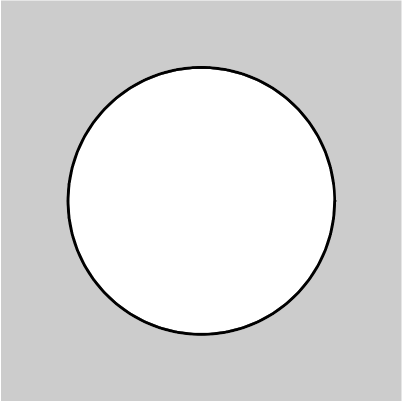
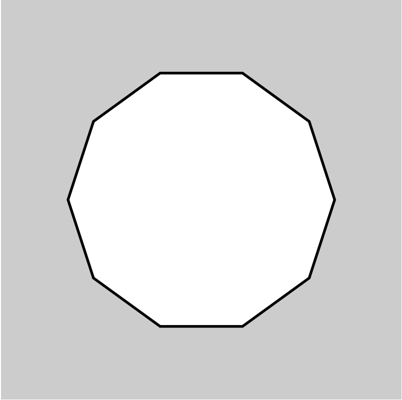
CMSC427 Points, polylines and polygons

Issue: discretization of continuous curve

- In theory, smooth curve:

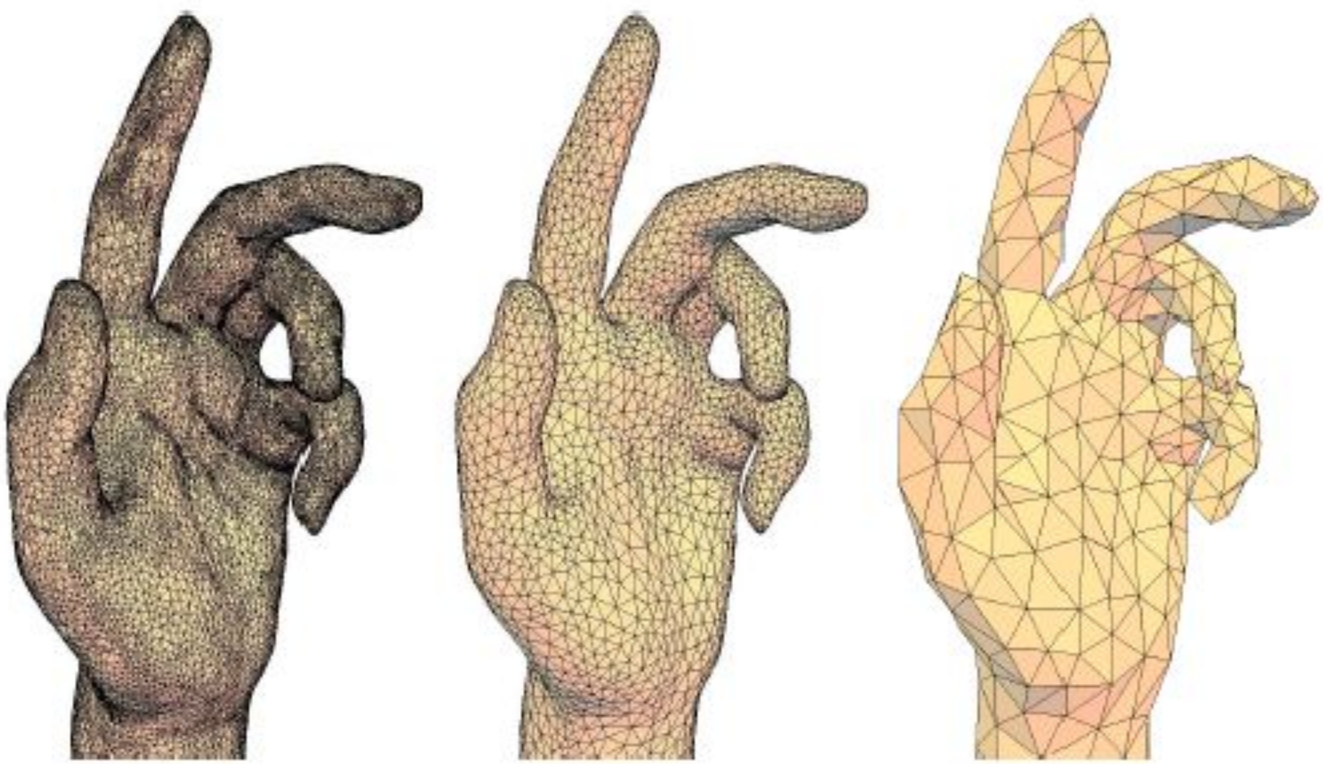


- In reality, piecewise discrete approximation:

