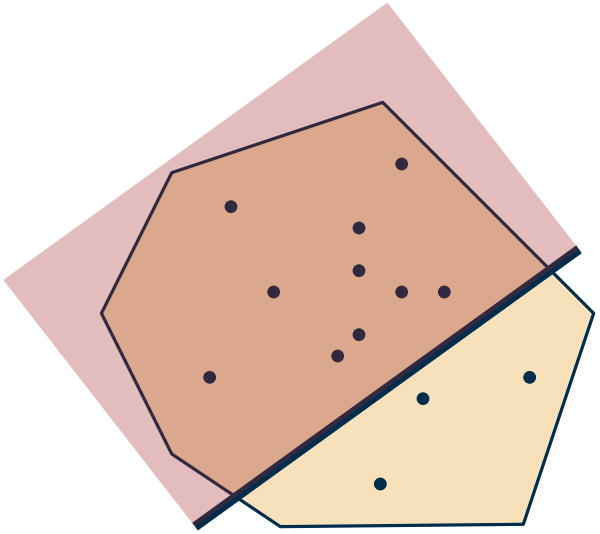
Step 2: find intersection for each part

- use binary search over vertices
- monotone property: above/below line

Edge cases? Runtime?

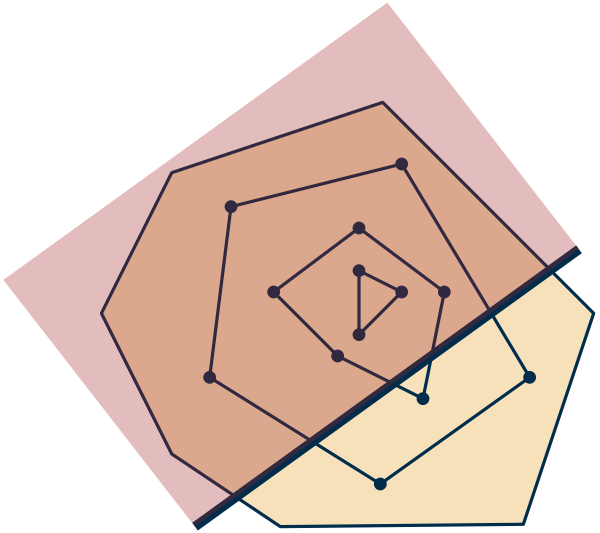
Pizza Guillotine II

b) Output all points in polygon of one side of line



Pizza Guillotine II

b) Output all points in polygon of one side of line

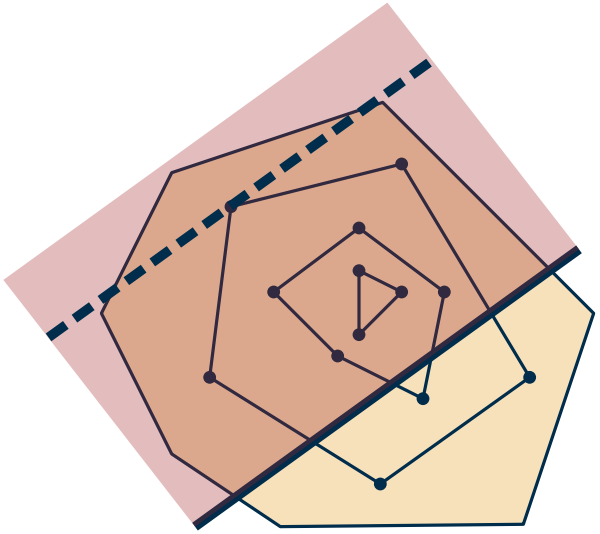


Pizza Guillotine II

b) Output all points in polygon of one side of line

Outermost convex hull

- FIND TANGENTS PARALLEL TO LINE on correct side

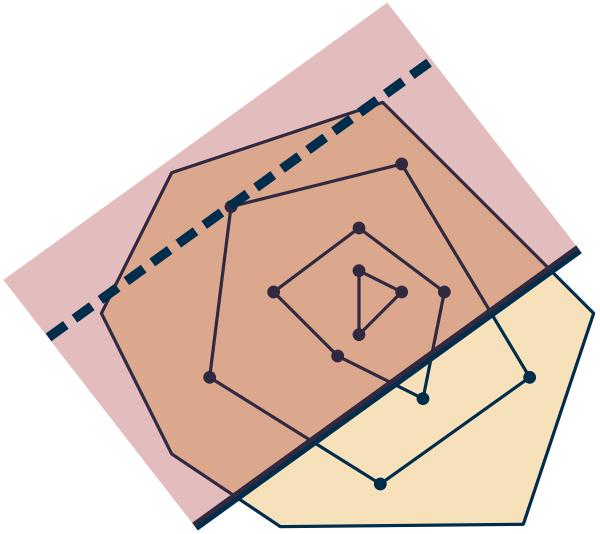


Pizza Guillotine II

b) Output all points in polygon of one side of line

Outermost convex hull

- FIND TANGENTS PARALLEL TO LINE on correct side
- linearly output all points on correct side



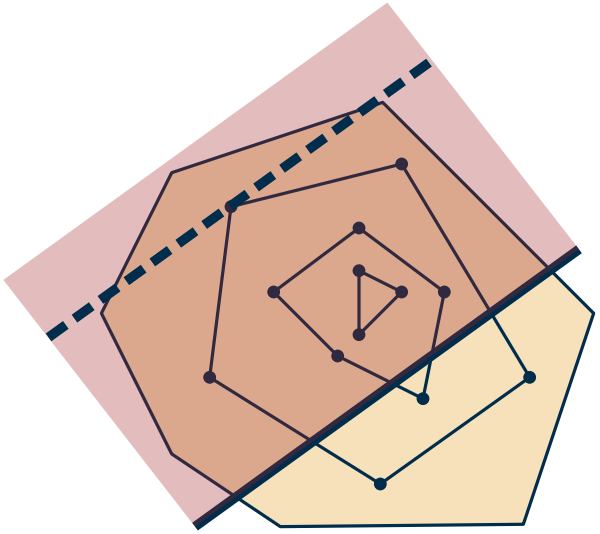
Pizza Guillotine II

b) Output all points in polygon of one side of line

Outermost convex hull

- FIND TANGENTS PARALLEL TO LINE on correct side
- linearly output all points on correct side

