29.07.2024

Algorithms in Computational Geometry - ED5310

G-slot → mostly Mon, Thur, Fri & occassional Wednesday.

Mostly programming assignments.

Book

→ Computational Geometry in C, Joseph O' Rourke

→ Machine learning Refined: Foundations, Algorithms, & Applications - Aggelos Konstantinos Kastaggelos, Jeremy Watt & Reza Borhani

Resources

→ Art gallery theorem & algorithms, Joseph O' Rourke [online]

→ Comp. Geo: Algorithms & Applications, Mark de Berg, Otfried, Marc, Mark

→ DSA, Ulman

Library

→ CGAL [mainly c++]

→ libigl [c++]

QT → visualisation tool

→ These don't include visualisation

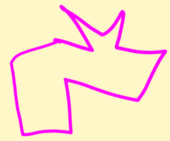
Some non-Euclidean geometry, but mostly Euclidean

Roughly 3 Assignments

Exam Pattern

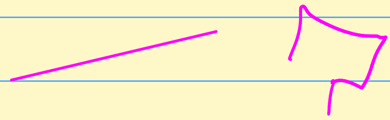
↳ Very flexible

Art Gallery Problem



Polygon: A closed loop with edges that intersect only @ vertices

Non-polygon: Overlapping lines, not closed



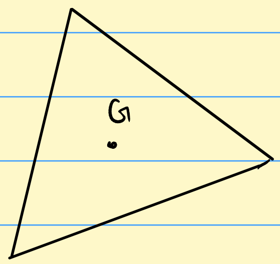
8 vertices → most complex?

31.07.2024

A convex polygon is a closed figure s.t the line segment joining any two interior point lies entirely inside the figure.

If non-adjacent sides intersect, not a polygon.

A polygon in a computer is an ordered set of vertices.



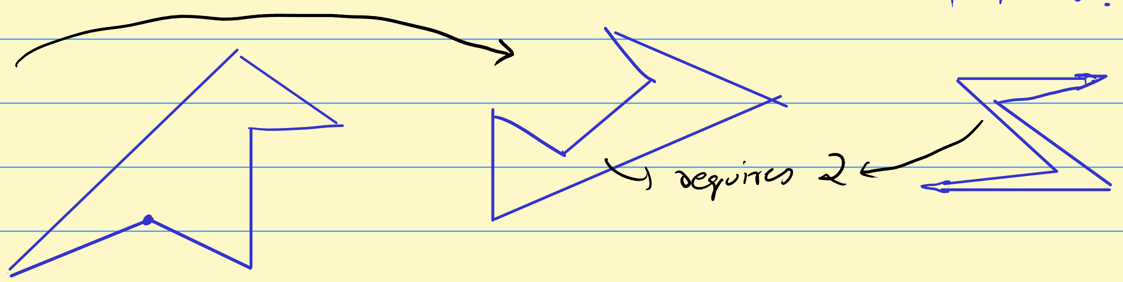
Guard can be put anywhere inside the polygon

$G(3) = 1$ → for a polygon of 3 sides, at least 1 guard is req^d.

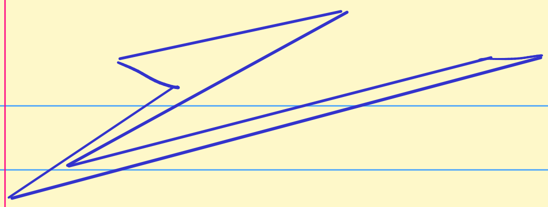
$G(4) = 1$ → just place at a non-convex edge if concave polygon

$G(5) = 2$ → How to conclusively show? Angle sum property?

$G(6) = 2$

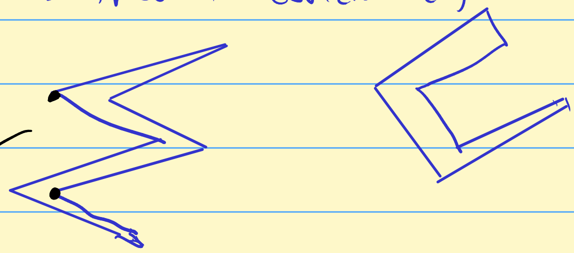


$G(7) = 2 \longrightarrow$ how to conclusively show



$G(8) = 2 \longrightarrow$ how to conclusively show

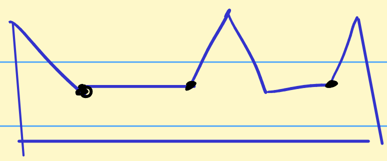
2 guards



$G(9) = 3$

The pattern is $G(n) = \left\lfloor \frac{n}{3} \right\rfloor$

for $G(n) = 3$, $n = 9$,



for $G(n) = 4$, $n = 12$

comb shape

How to prove?