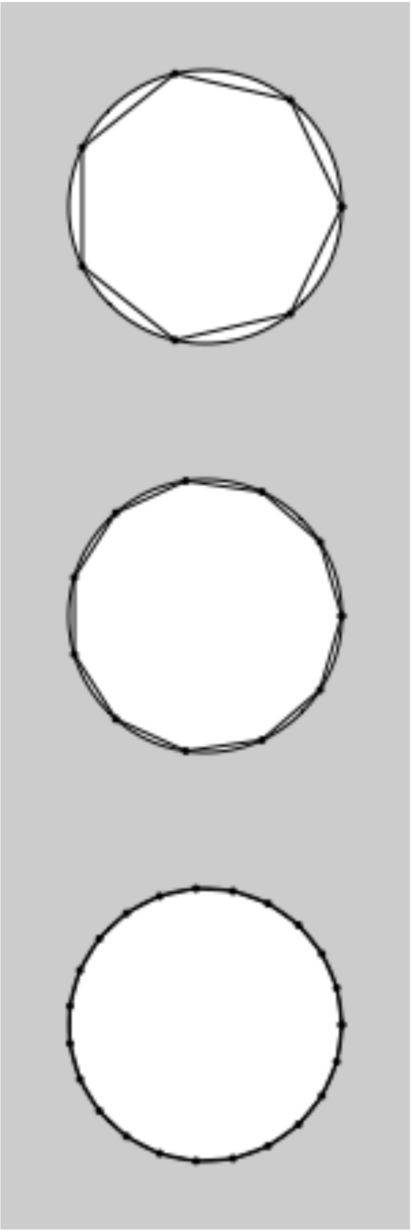


Modeling with discrete approximations

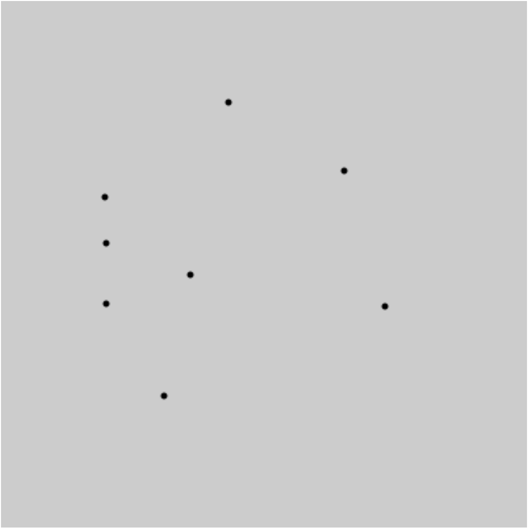
Increase fidelity with more points



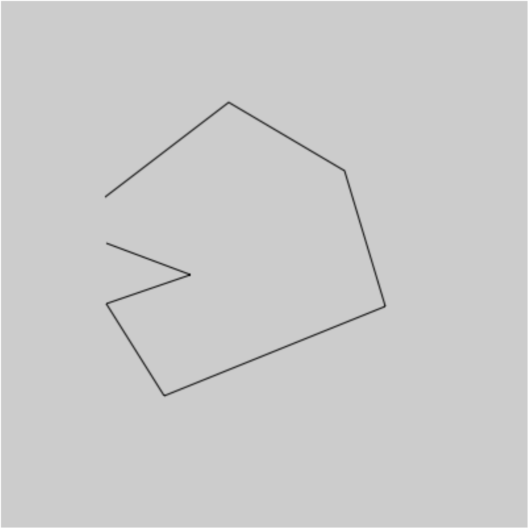
(a) 25,000 vertices. (b) 5,000 vertices. (c) 500 vertices.



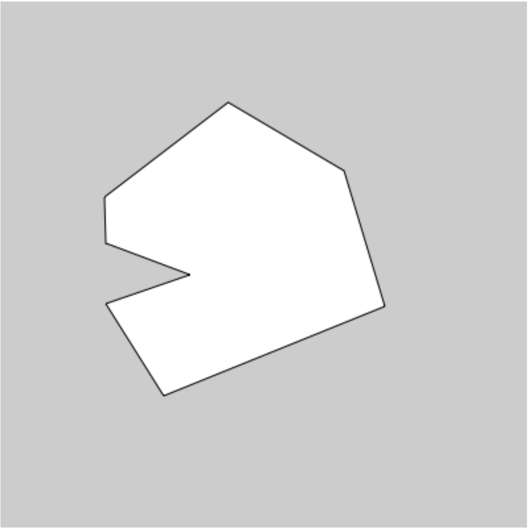
Points, polylines and polygons



Points
Also called
vertices



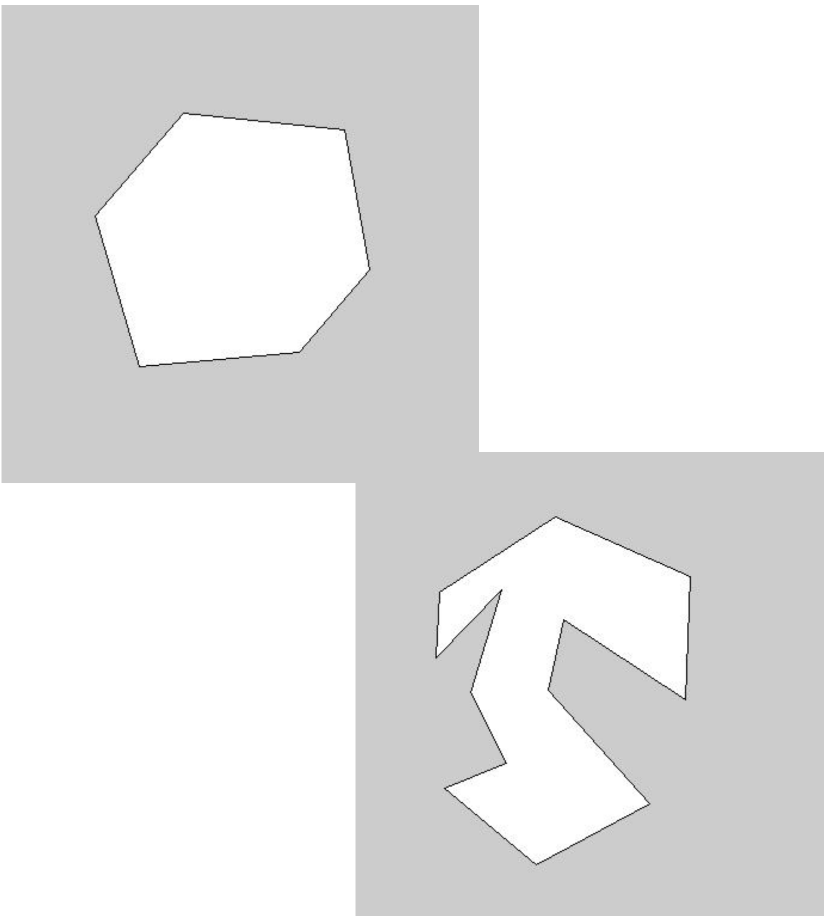
Polyline
Continuous sequence
of line segments



