Introduction to Geometric Algorithms

Applications of geometric algorithms

robotics, computer graphics, CAD/CAM, geographic information systems, ...

Geometric algorithms – scope

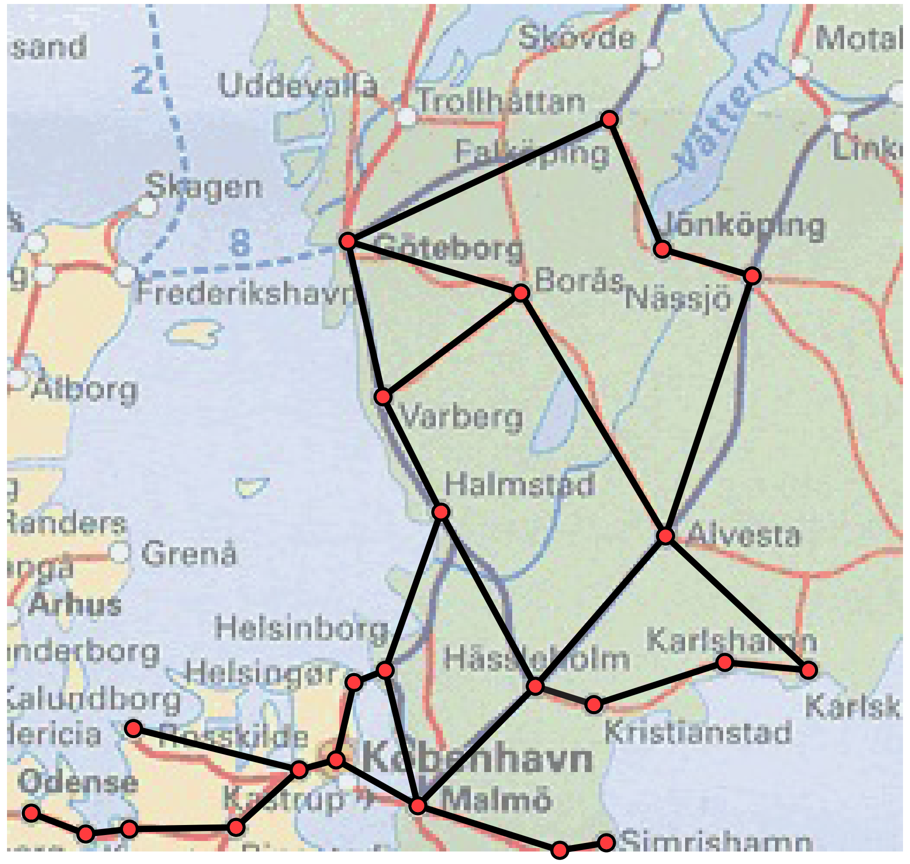
Geometric algorithms (practice):

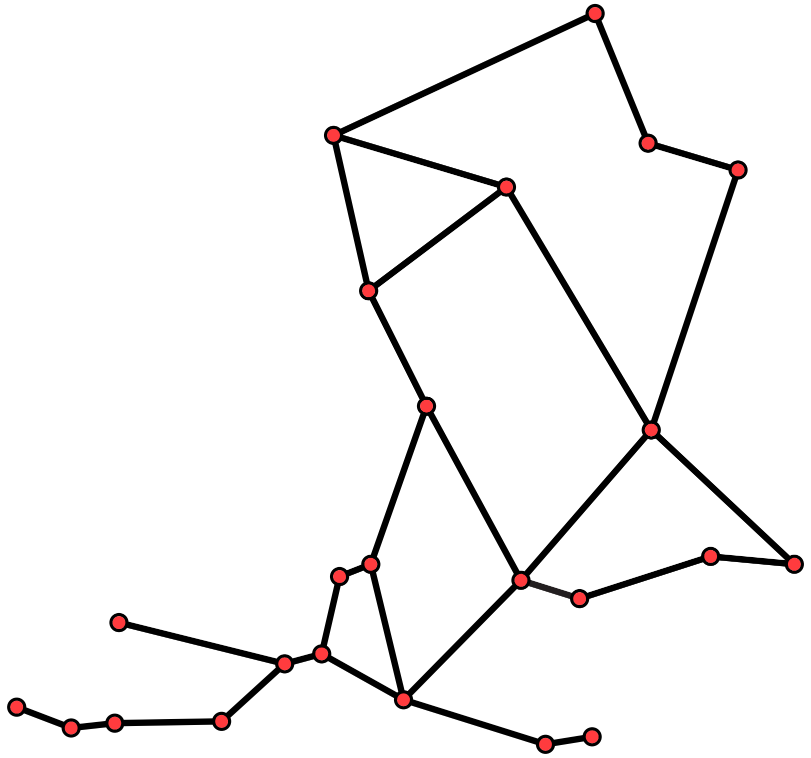
Study of geometric problems that arise in various applications and how algorithms can help to solve well-defined versions of such problems

Geometric algorithms (theory):

Study of geometric problems on geometric data, and how efficient algorithms that solve them can be