We're trying to find the maximum of all minimums

$G(n)$ is maxmin

↳ We're formulating a max over min problem.

Never going to give an optimal solution as it's a max

implies maxmin

sometimes necessary, always sufficient

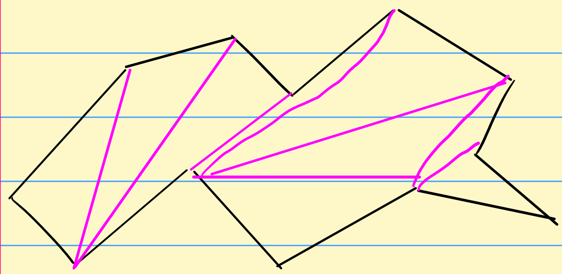
cube is a hexahedron [six sides]

If we have a guard @ every vertex of a polygon, we cover it entirely, but this is not true for polyhedron. How? Think of an example.

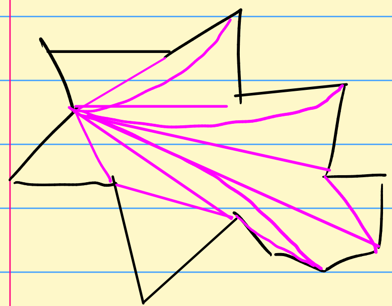
↳ Schonaret polyhedron.

01.08.2024

Draw max no. of non-crossing diagonals that lie inside a polygon



} Polygon-partioning problem
(or)
triangulating

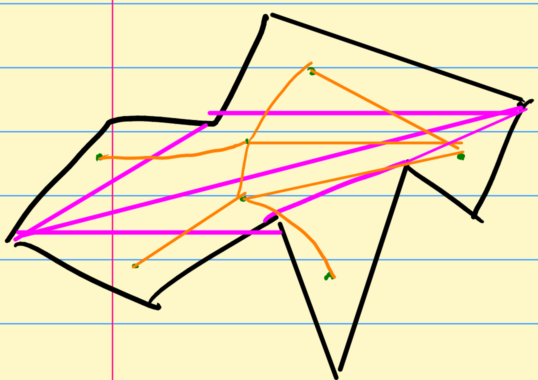


Graph

A set of vertices interconnected by edges

Directed graph ⟷ digraph

For the triangulated polygon, take vertices as nodes & sides & diagonals are edges of the graph

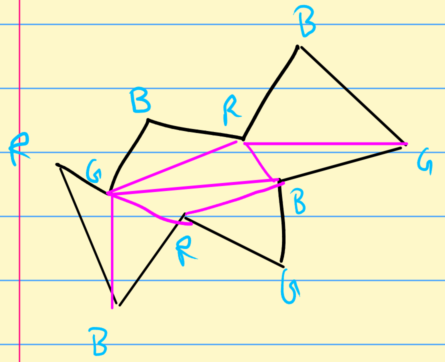


Green nodes → connect iff they share a diagonal → orange edges

Finding shortest path in a graph is a popular problem

Graph Coloring

Take R, G, B for triangles such that no 2 adjacent vertices have the same color



Observations [Forced coloring]

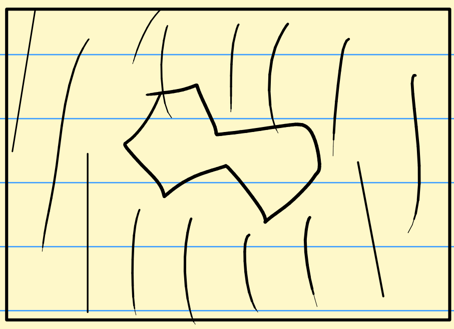
1. Every $\triangle$les has RGB
2. Always possible
3. Can reach any color from any other color in 2 steps or lesser
4. It is sufficient to place guards @ any 1 color to cover every triangle