Next convex hull



Pizza Guillotine II

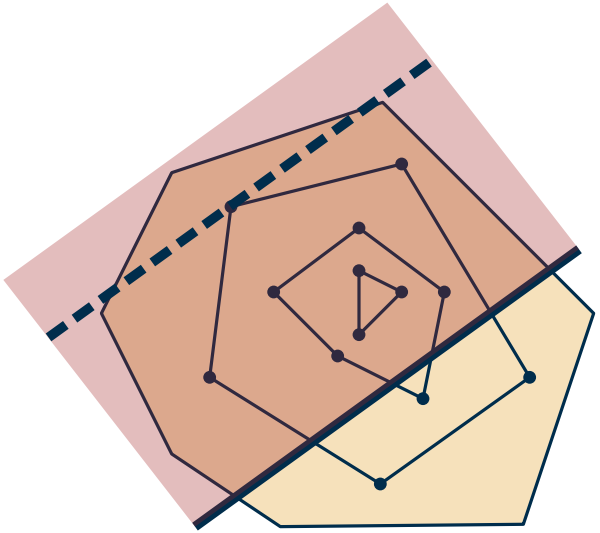
b) Output all points in polygon of one side of line

Outermost convex hull

- FIND TANGENTS PARALLEL TO LINE on correct side
- linearly output all points on correct side

Next convex hull

fractional cascading magic 🥳

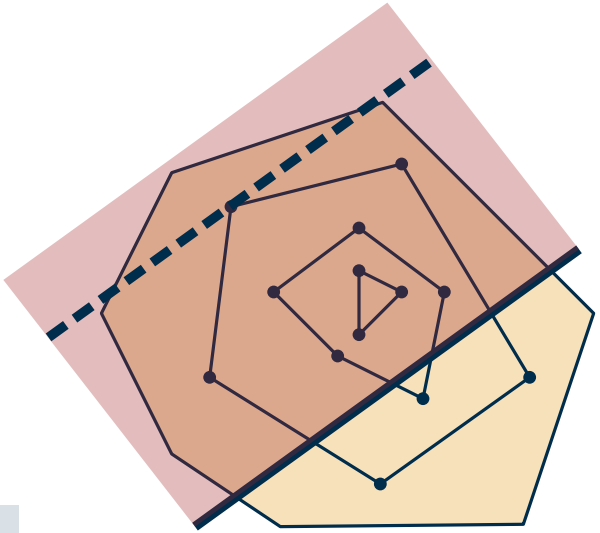


Pizza Guillotine II

b) Output all points in polygon of one side of line

Outermost convex hull

- FIND TANGENTS PARALLEL TO LINE on correct side
- linearly output all points on correct side



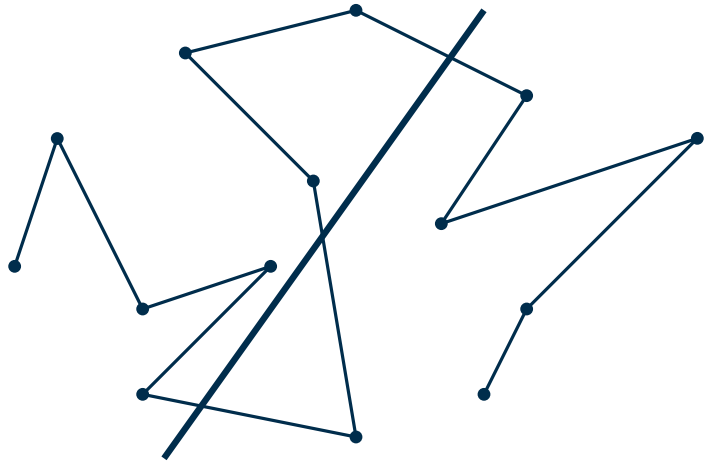
Next convex hull

fractional cascading magic 🥳

Why does this work? How to preprocess? Runtime?