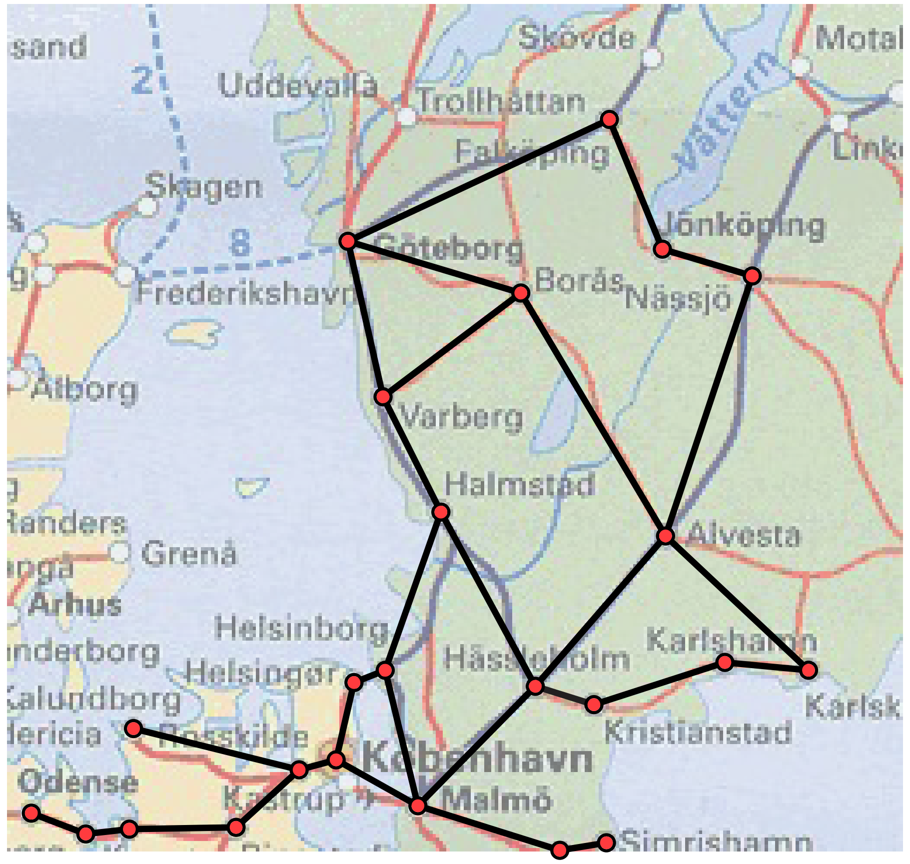
Geometric problems – examples

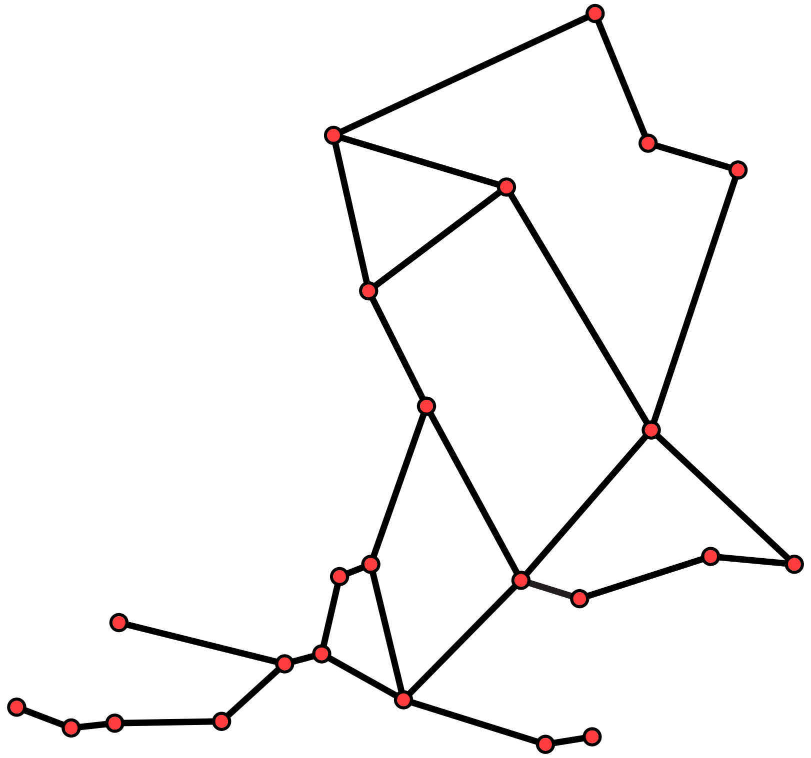




Geometric networks

Geometric problems – examples



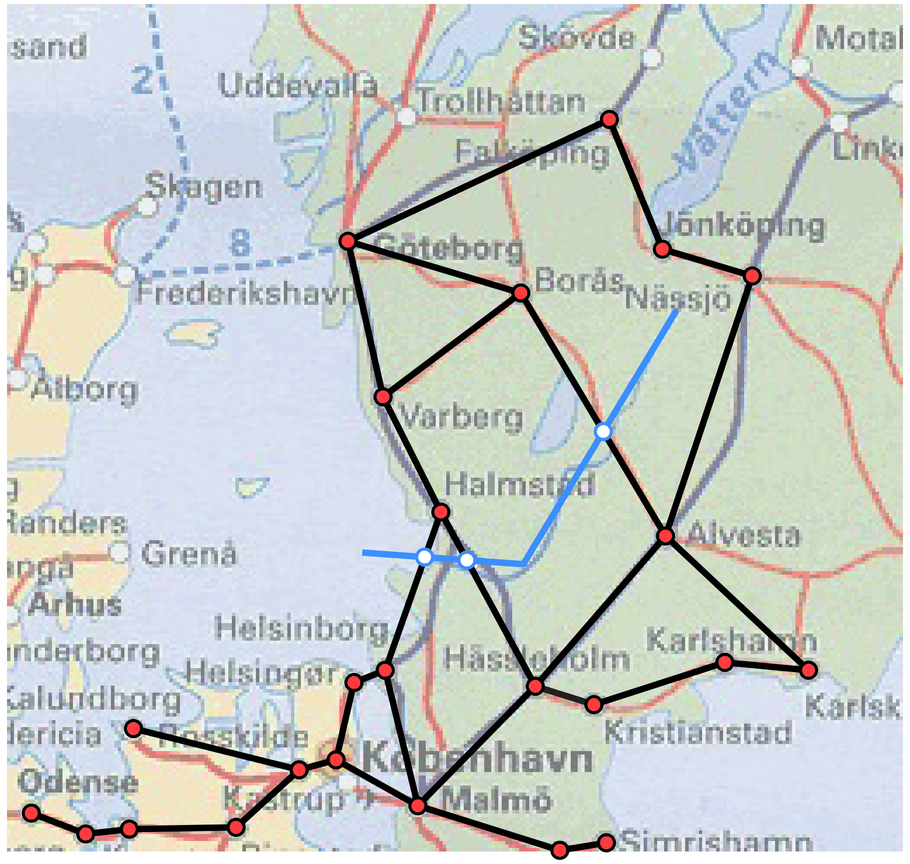


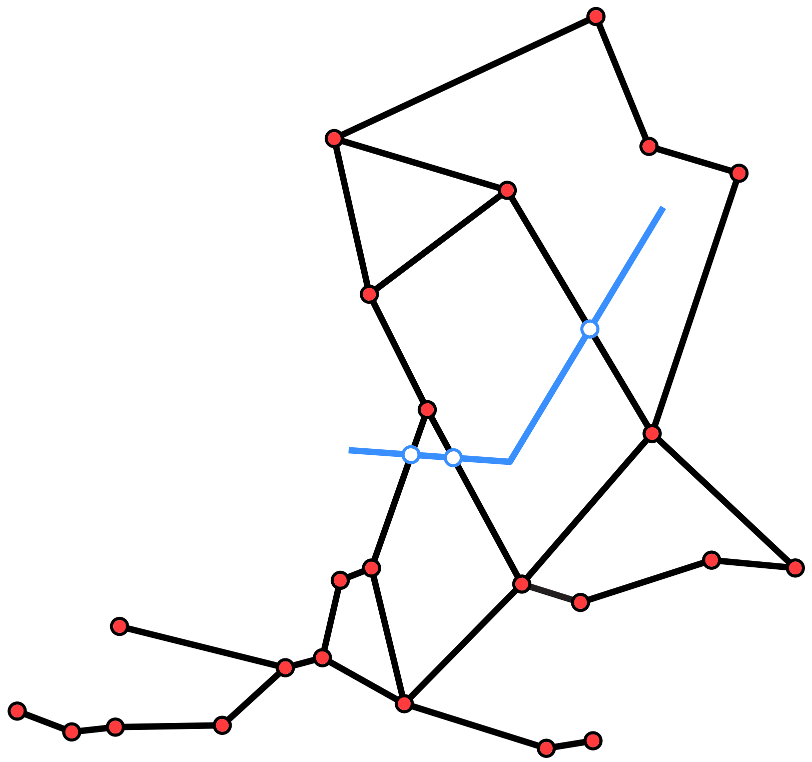
Geometric networks

- Are there relatively short routes between any two cities?

  Geometric spanner graphs (July 3)

Geometric problems – examples





Geometric networks

- Are there relatively short routes between any two cities?
  Geometric spanner graphs (July 3)
- Where do streets cross rivers?
  Line segment intersections (April 17)

Geometric problems – examples

Planar Subdivisions

- Which region did I click in?
  Point location queries (May 3)

Geometric problems – examples

Planar Subdivisions

- Which region did I click in?
  Point location queries (May 3)
- Zoom in a certain region?
  Range queries (June 12)

Topics of course

intro + plane sweep technique

- Convex hulls
- Line segment intersections, Plane sweep
- Art gallery problem, Polygon triangulation

Topics of course

intro + plane sweep technique

- Convex hulls
- Line segment intersections, Plane sweep
- Art gallery problem, Polygon triangulation

randomized incremental construction + Voronoi diagrams

- point location problem, vertical decomposition
- Delaunay triangulations
- Voronoi diagrams

Topics of course

intro + plane sweep technique

- Convex hulls
- Line segment intersections, Plane sweep
- Art gallery problem, Polygon triangulation

randomized incremental construction + Voronoi diagrams

- point location problem, vertical decomposition
- Delaunay triangulations
- Voronoi diagrams

arrangements + motion planning

- Line arrangements, Geometric duality
- Robot motion planning, Visibility graphs

Topics of course

intro + plane sweep technique

- Convex hulls
- Line segment intersections, Plane sweep
- Art gallery problem, Polygon triangulation

randomized incremental construction + Voronoi diagrams

- point location problem, vertical decomposition
- Delaunay triangulations
- Voronoi diagrams

arrangements + motion planning

- Line arrangements, Geometric duality
- Robot motion planning, Visibility graphs