This gives $\lfloor \frac{n}{3} \rfloor$

From generalised pigeon hole principle

Suppose n pigeons k holes, atleast one hole must have lesser than $\frac{n}{k}$ pigeons

Annulus Polygon



is $\left\lfloor \frac{n}{3} \right\rfloor$ still the correct answer

Triangulate, color and observe

02.08.2024

Assumptions made

1. 360°
2. Infinite vision
3. Stationary guards
4. No holes
5. Vertex guards
6. No curves
7. 2D problem
8. 1 vertex - 1 guard

Hunter's Problem [Fortress problem]

[It's outside the polygon]

In 3D → tetrahedronizable

Any 2D polygon can be triangulated but not every 3D polyhedron is tetrahedronizable.

Point Guard

The guard can be placed on (on) inside the border

The optimal case in the art gallery problem is np hard problem

1st Assignment

Pick 5 varieties and find out state of the problem

Hints:-

1. Illumination/Visibility problem

Look for survey/review paper. [IJCGA, CGTA, ACM journal]

Int. Journal on comp. Geo App.

Comp geo theory application

on experimental algorithms

Deadline:

Author, title, journal name, year published, page no.

Problem statement

Assumptions made

State of the problem

half a page

LaTeX

05.08.2024 EoD via Moodle

05.08.2024

To represent a solid, we need vertices, edges & faces [B-Rep]
↓
Boundary Representation

CSG ⇒ constructive solid geometry.

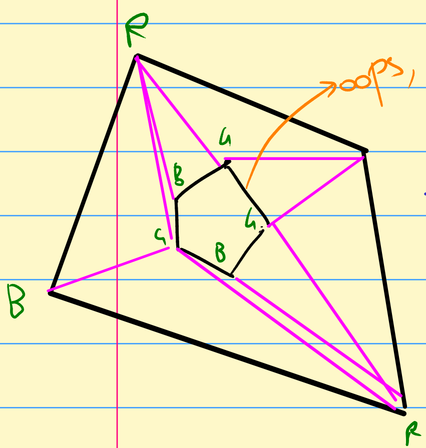
Half-edged Data Structure, Winged Edge [Data structures for solids]

A polyhedron can be represented as a collection of triangulated surfaces
Volumetric tetrahedronized files are also available.

.stl files are a collection of triangle. [very popular]
.obj, .step, .igs and many files exist
(.step ↓ B-Rep)
↳ meshlab is open-sourced for visualizing

All LaTeX files as a .zip & .pdf

polygon with holes



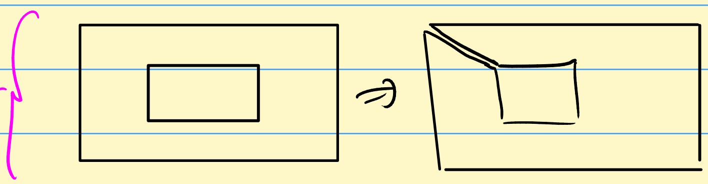
oops, monochromatic edge

Always the case? ⇒ No

→ $sol^n \Rightarrow \left\lfloor \frac{n+h}{3} \right\rfloor$ → h is no. of holes

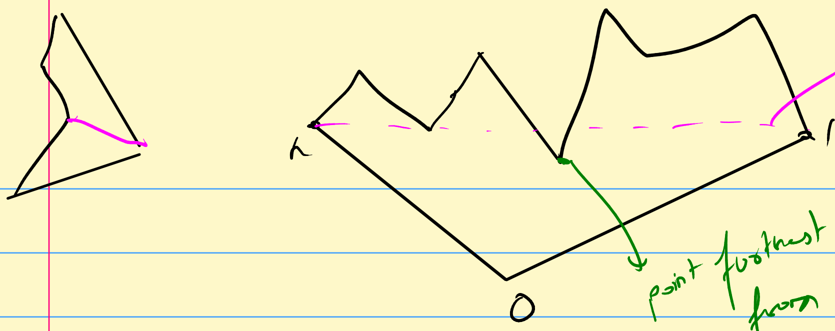
↓
Make multiply connected as simply connected

Try it out ←



But we must show every polygon can be triangulated

Every closed polygon will have at least one convex vertex
↓
use this to show every polygon has at least one diagonal



1-n is not a diagonal

point furthest from 1-n and join to 0.