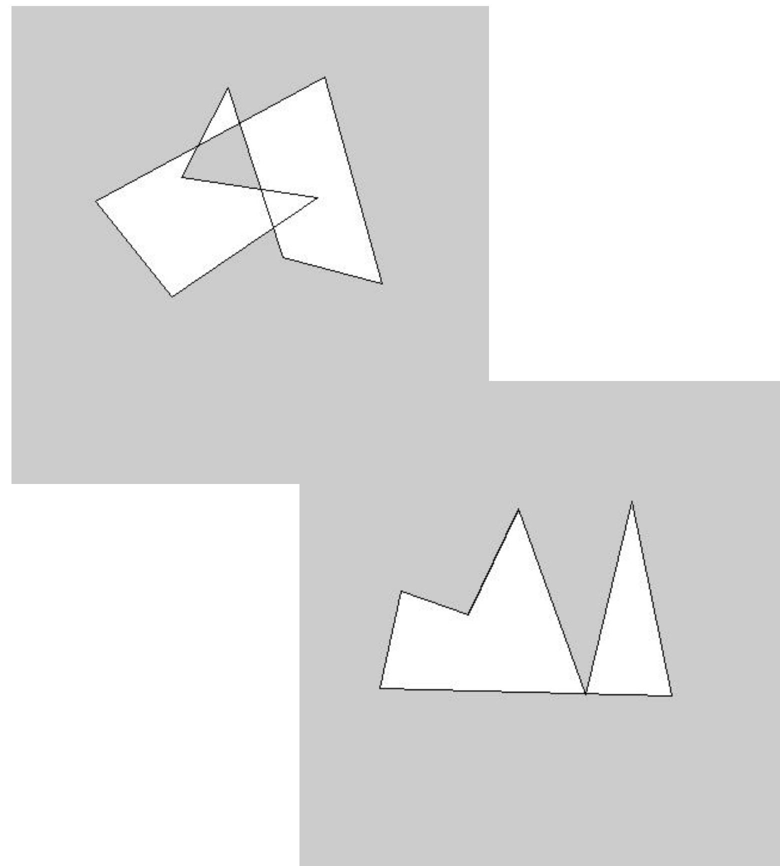
Polygon
Closed sequence
of line segments

Polygon properties I

- Simple
  no self-intersections
  no duplicate points



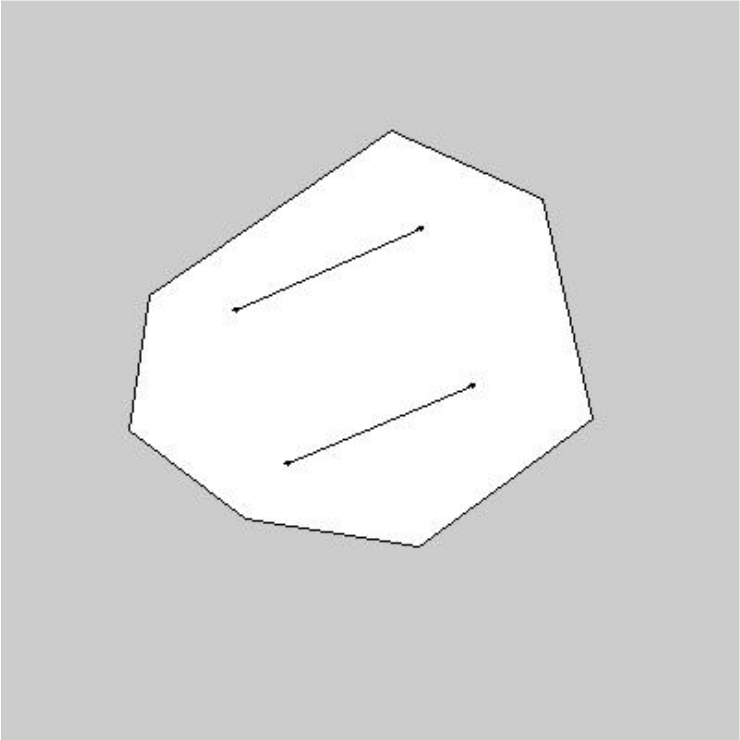
- Non-simple
  self-intersections
  duplicate points