Geometric Data Structures

- Range queries
- Windowing queries
- Quadtrees

Topics of course

intro + plane sweep technique

- Convex hulls
- Line segment intersections, Plane sweep
- Art gallery problem, Polygon triangulation

randomized incremental construction + Voronoi diagrams

- point location problem, vertical decomposition
- Delaunay triangulations
- Voronoi diagrams

arrangements + motion planning

- Line arrangements, Geometric duality
- Robot motion planning, Visibility graphs

Geometric Data Structures

- Range queries
- Windowing queries
- Quadtrees

Additional topics (may vary)

- Geometric spanners
- Well-separated pair decomposition
- Algorithm engineering

Before we begin

Geometry: points, lines, ...

Plane (two-dimensional), $\mathbb{R}^2$

Space (three-dimensional), $\mathbb{R}^3$

Space (higher-dimensional), $\mathbb{R}^d$

A point in the plane, 3-dimensional space, d-dimensional space:

$$p = (p_x, p_y), \quad p = (p_x, p_y, p_z), \quad p = (p_{x_1}, p_{x_2}, \ldots, p_{x_d})$$

A line in the plane: $y = m \cdot x + c$; represented by m and c

A half-plane in the plane: $y \leq m \cdot x + c$ or $y \geq m \cdot x + c$

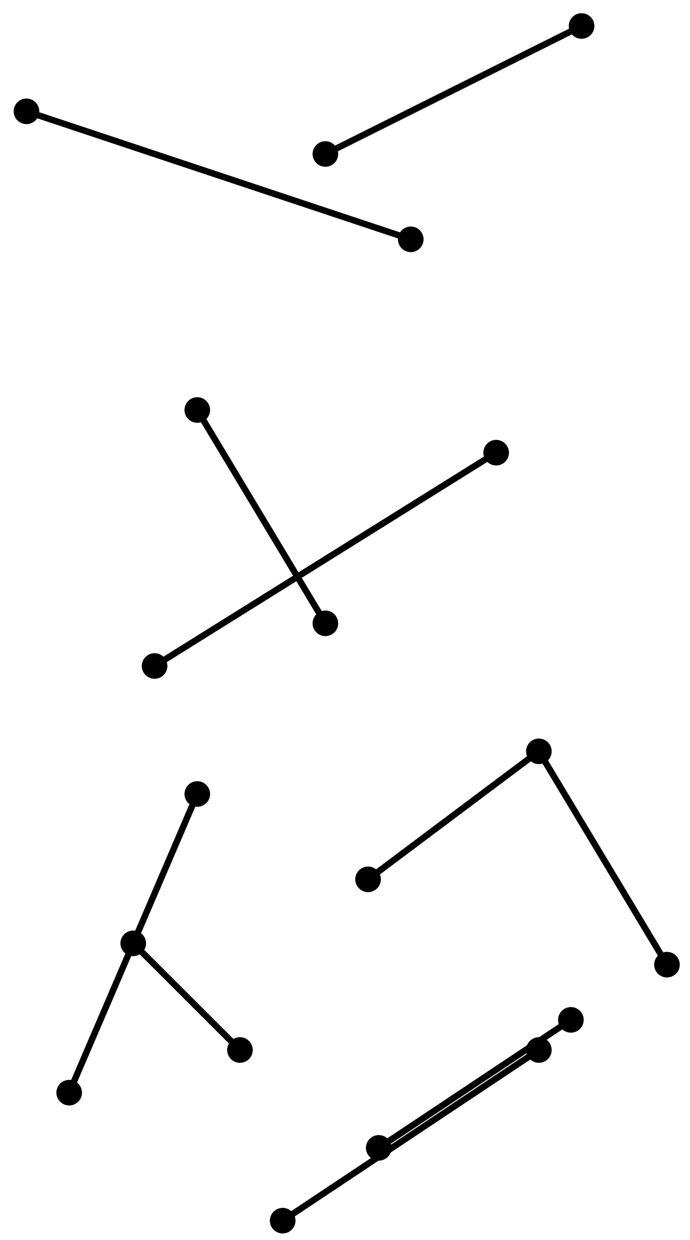
How to represent vertical lines? Not by m and c ...

Geometry: line segments

A line segment $\overline{pq}$ is defined by its two endpoints p and q: $(\lambda \cdot p_x + (1-\lambda) \cdot q_x, \lambda \cdot p_y + (1-\lambda) \cdot q_y)$, where $0 \leq \lambda \leq 1$.

Line segments are assumed to be closed = with endpoints (as opposed to open)

Two line segments intersect if they have some point in common. It is a proper intersection if it is exactly one interior point of each line segment.