$(n-2)\cdot 180^\circ$
↓
at least 3 convex

Once we prove diagonal exists, we recursively show triangulation is possible

↳ Every polygon has $(n-2)$ triangles & $(n-3)$ non-crossing diagonals
↓
Prove.

Always triangulatable, so how many triangulations? ⇒ H.W for next class

Take any 3 consecutive vertices in a polygon a, b, c. If ac is a diagonal, △abc is called an ear & b is called the ear tip.
Every $n \geq 4$ polygon will have 2 non-overlapping ears.
↓
Very interesting
Can be used for triangulating recursively

For $(n-3)$ non-intersecting diagonals proof,

after D diagonals, T triangles are formed with 3T sides

These 3T sides include double counted diagonal & single counted polygon sides

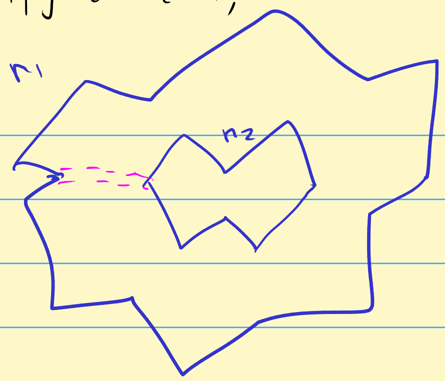
$$3T = n + 2D \Rightarrow T = \frac{n+2D}{3} \Rightarrow \frac{n+2D}{3}\cdot 180^\circ = 180^\circ(n-2)$$

$$\Rightarrow n + 2D = 3n - 6 \Rightarrow 2n = 2D + 6 \Rightarrow \boxed{D = n-3}$$

Hence, proved.

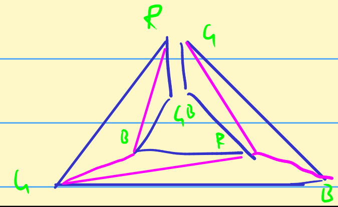
$$T = \frac{n + 2(n-3)}{3} = n-2 \Rightarrow \text{hence, proved.}$$

for multiply connected,



$n_1+n_2 \longmapsto n_1+n_2+2$

Now applying $G(n_1+n_2+2) = \left\lfloor \frac{n_1+n_2+2}{3} \right\rfloor$



12.08.2024

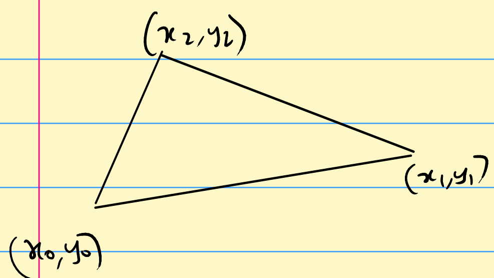
No. of triangulations = CATALAN NUMBER!! $\frac{{}^{2n}C_n}{n+1}$

On the visibility of continuous curves [paper]

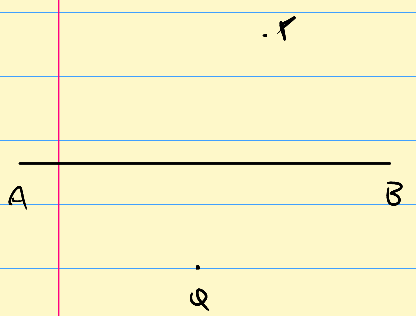
→ Approximation ratio = 2 → our solution is not twice worse than the optimal sol^n

→ Every polyhedron need not be tetragonizable → Schonard's polyhedron

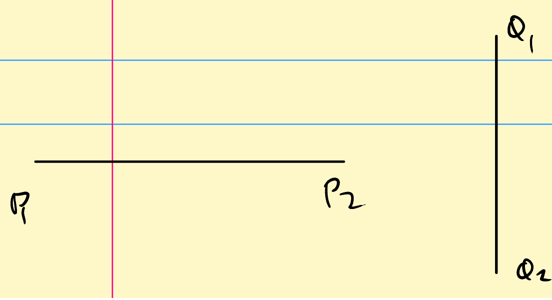
→ Given a polygon you show a triangulation → finding intersection



$$= \frac{1}{2}\begin{vmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix}$$ → area of the



sign of areas is opposite, opposite sides of line segment.