Pizza Guillotine III

c) Output all intersections of line with a polygon chain

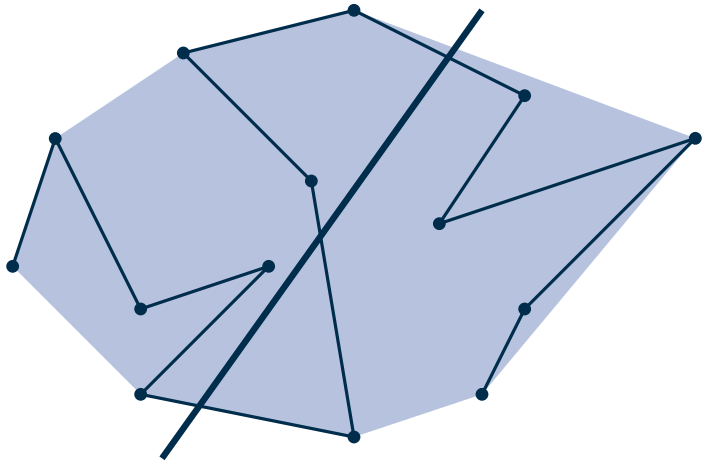


Pizza Guillotine III

c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts

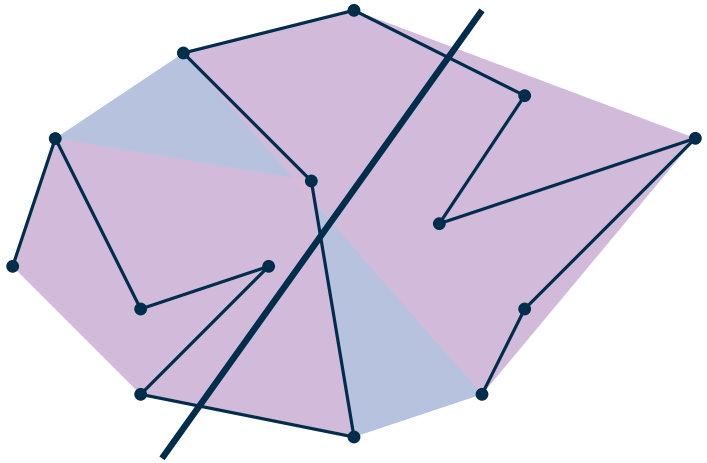


Pizza Guillotine III

c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts

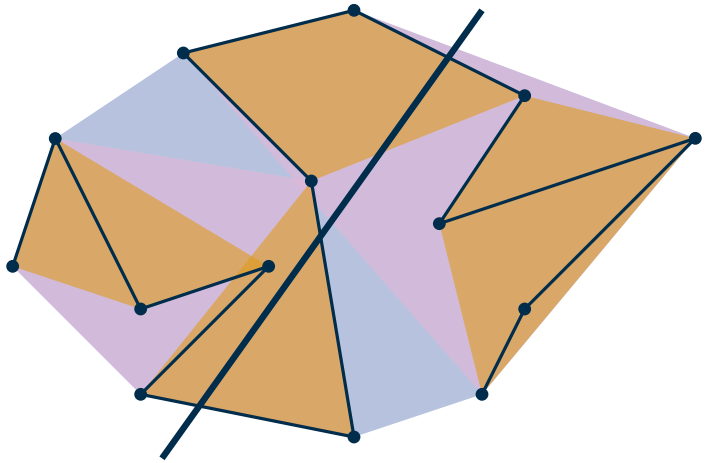


Pizza Guillotine III

c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts

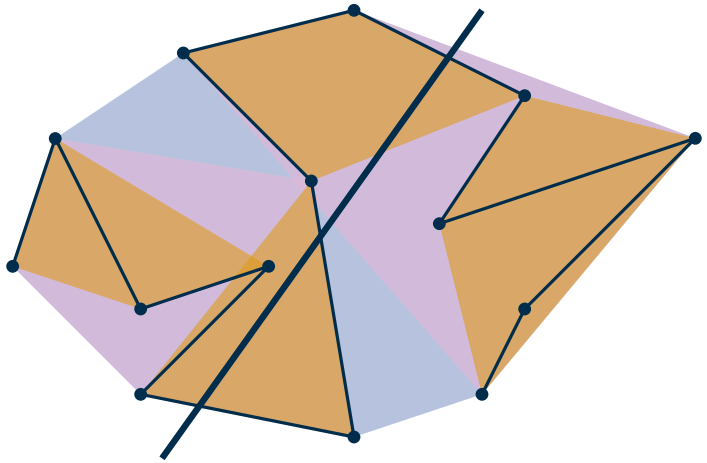


Pizza Guillotine III

c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts
- build binary tree from convex hulls



Pizza Guillotine III

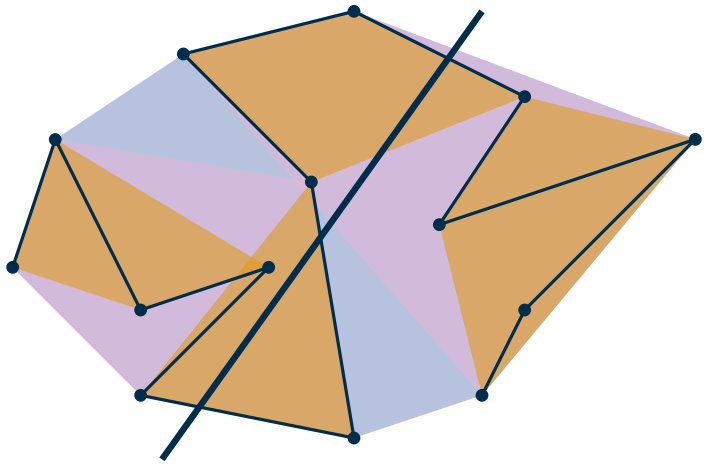
c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts
- build binary tree from convex hulls

Query

- traverse nodes in binary tree that are intersected



Pizza Guillotine III

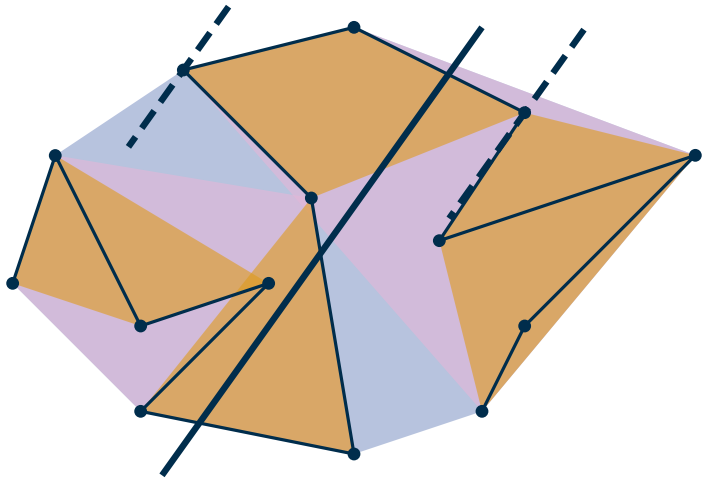
c) Output all intersections of line with a polygon chain

Preprocessing

- determine convex hulls recursively and divide path in two parts
- build binary tree from convex hulls

Query

- traverse nodes in binary tree that are intersected
- intersected if line between parallel tangents (use FIND TANGENTS PARALLEL TO LINE)



Pizza Guillotine III

c) Output all intersections of line with a polygon chain

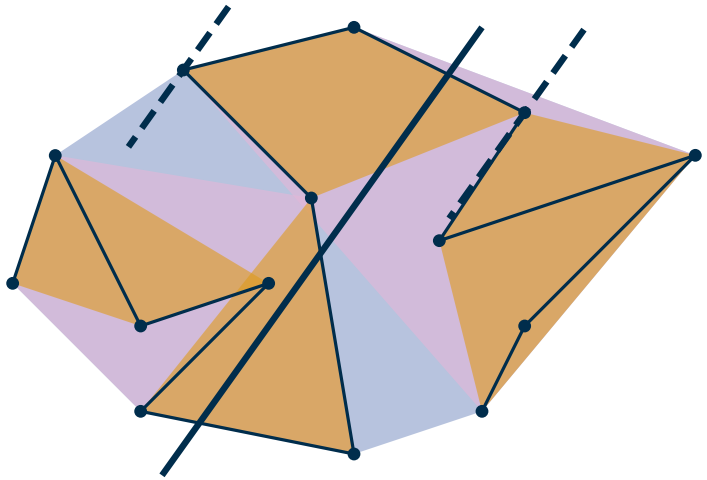
Preprocessing

- determine convex hulls recursively and divide path in two parts
- build binary tree from convex hulls

Query

- traverse nodes in binary tree that are intersected
- intersected if line between parallel tangents (use FIND TANGENTS PARALLEL TO LINE)

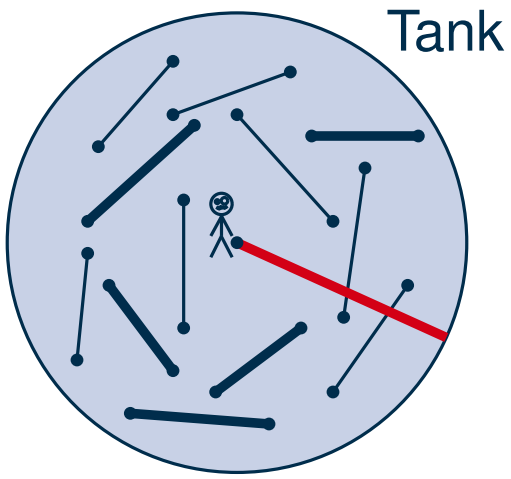
more fractional cascading magic 🥳



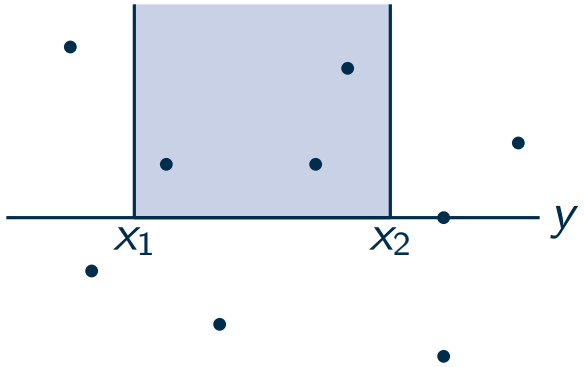
Runtime?