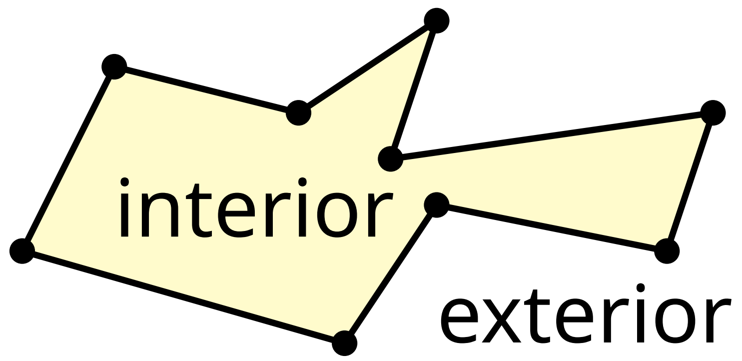


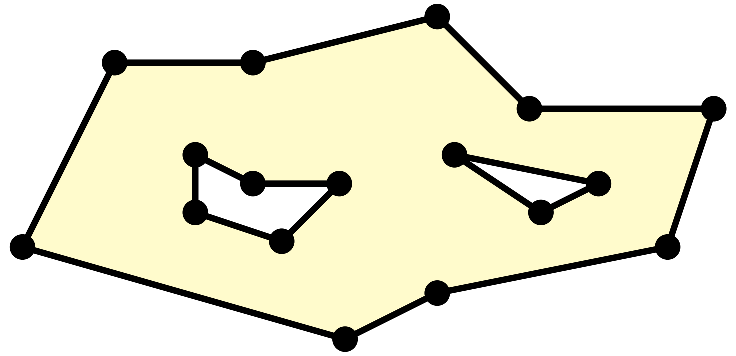
Polygons: simple or not

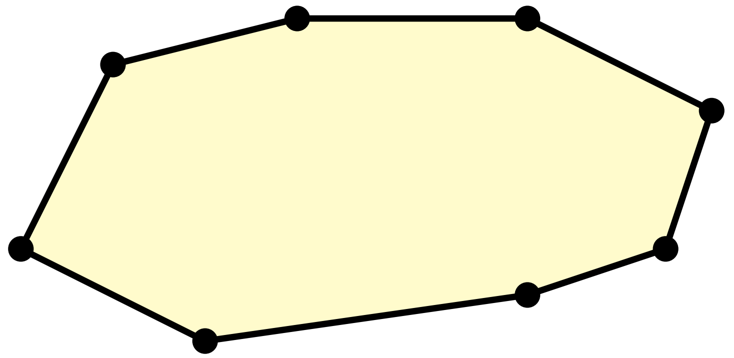
A polygon is a connected region of the plane bounded by a sequence of line segments

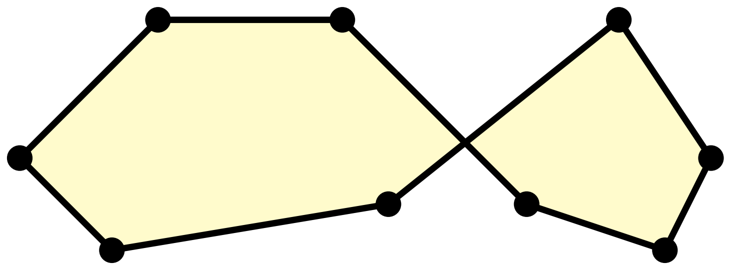
- simple polygon
- polygon with holes
- convex polygon
- non-simple polygon

The line segments of a polygon are called its edges, the endpoints of those edges are the vertices.