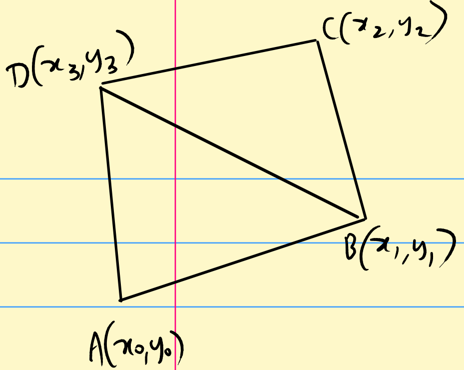


We need 4 cross products to find the area

↳ called PREDICATE METHOD

left on predicate

write a code for this



$$\text{area} = \frac{1}{2}\left|\vec{AB} \times \vec{AD}\right| + \frac{1}{2}\left|\vec{BC} \times \vec{BD}\right|$$

$$= \frac{1}{2}\left[(x_1 - x_0)(y_3 - y_0) - (x_3 - x_0)(y_1 - y_0)\right]$$
$$+ \frac{1}{2}\left[(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)\right]$$

To find area of polygon, we need not evaluate the diagonals, i.e., no need of triangulation

$$= \frac{1}{2}\left[x_1\left(y_3 - y_0 - y_3 + y_1 + y_2 - y_1\right) + x_2(y_3 - y_1) + x_3\left(y_0 - y_1 + y_1 - y_2\right) + x_0\left(y_0 - y_3 + y_1 - y_0\right)\right]$$

$$= \frac{1}{2}\left[x_0(y_1 - y_2) + x_1(y_2 - y_0) + x_2(y_3 - y_1) + x_3(y_0 - y_2)\right]$$

$$= \frac{1}{2}\left[(x_0 - x_2)(y_1 - y_3) - (x_3 - x_1)(y_0 - y_2)\right] = \frac{1}{2}\sum(x_0 y_1 - x_1 y_0)$$

In general, $= \sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i)$; $x_n = x_0$; $y_n = y_0$

Next Assignment

→ input as the polygon, output as the triangulation [25th Aug]

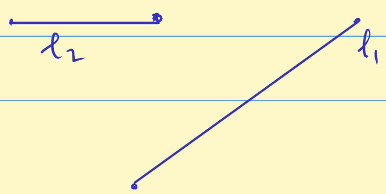
→ compare runtime with CGAL runtime

→ Capture the video and explain a video of max 8 min

→ Moodle Submission

16.08.2024

How to see if 2 line segments intersect?



∵ Both points of line l_1 are on same side, must not be intersecting

↳ But we must run this check for both line segments.

We need 4 of these predicates to determine intersection.

↓ How to use this in triangulation

How to compare 2 algorithms?

↳ Raw speed becomes machine dependant.

↳ So we use run time & space complexity.

```
i = 0        → initialization / assignment
while i < n
    i = i + 1
print i
```

clearly loop takes the most time wrt the loop

↳ Arithmetic, comparison are all CONSTANT TIME operations.

We consider the growth of the algorithm wrt input size [n]

n	n^2	n^3	2^n
1	1	1	2
2	4	8	4
3	9	27	8
4	16	64	16
⋮	⋮	⋮	⋮
1000	10^6	10^9	2^{1000}

growth is different. In fact $n + n^2 + n^3 < 2^n$ is smaller for large enough n.

For small input sizes, numbers are similar, but for large inputs, it does matter

Check what is underlying data structure allocation in python.

↳ python has identifiers & not variables [python takes 28 bytes]

Check

Memory management is key in any large program.

Linked list is a self-referencing structure

Passport: {Name, *next}, {Age, *next}

↗ the next ptr takes space even if it has nothing.

What is the bare minimum memory a self-referencing structure take?

