UNIVERSITY OF DENVER

Lecture Overview

- Four common representations:
  - Grids
  - Waypoint graphs
  - Navigation meshes
  - Delaunay Triangulation

For each representation

- Localization
  - How do you find where you are in the representation?
- Generation
  - How is the representation generated?
- Dynamic changes
  - How are dynamic changes handled?
- Planning
  - How are possible actions computed?
- Memory
  - How much memory is required?

UNIVERSITY OF DENVER

Grids

- Very simple method to implement
  - Partially depends on your world representation
- Used by Dragon Age series
- Used in early Starcraft/Warcraft games
- May not be able to represent 2.5/3d worlds

Waypoint graphs

- Set down pathfinding "nodes" in the world
  - Can only travel between these nodes
  - Referred to as "Dirichlet domains" in the book
  - Also points of visibility

See also: http://www.ai-blog.net/archives/000152.html

Navigation mesh

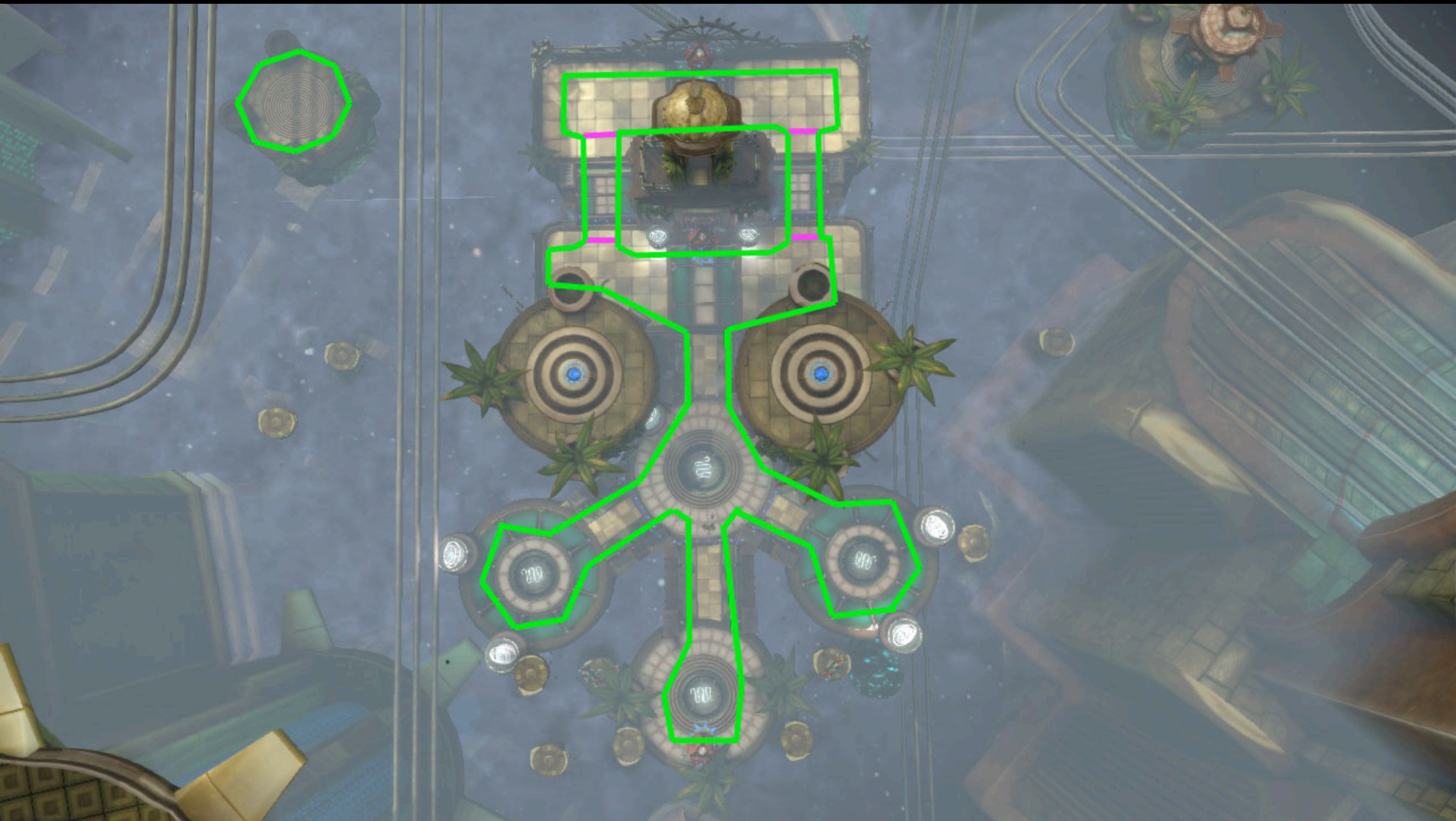
- Break the world up into convex* polygons
- All area inside a polygon is considered passable