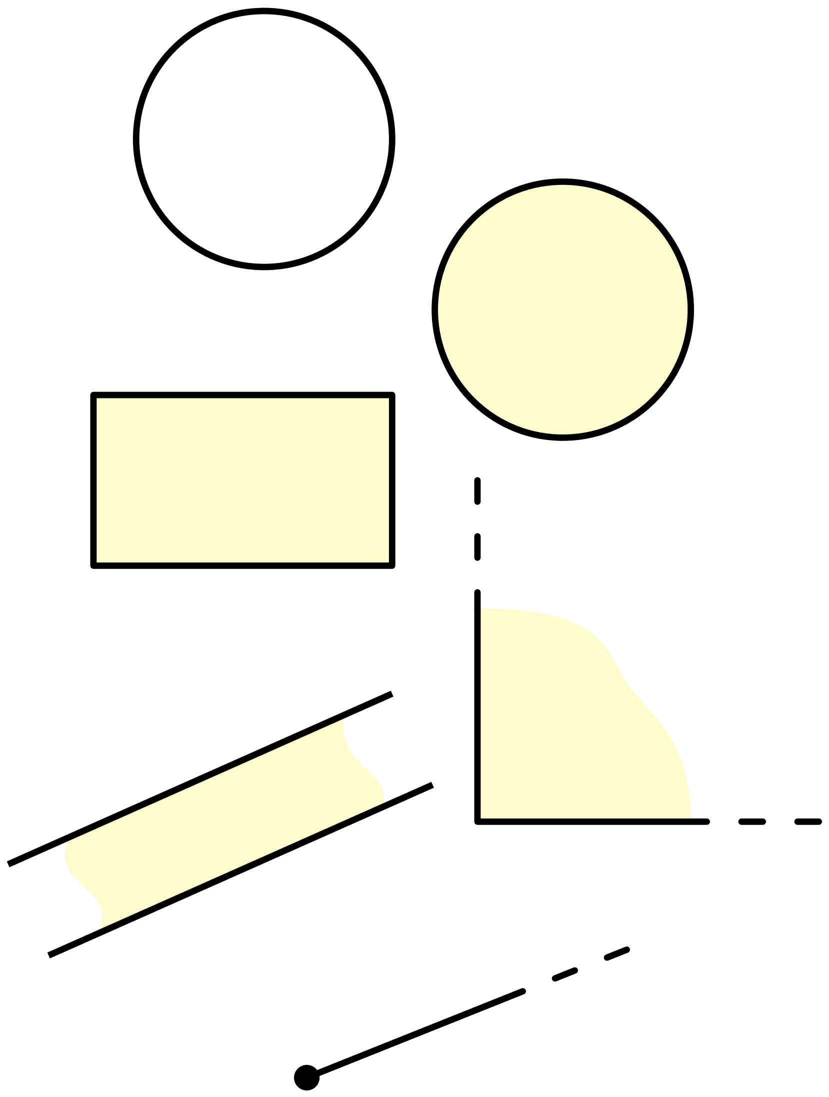




Other shapes: rectangles, circles, disks

A circle is only the boundary, a disk is the boundary plus the interior

Rectangles, squares, quadrants, slabs, half-lines, ...



Relations: distance, intersection, angle

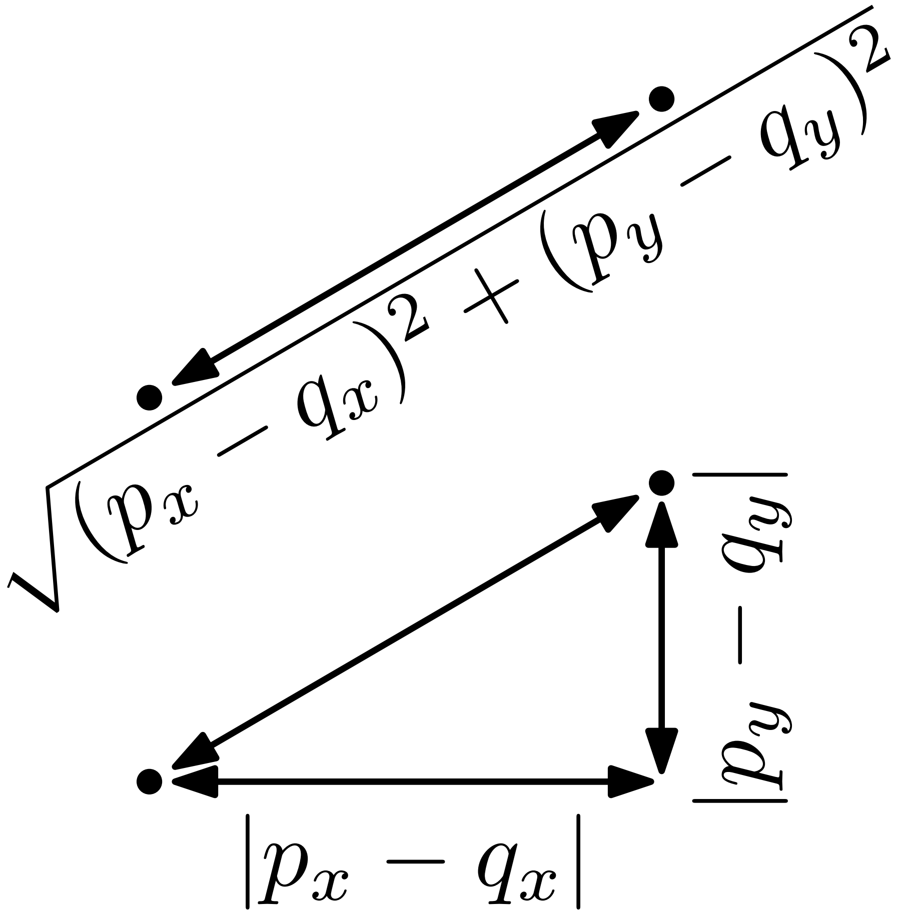
The distance between two points is generally the Euclidean distance:

$$\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

Another option, the Manhattan distance:

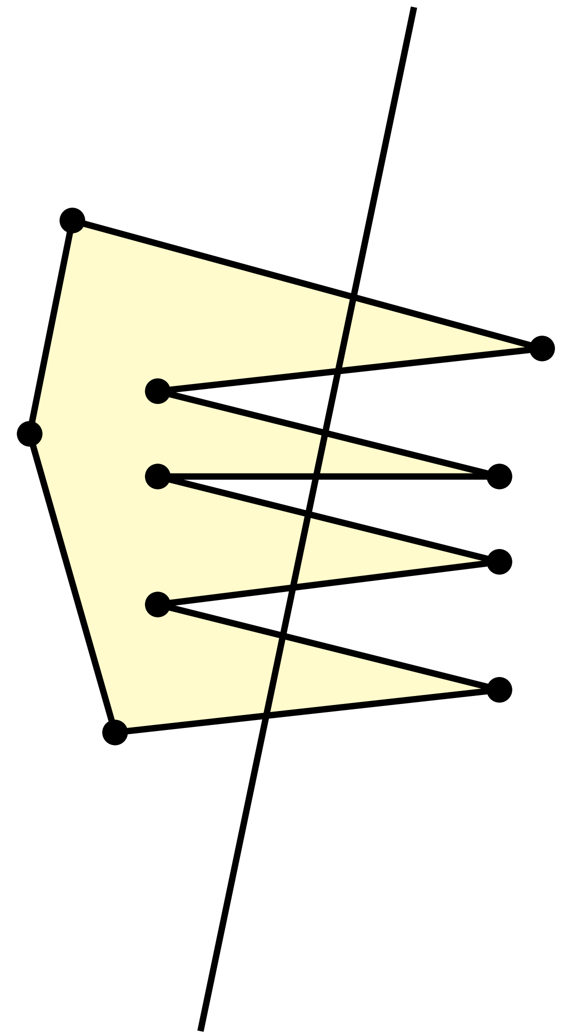
$$|p_x - q_x| + |p_y - q_y|$$

The distance between two geometric objects other than points usually refers to the minimum distance between two points that are part of these objects