Big O Notation

↳ Best case analysis, worst case analysis, average case analysis

worst case → Big O

average case → harder so we don't do → only amortized form

HW

Go to any data structure book & find exact defⁿ of Big O Notation

```
i := 0
while i < n
    print i
    i := i+1
```

worst case $O(2n+1) \sim O(n)$

Quick sort is $O(n^2)$. Merge sort is $O(n \log n)$. To bring down complexity, algorithm becomes more complex.

```
i := 0
while i^2 < n
    i++
```

$O(\sqrt{n})$

```
i := 1
while i < n
    i := i × 2
```

$O(\log_2 n)$

↳ s.t runs in log n.

What is the complexity of triangulation? What is the algorithm? → confirm n^3 / n^4

↳ $O(n^3) \Rightarrow n^2$ for $^nC_2$ (selecting 2 points) & n for checking intersection

How to reduce $n^3 \to n^2$?

19.08.2024

Triangulation in simplest form is $O(n^4)$.

TO reduce to n^3, instead of checking n^2 diagonals, but we can get n diagonals if we know it's internal or not.

But is the complexity of detecting a diagonal is taking to n^4.

<u>In-cone Test:</u> Read!! [The book] → Use in assignment

↳ Diagonal type can be done in constant or linear time?

To reduce to n^3, we use ears of polygons. n^2 to find ear tip & n for recursion

But how to make it n^2 with ears? Available in book

Compare assignment runtime with Naive algorithm, not just CGAL

The book has the entire code! Read the book!!!

↳ Problem with assignment → Any random n-points need not be a simple polygon!
find a random polygon generator tool. ⇒ Extension

How to go below n^2? Can we do $n \log n$? Divide & conquer exists.

Any polygon can be broken into 2 open CHAINS.

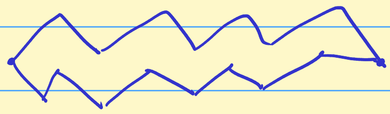
Monotone Polygon: If a polygon can be broken into 2 monotone polygon chains. [We consider monotone wrt y axis → always increasing y axis]

Why is this interesting? Coordinates are already sorted! No time to identify polygon?

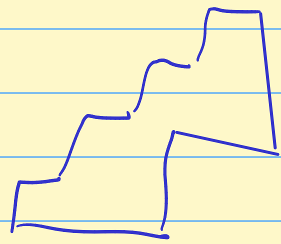
↳ Triangulating this monotone polygon?

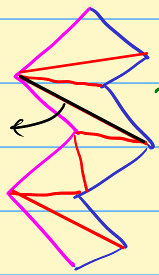
22.08.2024

Monotone polygon [about x-axis]



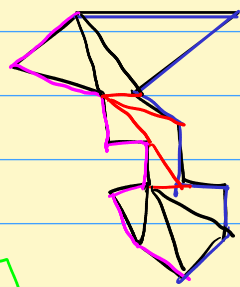
about y-axis



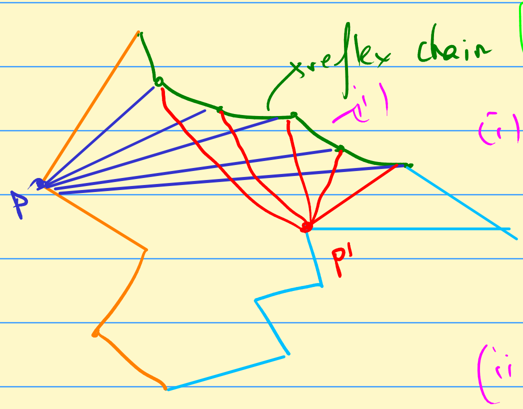


→ Check for diagonal intersection only if the other point on blue has a y-coordinate crossing y-coordinate of next point on pink

What about more complex monotone polygons? → Like



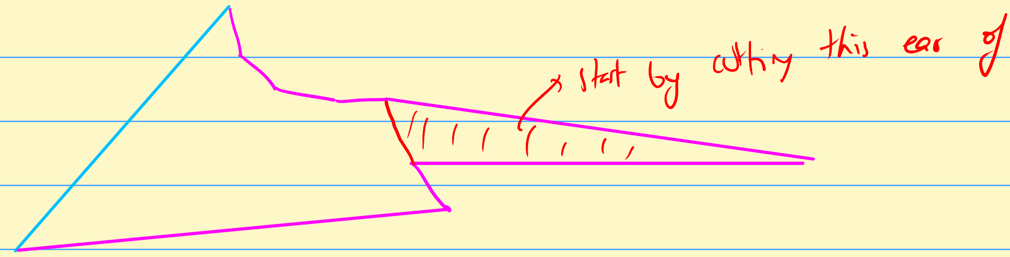
Alternating is very simple? only 2 hard cases exist