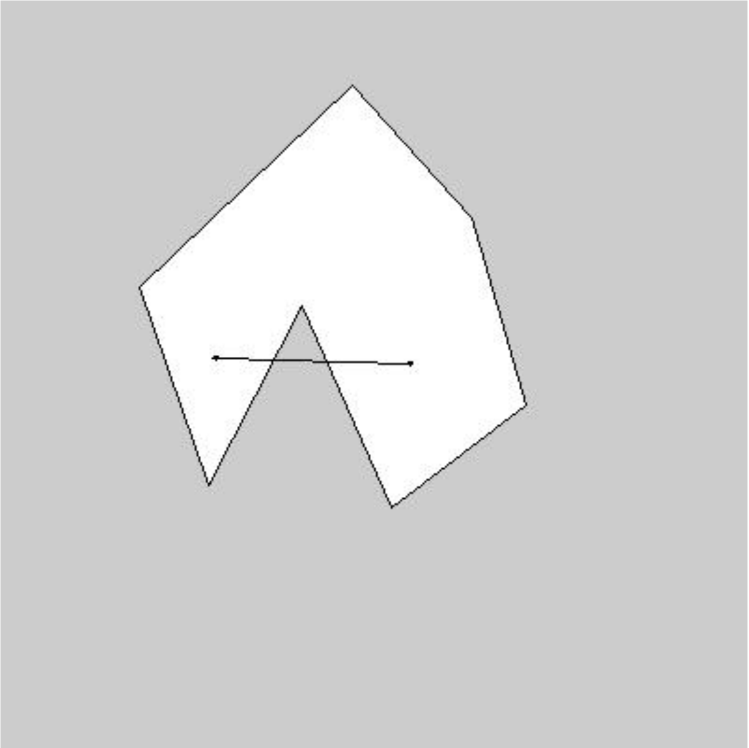


Polygon properties II

- Convex polygon
  Any two points in polygon can be connected by inside line

- Concave polygon
  Not true of all point pairs inside polygon

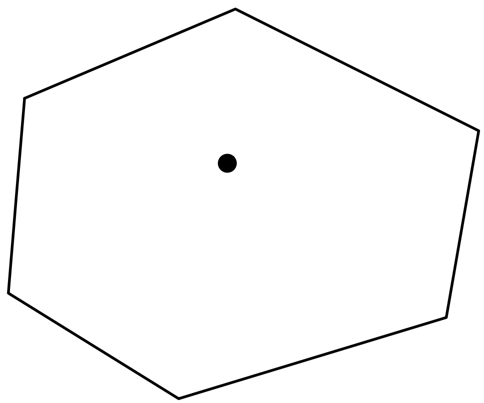




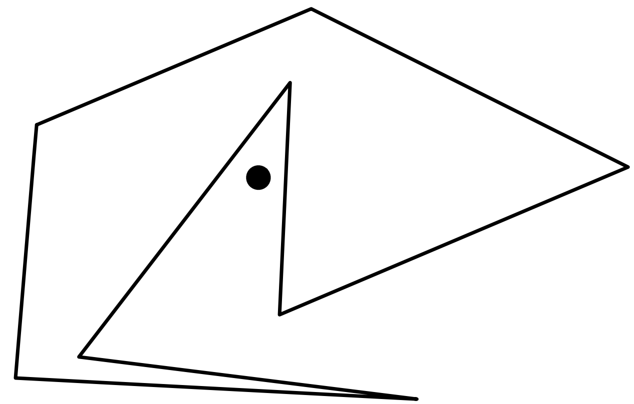
Point in polygon problem

Is P inside or outside the polygon?

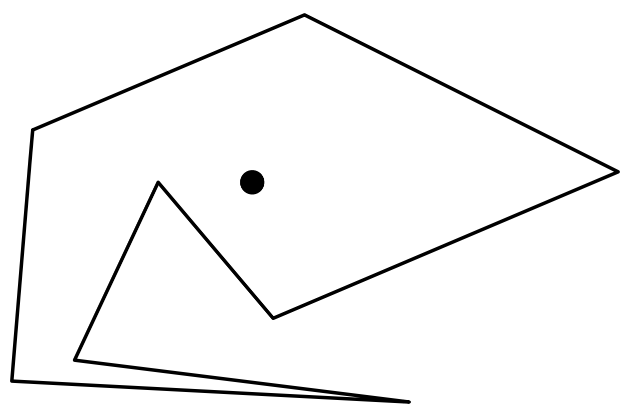
Case 1



Case 2



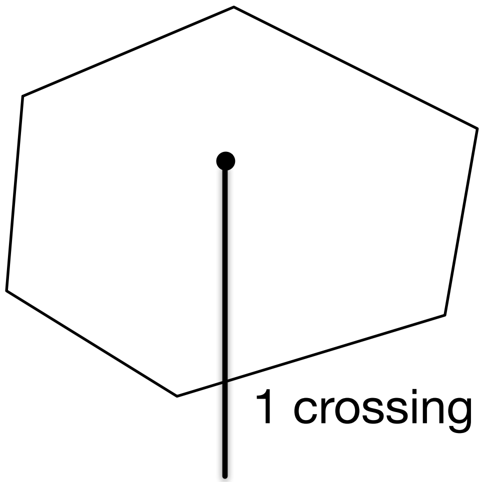
Case 3



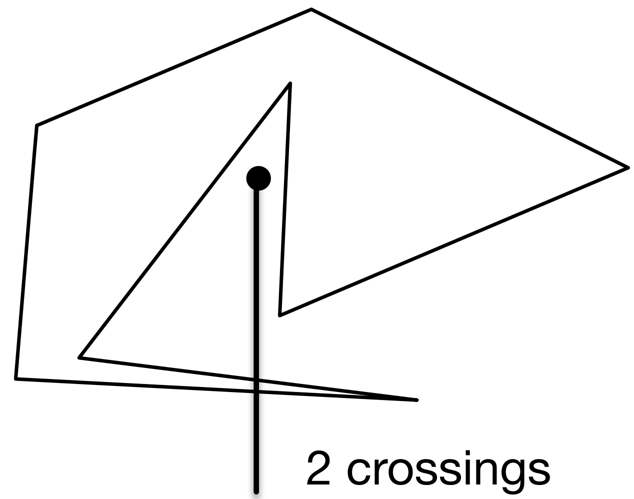
Point in polygon problem

Is P inside or outside the polygon?