Relations: distance, intersection, angle

Definition: The intersection of two geometric objects is the set of points (part of the plane, space) they have in common

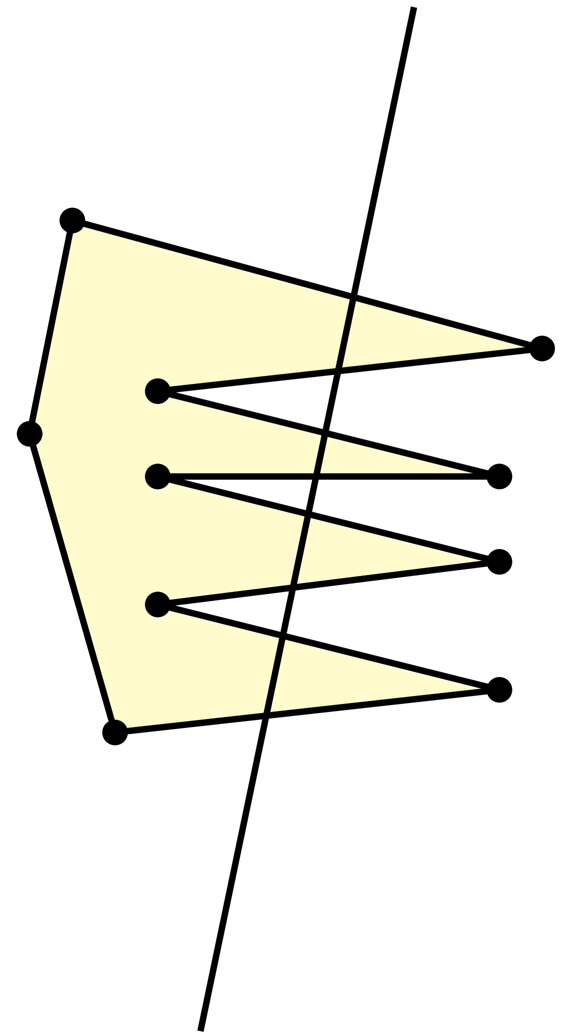


Question 1: What is the maximum number of intersection points of a line and a simple polygon with 10 vertices (trick question)?

Relations: distance, intersection, angle

Definition: The intersection of two geometric objects is the set of points (part of the plane, space) they have in common

Question 2: What is the maximum number of intersection points of a line and a simple polygon *boundary* with 10 vertices (still a trick question)?



Description size

A point in the plane can be represented using 2 reals

Description size

$$y = m \cdot x + c$$

A point in the plane can be represented using 2 reals

A line in the plane can be represented using 2 reals and a Boolean (for example)

$$\{false, m, c\}$$

$$x = c$$

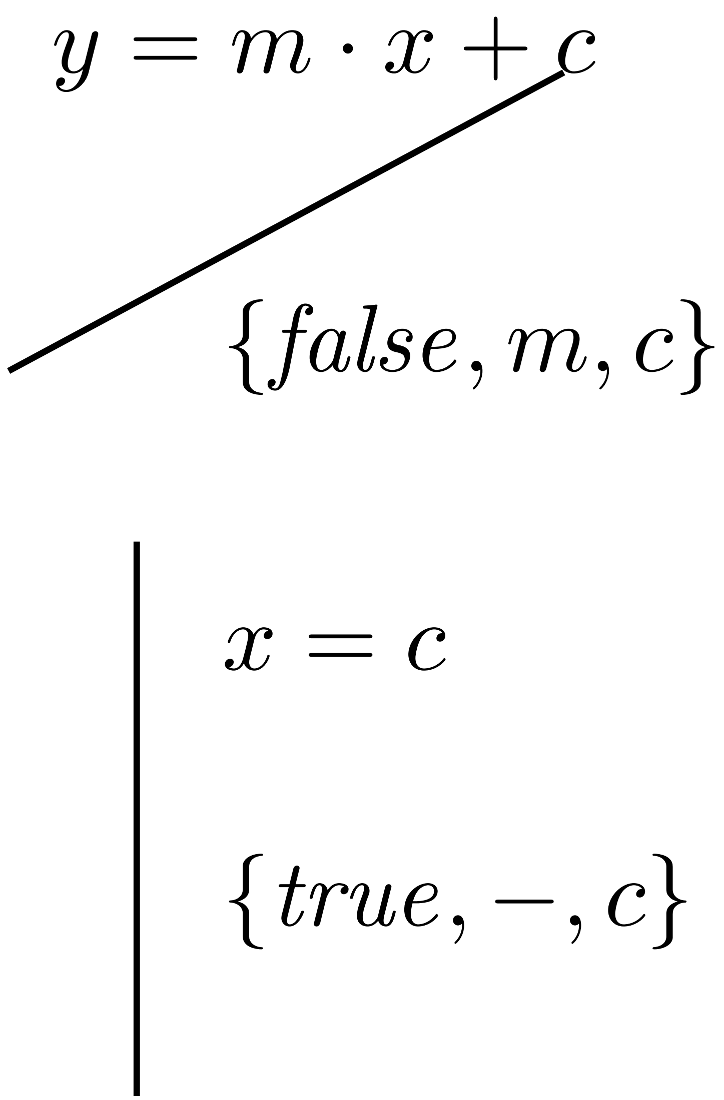
$$\{true, -, c\}$$

Description size

A point in the plane can be represented using 2 reals

A line in the plane can be represented using 2 reals and a Boolean (for example)