(i) All y-coordinates on reflex chain are above the normal chain.
↓
Join P with all

(ii) Somewhere the reflex chain turns & the some y-coordinates are below P.
↓
Join P' with all [forms ears]

Worst case is a reflex chain followed by a convex vertex.



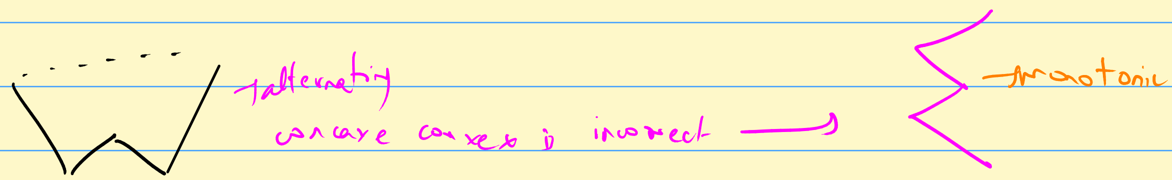
Doing this, we prevent $O(n^2)$ complexity

↳ Use stacks LIFO for implementation

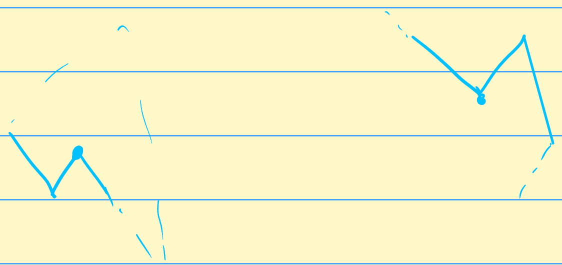
We're not repeating any diagonals, & $(n-2)$ diagonals, so only $O(n)$ complex
merging is $O(n)$

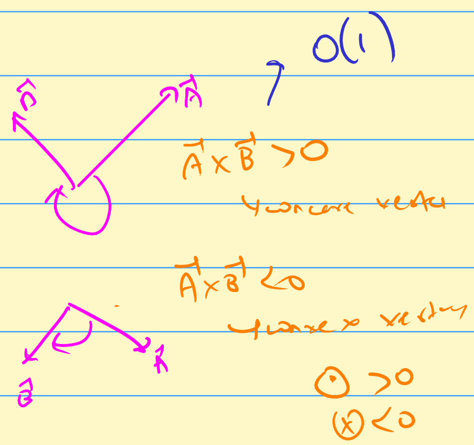
But how to identify if a polygon is monotone? How to find monotone polygons within a polygon? What is the complexity of cutting?

When is a polygon not monotone?



A reflex vertex that has adjacent vertices both on same side of the vertex are called Interior cusps.



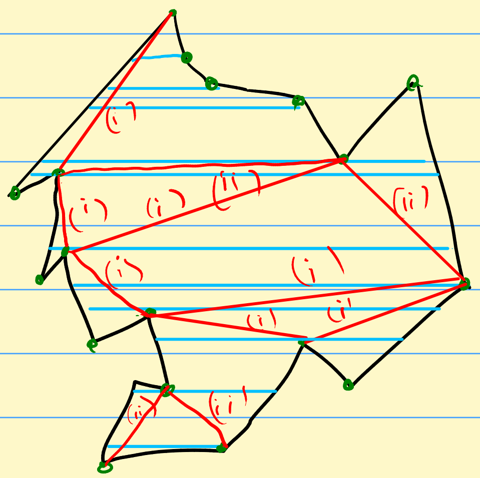


So we must remove these interior cusps.

Trapezoidalization → Quadrilateral with 2 $\parallel$lel sides.

↳ A $\triangle$le is a degenerate trapezoid

↳ How to?



Every line has a SUPPORTING VERTEX.