Assignment 4

Mission Impossible

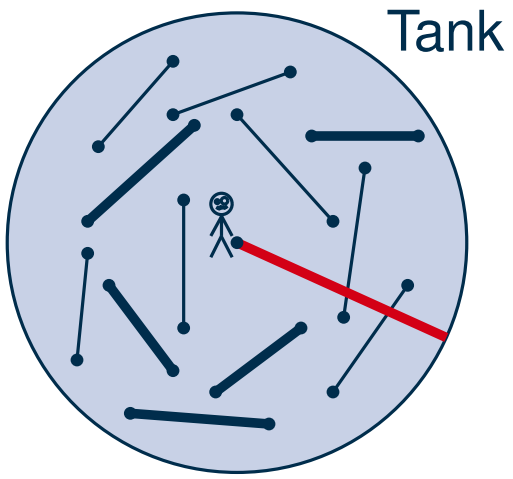


1.5 Range Queries

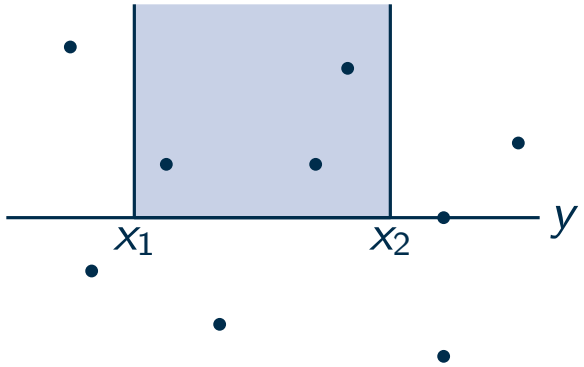


Assignment 4

Mission Impossible



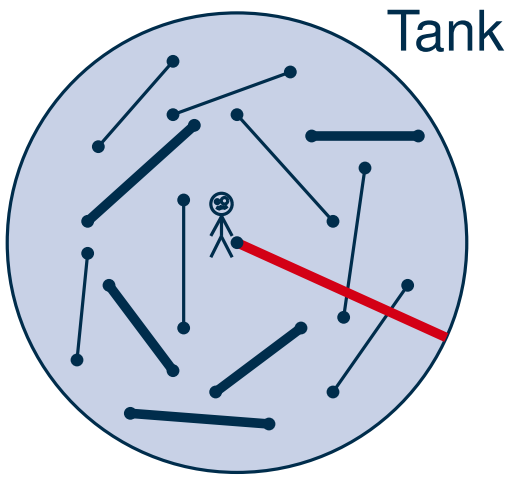
1.5 Range Queries



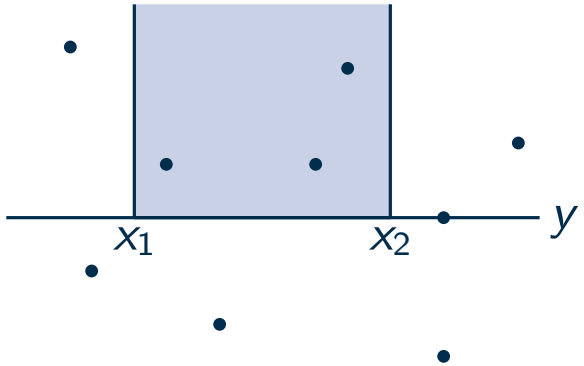
- output something in $O(\log(n) + k)$
- preprocessing time $O(n \log(n))$

Assignment 4

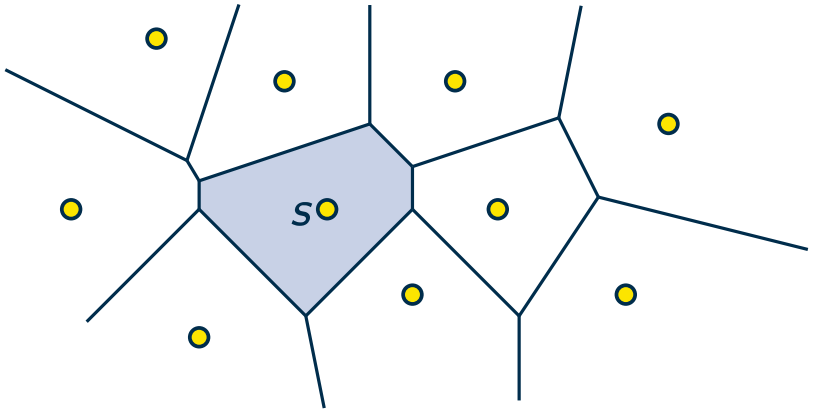
Mission Impossible



1.5 Range Queries



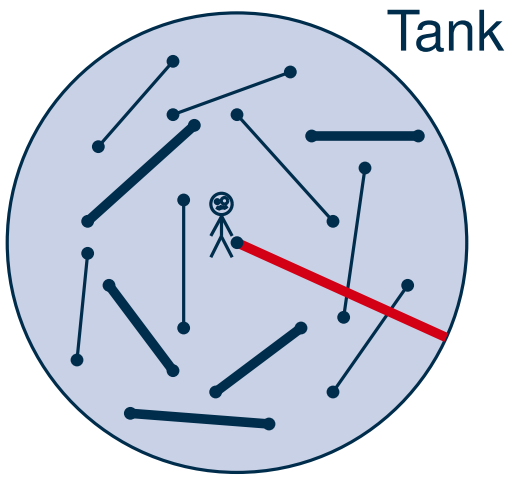
Voronoi diagram



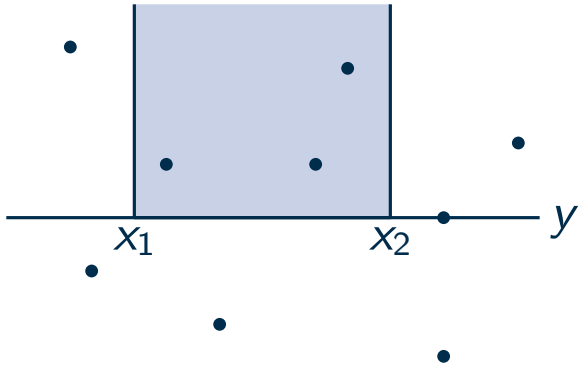
- output something in $O(\log(n) + k)$
- preprocessing time $O(n \log(n))$