A line segment can be represented by 2 points, so 4 reals



Description size

A point in the plane can be represented using 2 reals

A line in the plane can be represented using 2 reals and a Boolean (for example)

A line segment can be represented by 2 points, so 4 reals

A circle (or disk) requires 3 reals to store it (center point and radius)

$$y = m \cdot x + c$$

$$\{false, m, c\}$$

$$x = c$$

$$\{true, -, c\}$$

Description size

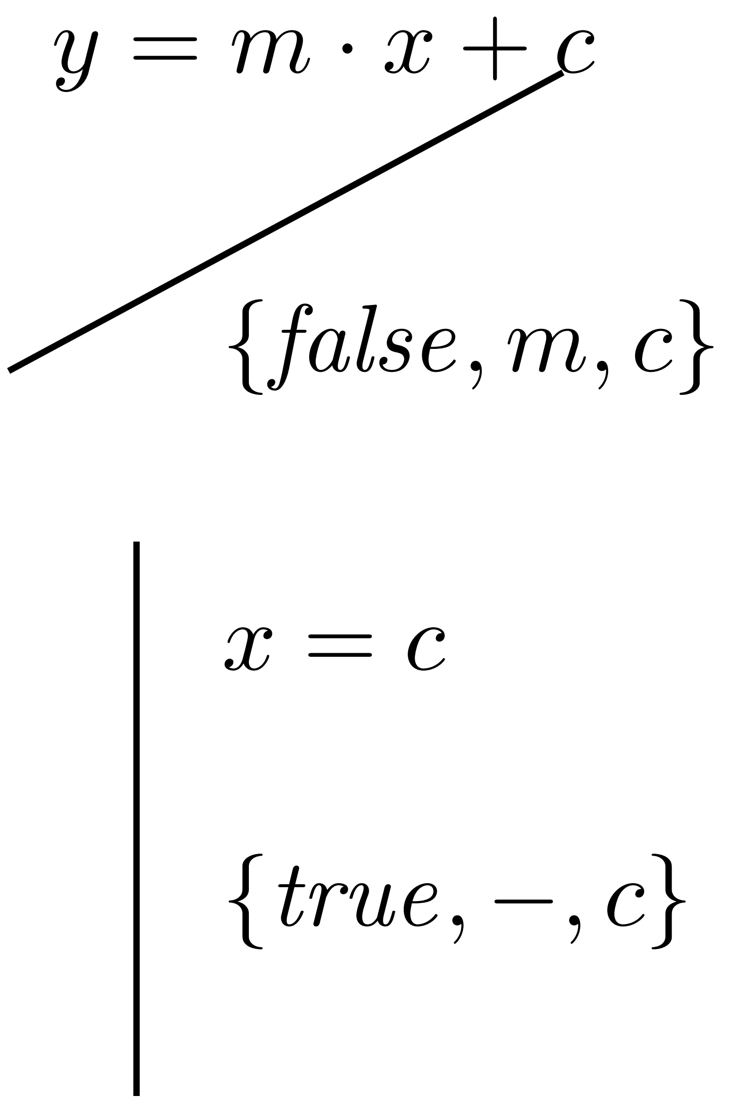
A point in the plane can be represented using 2 reals

A line in the plane can be represented using 2 reals and a Boolean (for example)

A line segment can be represented by 2 points, so 4 reals

A circle (or disk) requires 3 reals to store it (center point and radius)

An axis-aligned rectangle requires 4 reals to store it



Description size and computation time

A simple polygon in the plane can be represented using $2n$ reals if it has n vertices (and necessarily, n edges)

A set of n points requires $2n$ reals

A set of n line segments requires $4n$ reals

Description size and computation time

A simple polygon in the plane can be represented using $2n$ reals if it has n vertices (and necessarily, n edges)

A set of n points requires $2n$ reals

A set of n line segments requires $4n$ reals

A point, line, circle, $\ldots$ requires $O(1)$, or constant, storage

A simple polygon with n vertices requires $O(n)$, or linear, storage

We assume: any computation (distance, intersection) on two objects of $O(1)$ description size takes $O(1)$ time!

Algorithms, efficiency

Recall from your undergraduate algorithms and data structures course:

- a set of n real numbers can be sorted in $O(n \log n)$ time
- a set of n real numbers can be stored in a data structure that uses $O(n)$ storage and that allows searching, insertion, and deletion in $O(\log n)$ time per operation

Algorithms, efficiency

Recall from your undergraduate algorithms and data structures course:

- a set of n real numbers can be sorted in $O(n \log n)$ time
- a set of n real numbers can be stored in a data structure that uses $O(n)$ storage and that allows searching, insertion, and deletion in $O(\log n)$ time per operation

These are fundamental results in 1-dimensional computational geometry!