↓

interior supporting vertex [in that line] is an interior cusp

(i) At an upward cusp, connect to upward supporting vertex

(ii) At a downward cusp, connect to downward supporting cusp

Removing the ears, we get a monotone set of pieces!!

↳ we removed cusps.

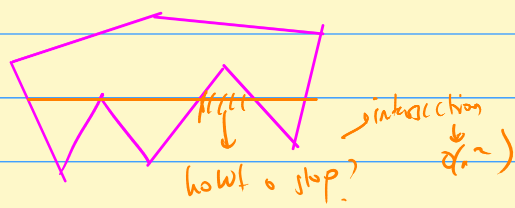
Monotone → $O(n)$ → top & bottom-most are simply the chain links

But how do we know where to stop drawing the horizontal lines

We have to use plane sweep idea!

↳ slide a line through a polygon & mark event when line intersects point

↳ Binary data structure → Insertion & Deletion



$O(n \log n)$ → for splitting → $O(n)$ → for triangulating.

HW

What other ways to partitioning can we do?

↳ Given a polygon, how to make convex pieces?

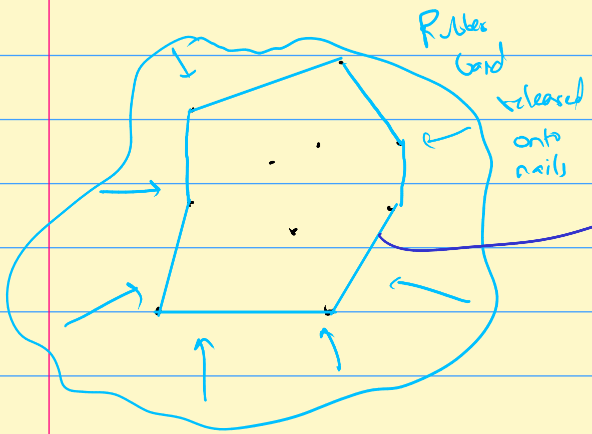
↳ come up with algorithm & complexity

23.08.2024

Convex Hull

↳ used everywhere. Explore

So far, we input an ordered set of vertices as a polygon into the computer



→ Always a convex polygon!
↓ called a convex hull as all points are inside or on it.

We mean the boundary when we say a convex hull

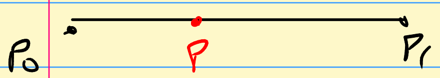
An application is in obstacle avoidance for robots. We form a convex hull around the obstacle. If we don't hit the convex hull, we won't hit the interior.

↳ Applied in MRI, cancer & other things.

Find out applications of convex hulls in polygon.

What is the difference btwn convex & concave polygons

(i) Reflex angle
(ii) There are some lines with points both inside & outside of P.



↳ Any point on P_0P_1 can be represented as a Linear combination of P_0 P_1.

$P = \alpha_0 P_0 + \alpha_1 P_1$; $\sum \alpha = 1 \longrightarrow$ between P_0 & P_1 [Parameterized form]

$P = \alpha_0 P_0 + P_1 - \alpha_0 P_1 = \alpha_0 (P_0 - P_1) + P_1$

$P(t) = (1-t) P_0 + t P_1$; $t \in [0,1] \rightarrow$ Convex combination

If we have 3 non-collinear points, we get points on or inside the Δle.
↳ $0 \leq \alpha_i \leq 1$, $\sum \alpha = 1$

H.W

Difference btwn convex combination & affine combination

Convex optimization gives points on (or) inside the convex hull.

Set of all combination combination gives the entire convex hull.

In D-dimensions, the set of all combinations of (D+1) points span a shape in D-dimensions.

Half-line/plane has all points on one side. Intersection of all such half-lines gives the convex hull!

↳ Try showing uniqueness of convex hull using convex combinations.

A convex hull is the smallest convex polygon enclosing all the points, so least area. Also the smallest perimeter.

↳ ~ Minimal spanning set
↳ basis set?

What if we allowed non-convex hulls, the minimal area is for the polygon touching all points. But perimeter increases.

Multi-objective optimization → Area-perimeter problem

Explore Minimum Area Polygonalization (MAP) problem. ↑ will be the convex hull

↳ How to find MinAP?