Assignment 4

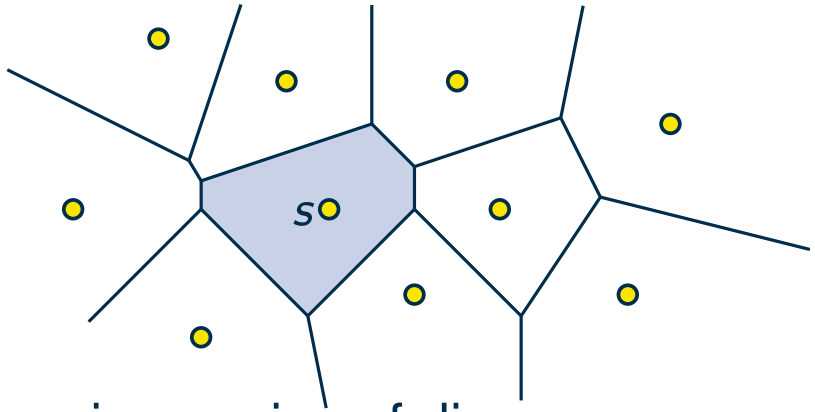
Mission Impossible



1.5 Range Queries



Voronoi diagram

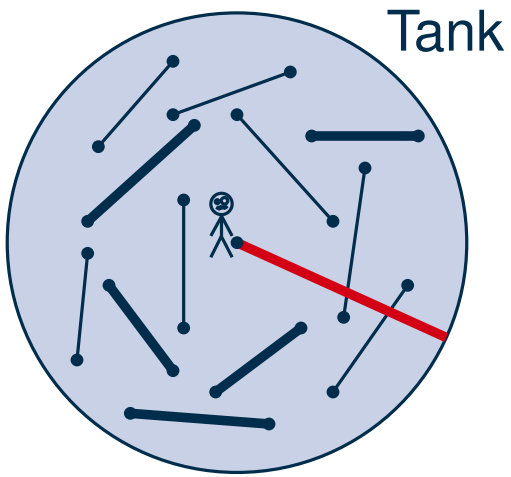


- maximum size of diagram

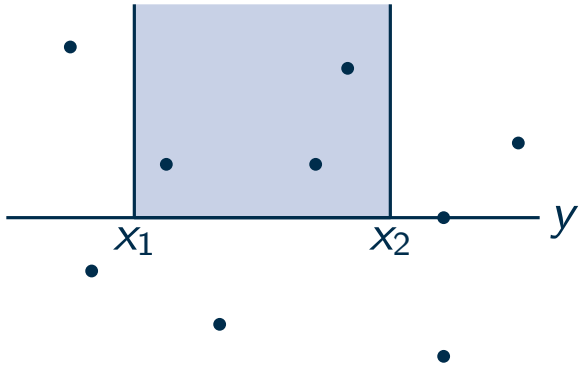
- output something in $O(\log(n) + k)$
- preprocessing time $O(n \log(n))$