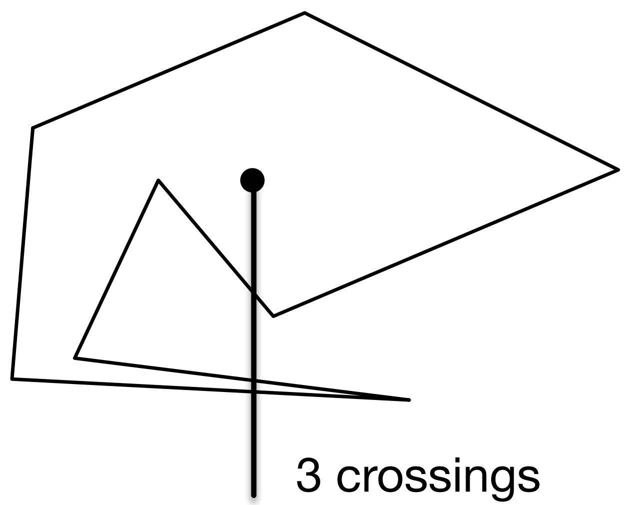
Case 1



Case 2



Case 3



Point in polygon problem

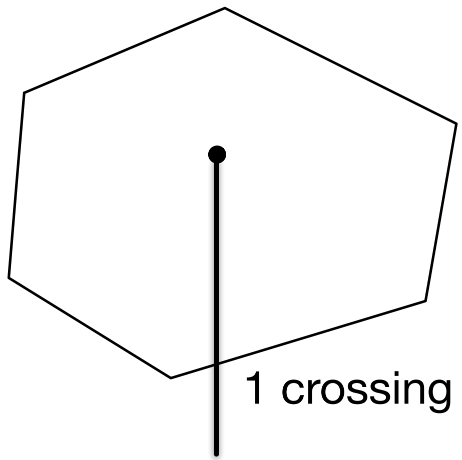
Is P inside or outside the polygon?

Odd crossings – inside

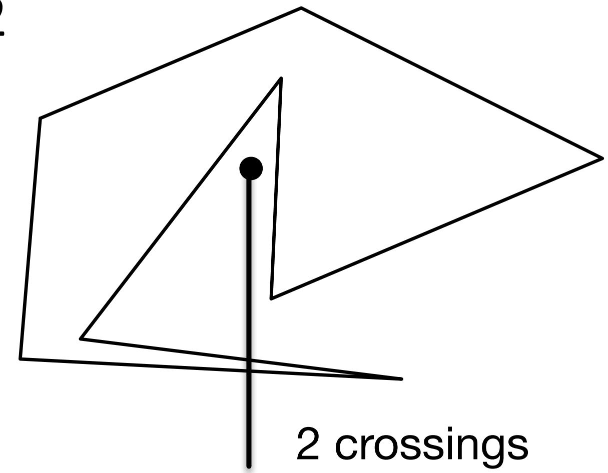
Even crossings – outside

Algorithmic efficiency?