Assignment 4

Mission Impossible

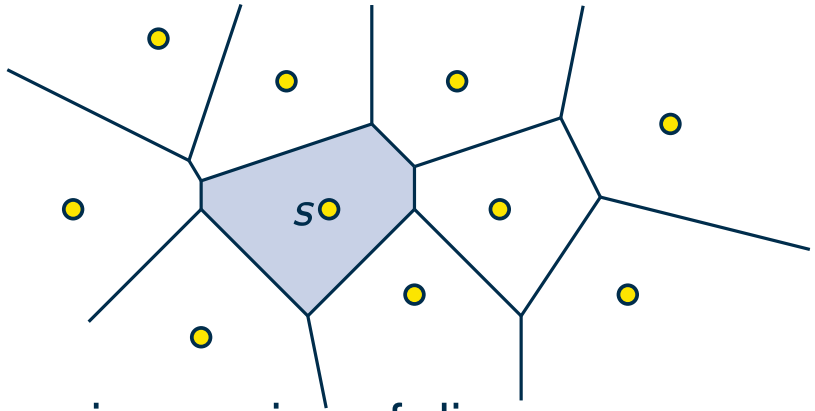


1.5 Range Queries



- output something in $O(\log(n) + k)$
- preprocessing time $O(n \log(n))$

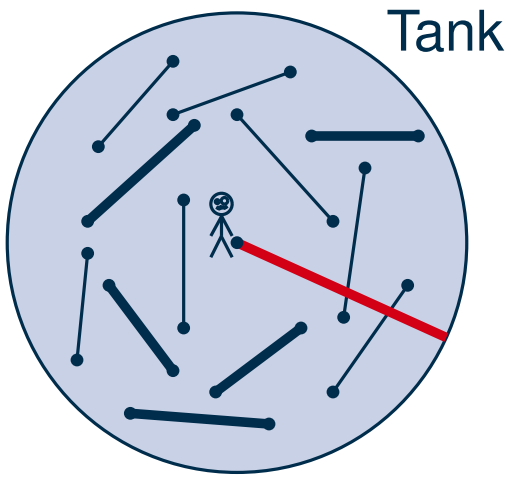
Voronoi diagram



- maximum size of diagram
- lower bound for algorithm

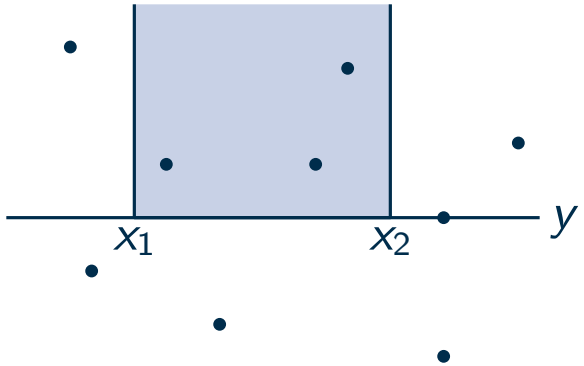
Assignment 4

Mission Impossible

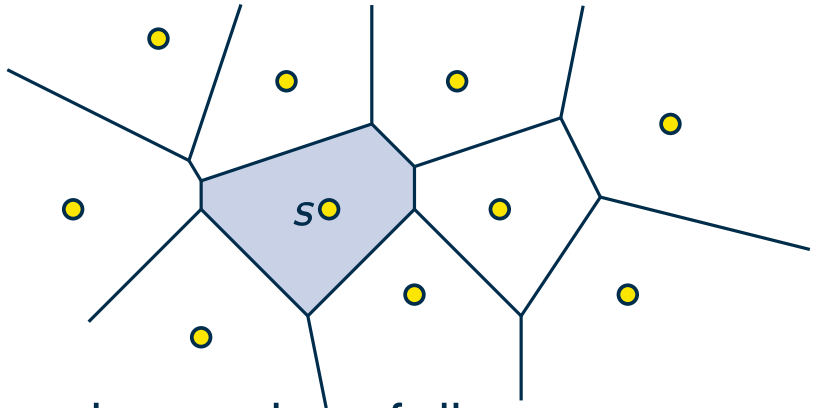


- output something in $O(\log(n) + k)$
- preprocessing time $O(n \log(n))$

1.5 Range Queries



Voronoi diagram



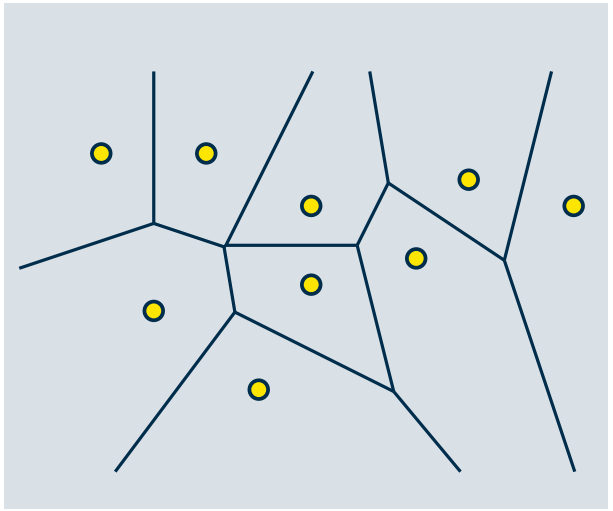
- maximum size of diagram
- lower bound for algorithm

Closest Point

- find closest point for each point in $O(n \log(n))$

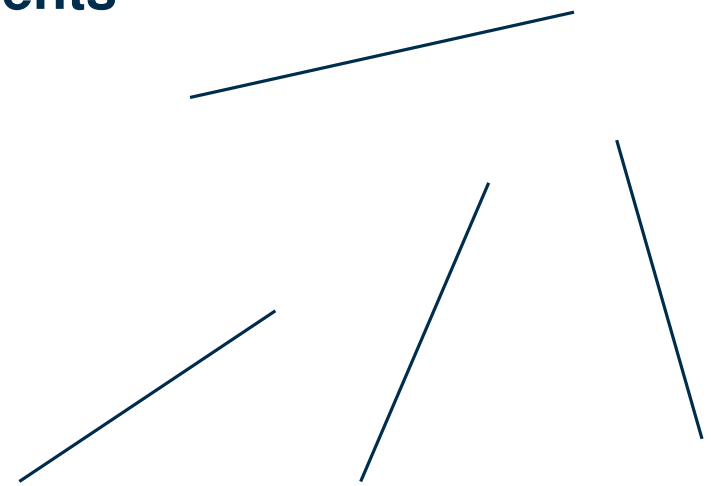
Voronoi Diagram of Segments

Points

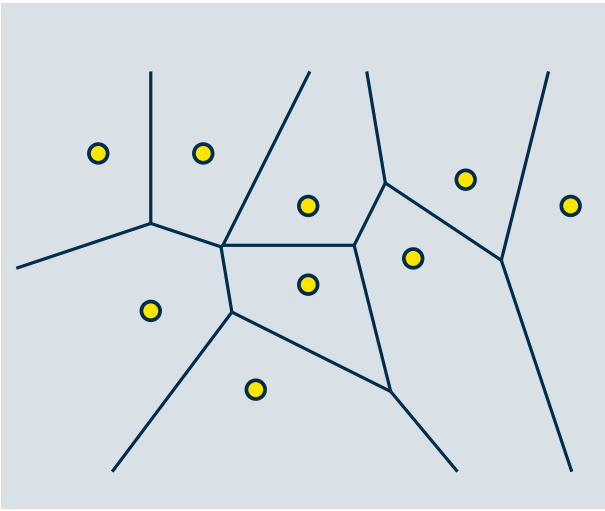


Voronoi Diagram of Segments

Segments



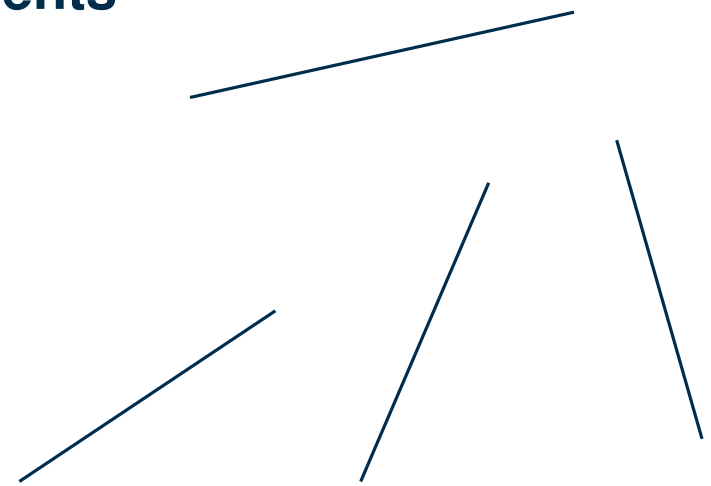
Points



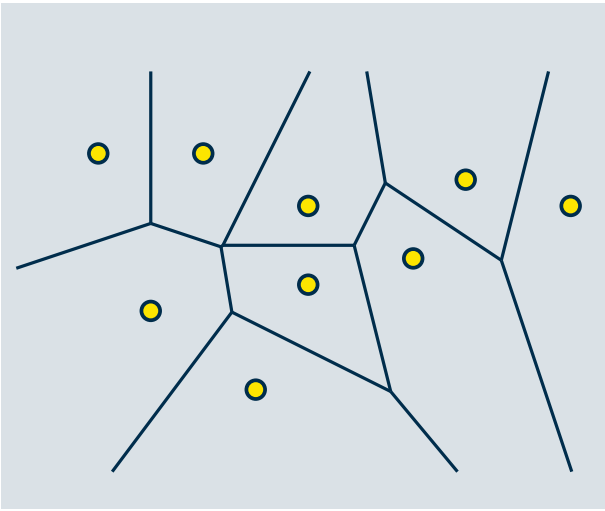
What does VD of segments even mean?

Voronoi Diagram of Segments

Segments



Points

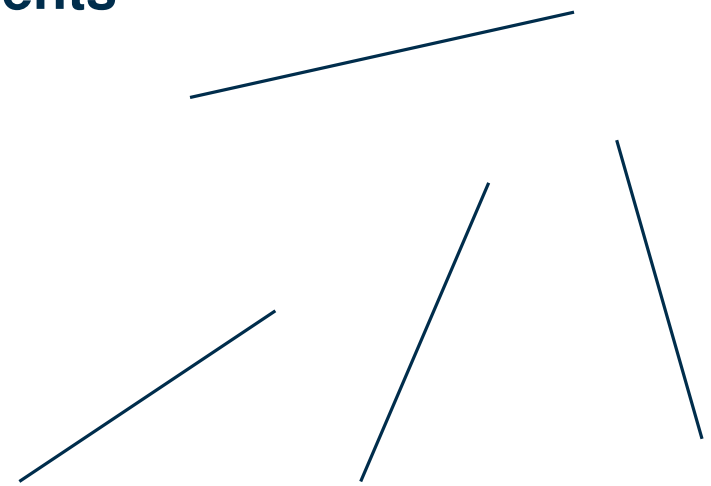


What does VD of segments even mean?