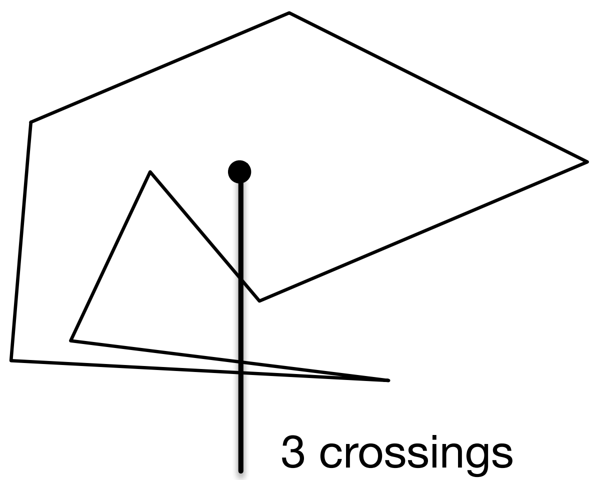
Case 1



Case 2

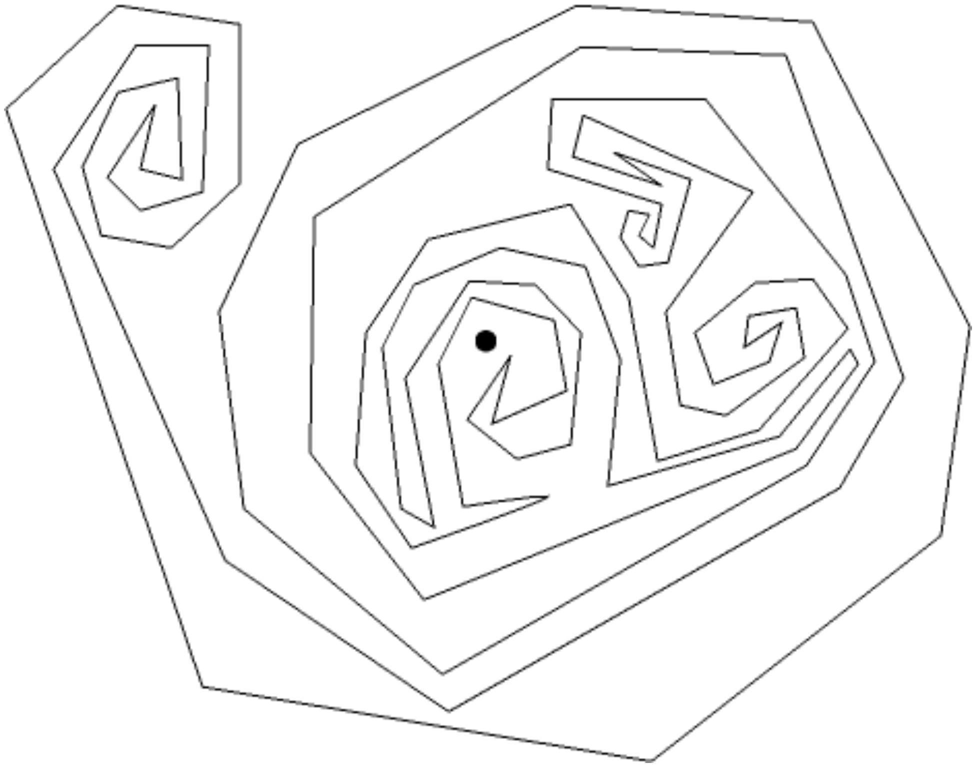


Case 3



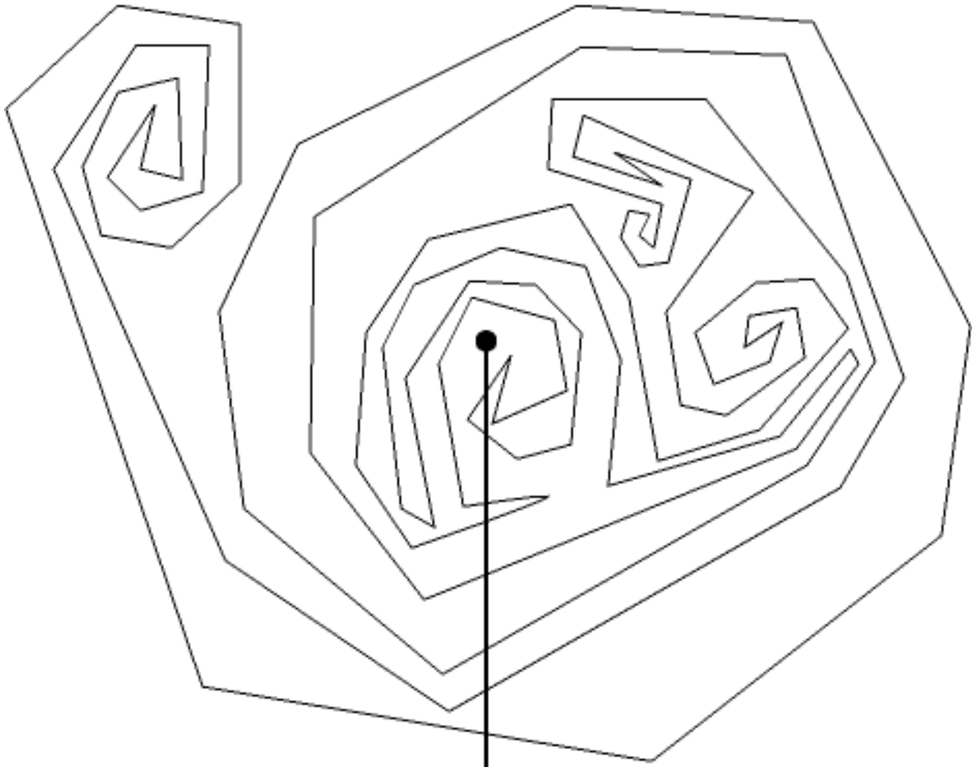
Point in polygon problem

Is P inside or outside the polygon?



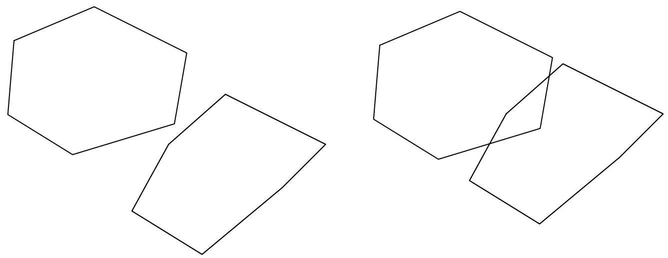
Point in polygon problem

Is P inside or outside the polygon?

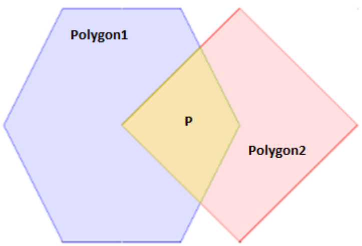


Other polygon problems

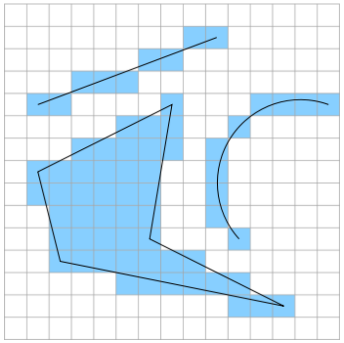
- Polygon collision
  - Return yes/no



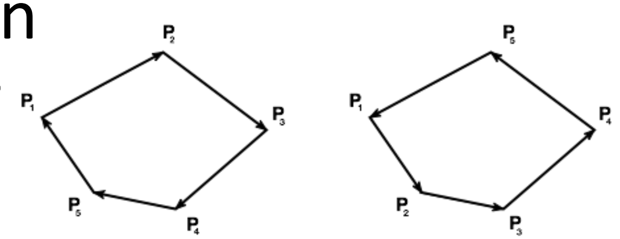
- Polygon intersection
  - Return polygon of intersection (P)



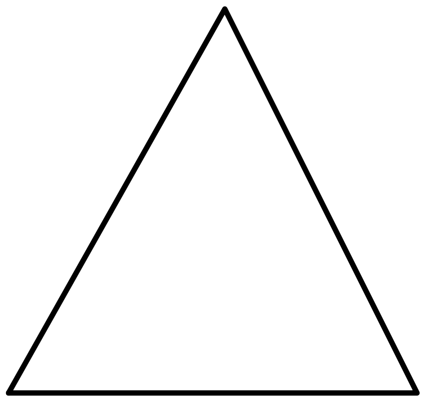
- Polygon rasterization
  - Return pixels that intersect



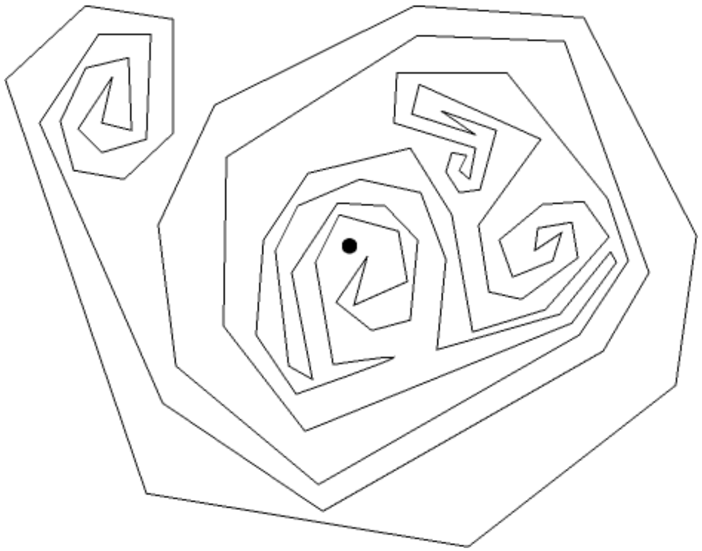
- Polygon winding direction
  - Return clockwise (CW) or counterclockwise (CCW)