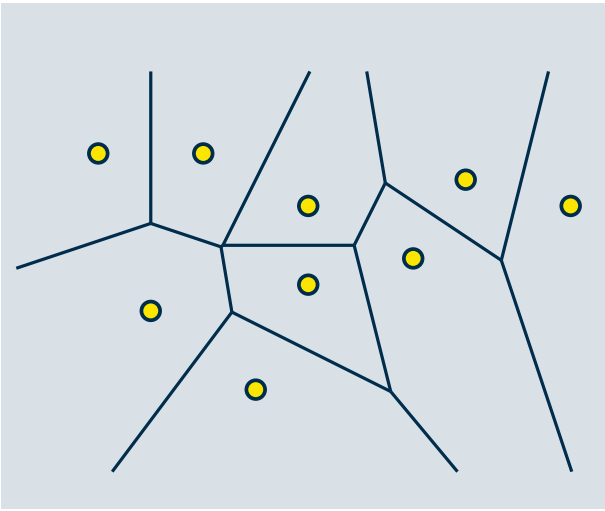
Which points are vertices of degree 3?

Voronoi Diagram of Segments

Segments



Points



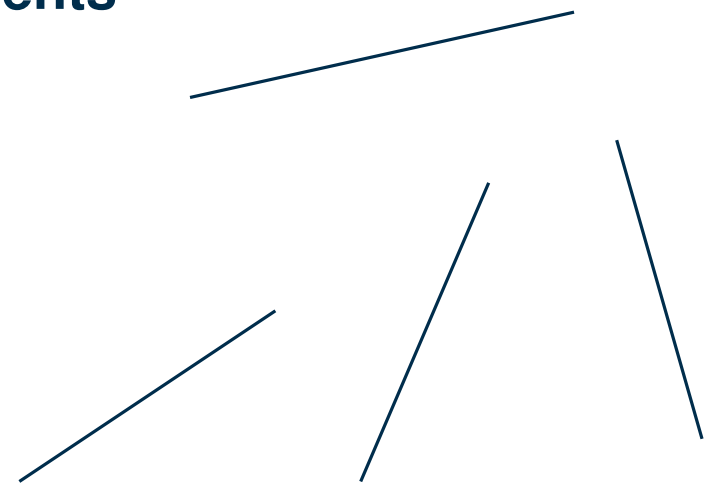
What does VD of segments even mean?

Which points are vertices of degree 3?

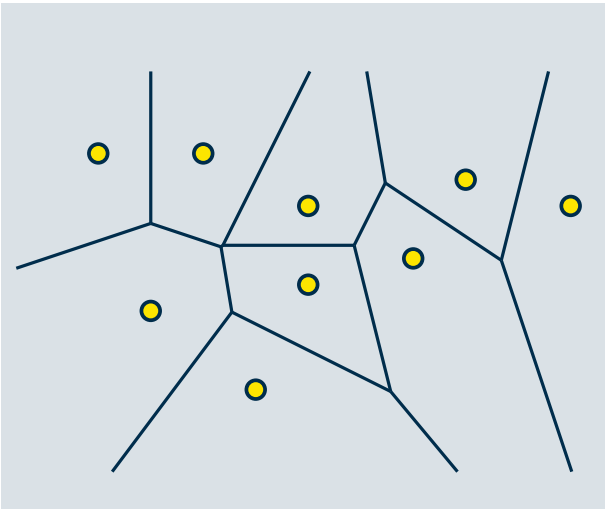
What does it look like for two segments?

Voronoi Diagram of Segments

Segments



Points



What does VD of segments even mean?

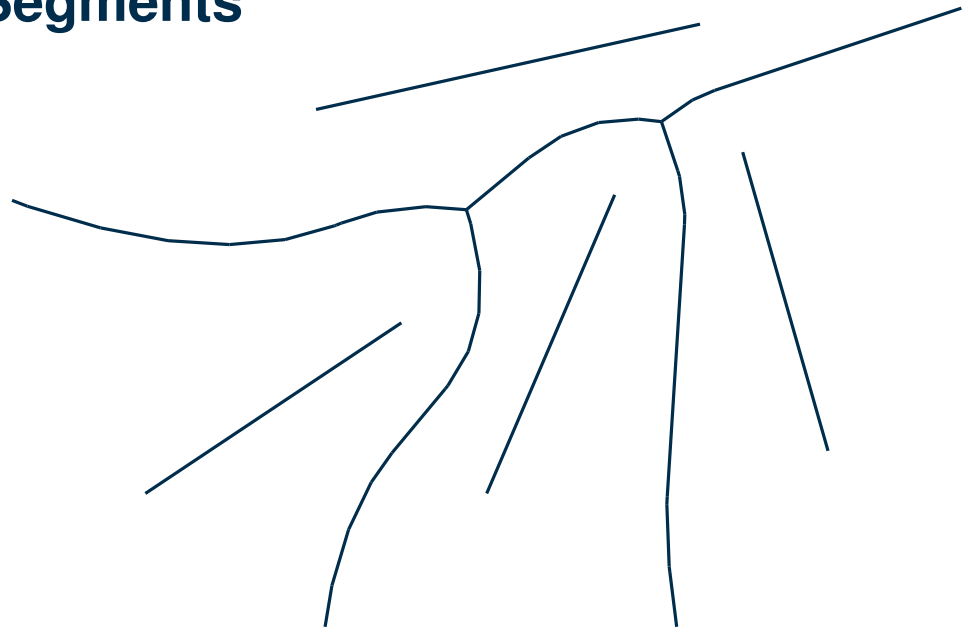
Which points are vertices of degree 3?

What does it look like for two segments?

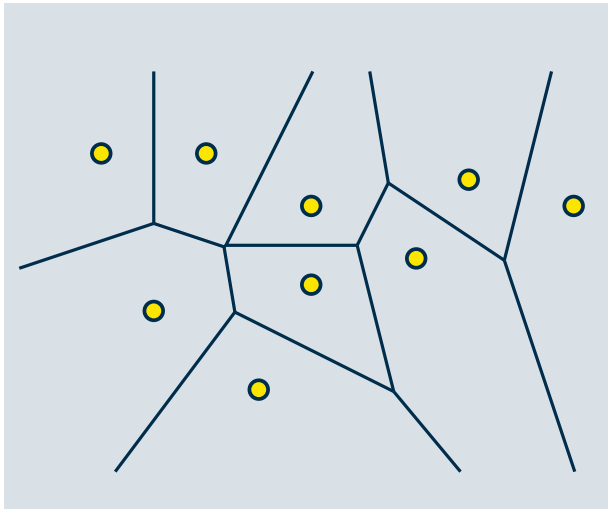
What does the beachline look like?