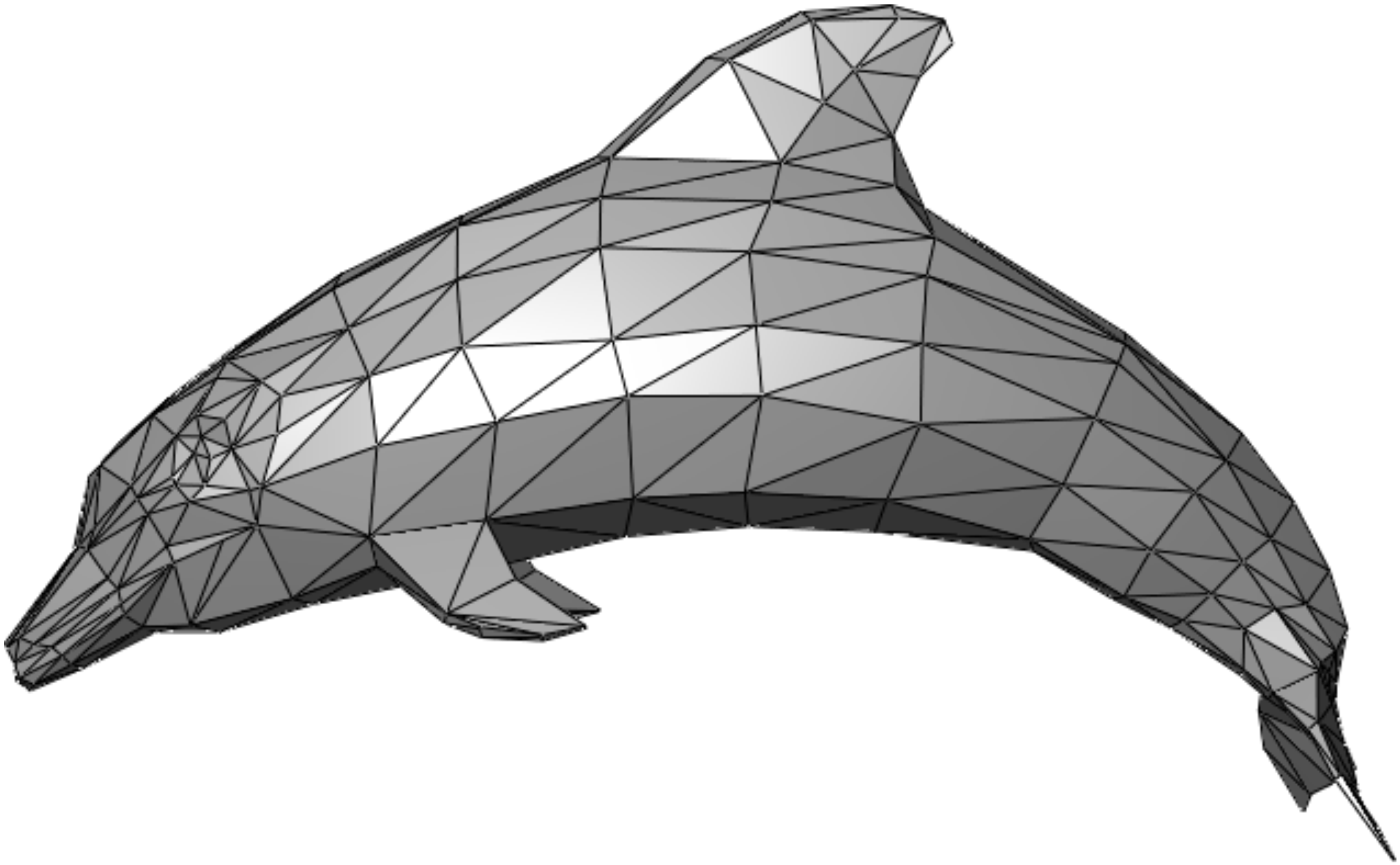
Moral: easier with simple, convex, low count polygons



Vs



Triangular mesh



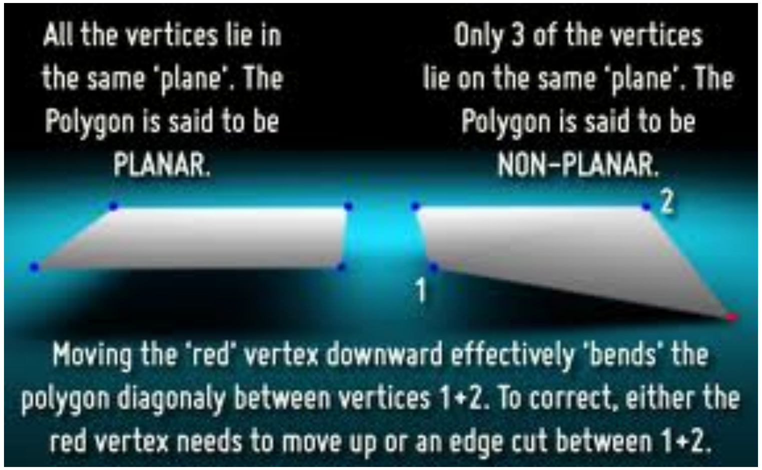
Why triangles …

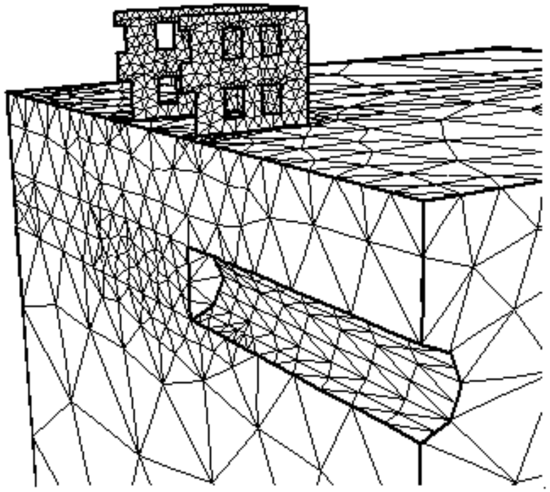
Why triangles?