Voronoi Diagram of Segments

Segments



Points



What does VD of segments even mean?

Which points are vertices of degree 3?

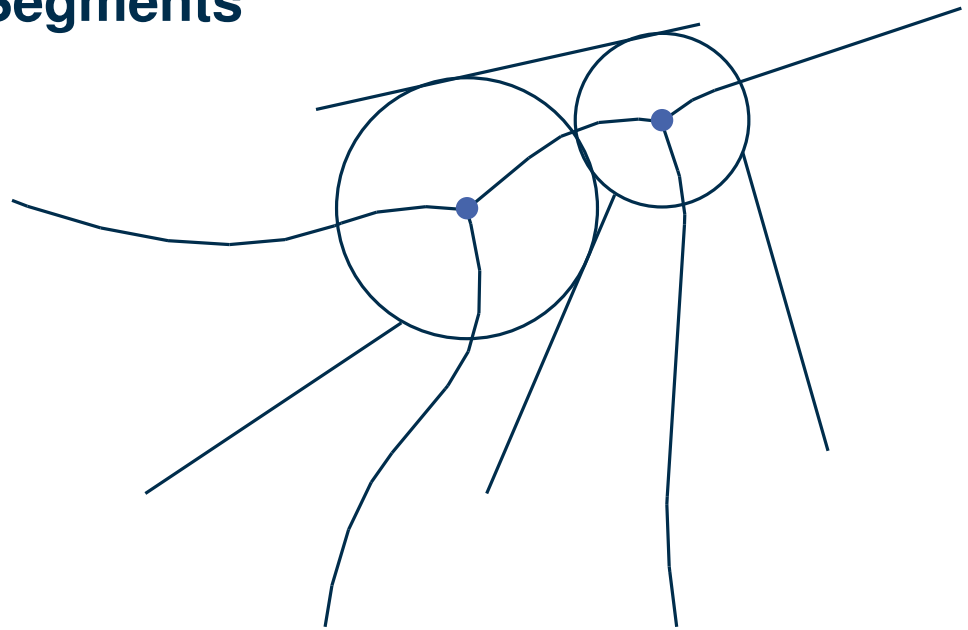
What does it look like for two segments?

What does the beachline look like?

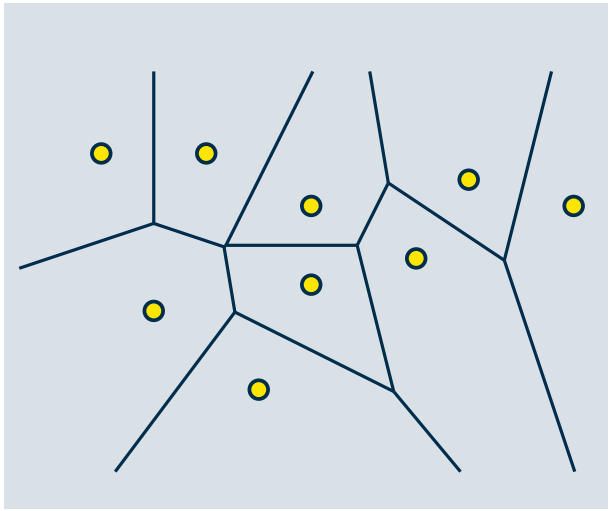
https://onlineumfrage.kit.edu/evasys/online.php?p=LCCHE

Voronoi Diagram of Segments

Segments



Points



What does VD of segments even mean?

Which points are vertices of degree 3?

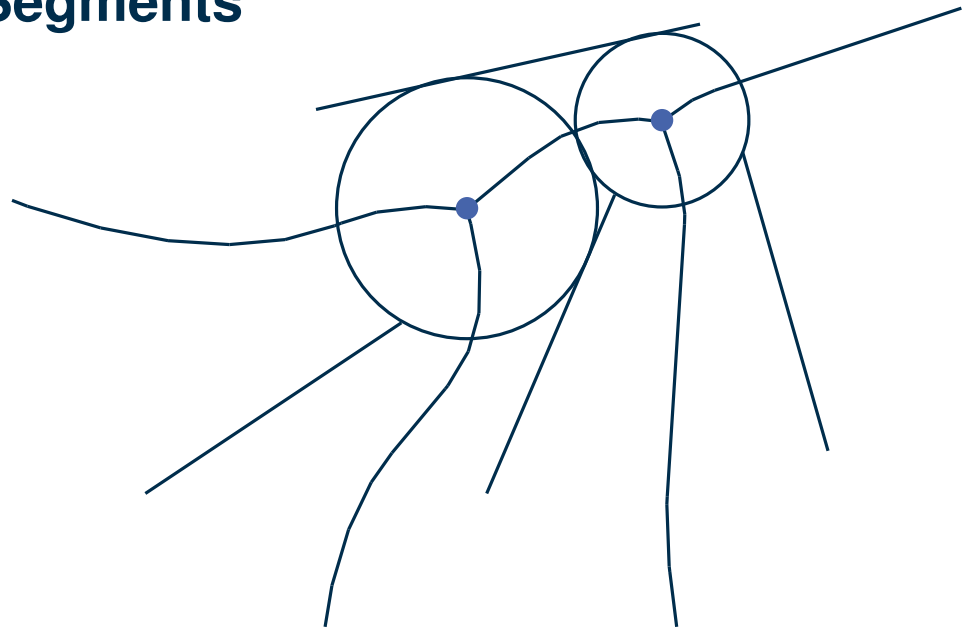
What does it look like for two segments?

What does the beachline look like?

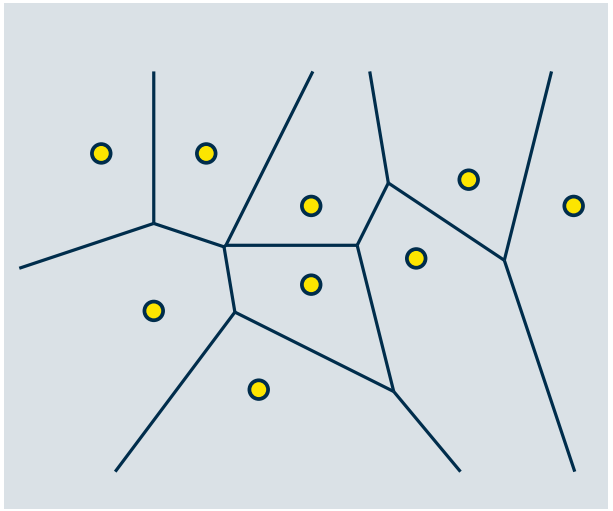
https://onlineumfrage.kit.edu/evasys/online.php?p=LCCHE

Voronoi Diagram of Segments

Segments



Points



What does VD of segments even mean?

Which points are vertices of degree 3?

What does it look like for two segments?

What does the beachline look like?

https://onlineumfrage.kit.edu/evasys/online.php?p=LCCHE