1. Easiest polygon to rasterize

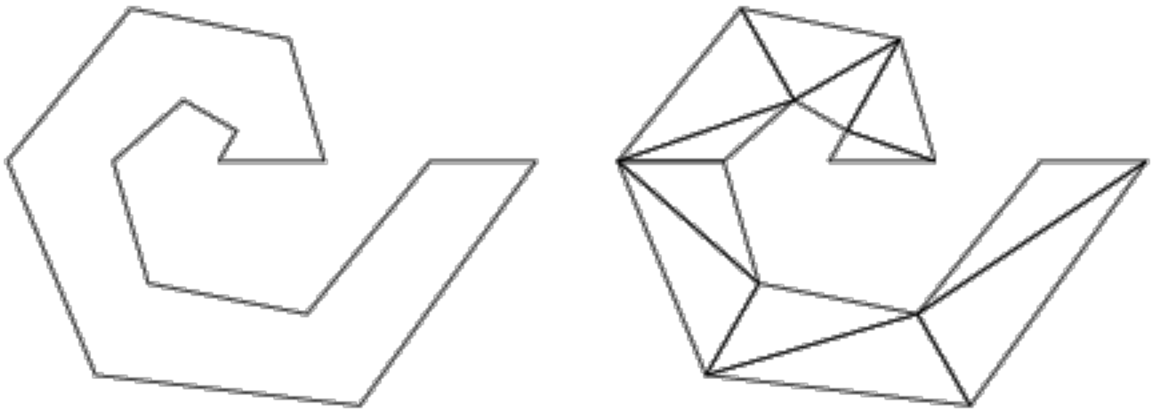
2. Polygons with n > 3 can be non-planar

3. Lighting computations in 3D happen at vertices - more vertices give smoother illumination effects





Polygon triangulation

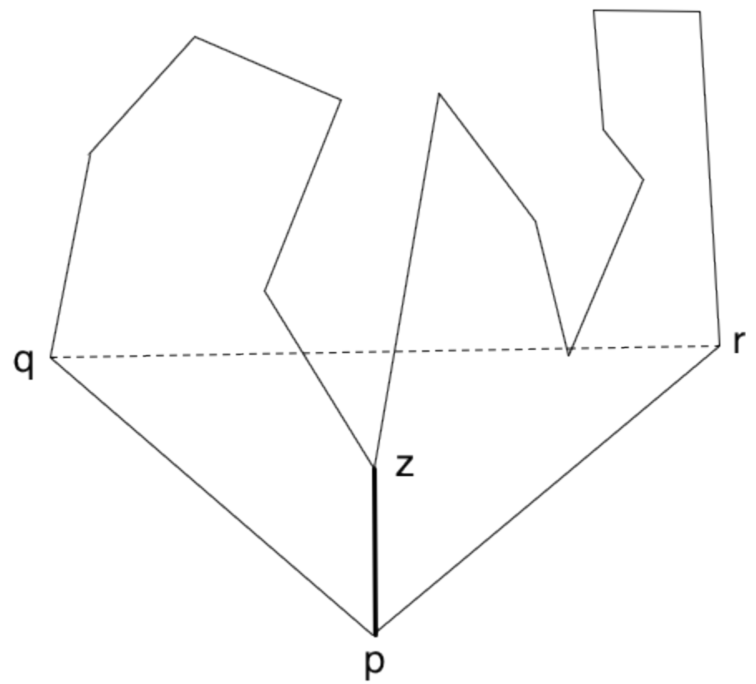


Theorem: Every simple polygon has a triangulation

- Proof by induction

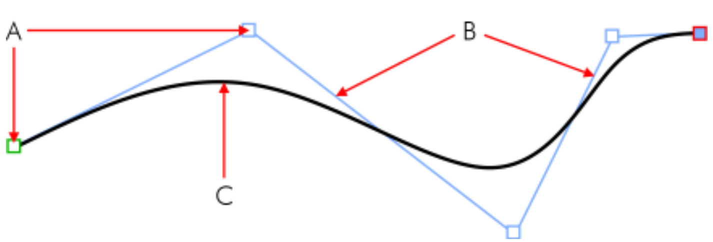
  Base case: n = 3

  Inductive case
  A) Pick a convex corner p. Let q and r be pred and succ vertices.
  B) If qr a diagonal, add it. By induction, the smaller polygon has a triangulation.
  C) If qr not a diagonal, let z be the reflex vertex farthest to qr inside △pqr.
  D) Add diagonal pz; subpolygons on both sides have triangulations.



Parametric curves vs. polylines

- Parametric curves
  1. Model objects by equation
  2. Complex shapes from few values
  3. Modeling arbitrary shape can be hard
- Polylines
  1. Model objects by data points
  2. Complex shapes need additional data
  3. Can model any shape approximately
- Looking forward
  - Use polylines to control general parametric curves
  - B-splines, NURBS



What you should know after today

1. Definitions of polyline and polygons
2. Polylines and polygons as piecewise discrete approximations to smooth curves
3. Definitions of properties of polygons (simple/non-simple, concave/convex)
4. Definition of point-in-polygon problem and crossing solution
5. Triangles are good (simplest polygon, always planar, easy to rasterize, more is good)
6. Definition of polygon triangulation (don't need to know the theorem yet)