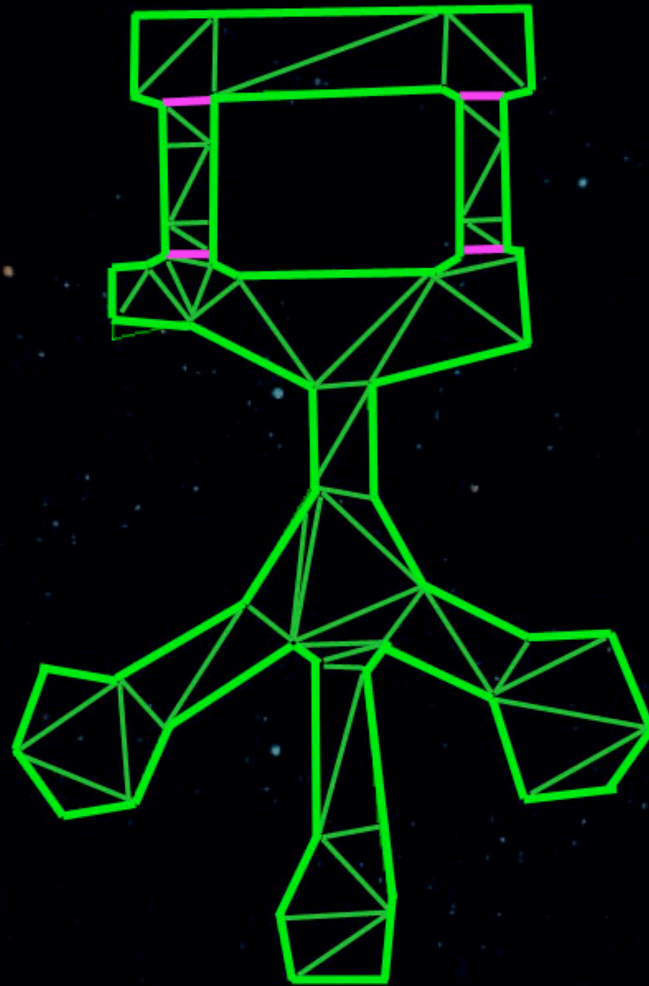
See also: http://www.ai-blog.net/archives/000152.html

PlayStation Move Heroes

PlayStation Move Heroes

Terrain Representation





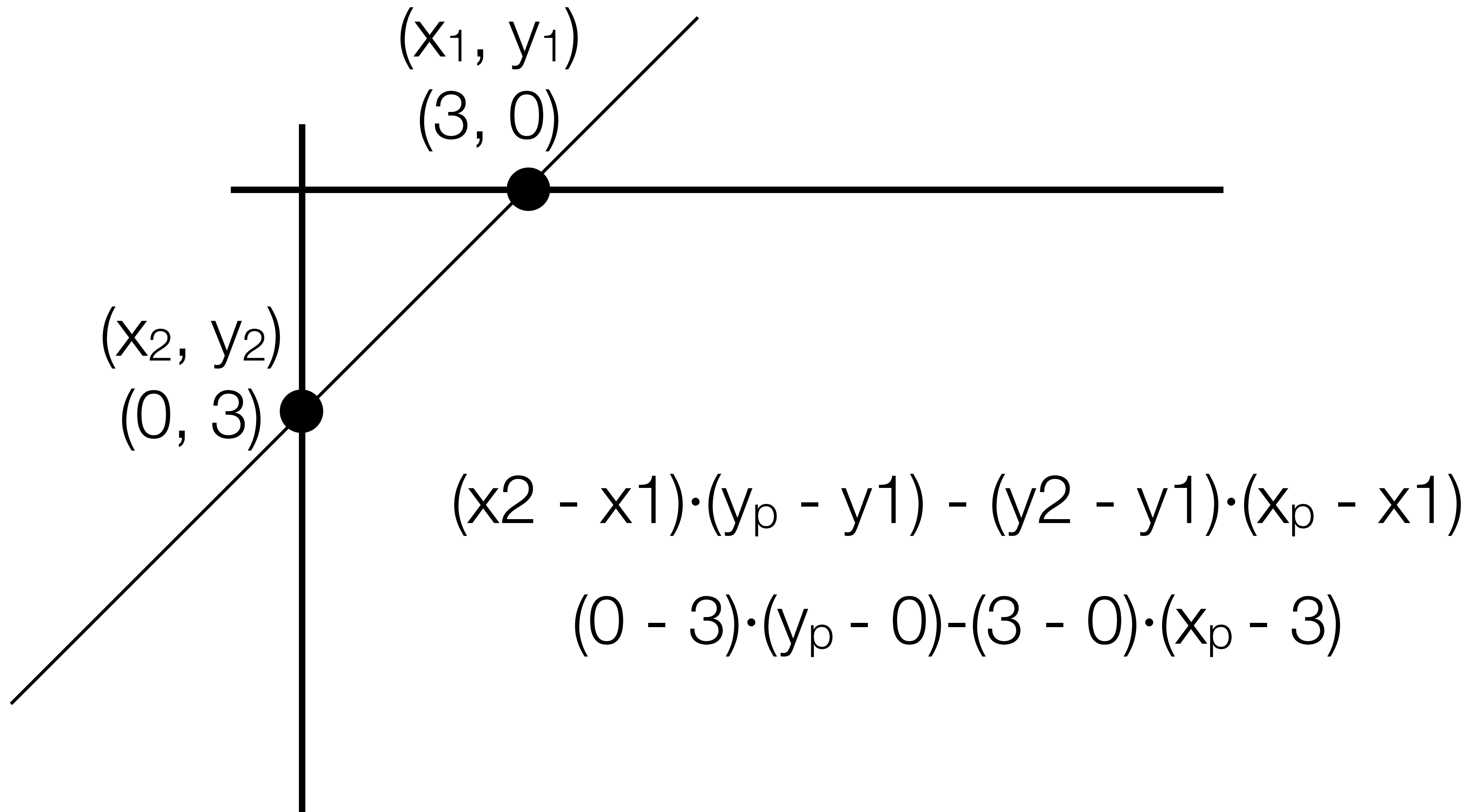
Navigation mesh: Localization

- If you know where everyone starts, it isn't (too) hard to update positions
  - See the book for a bit about this
- But, pathfinding requires moving to a location selected at "random" by the user playing the game
  - Must be able to localize the mesh containing that point quickly
- One of the harder/more expensive tasks for nav meshes

Localization

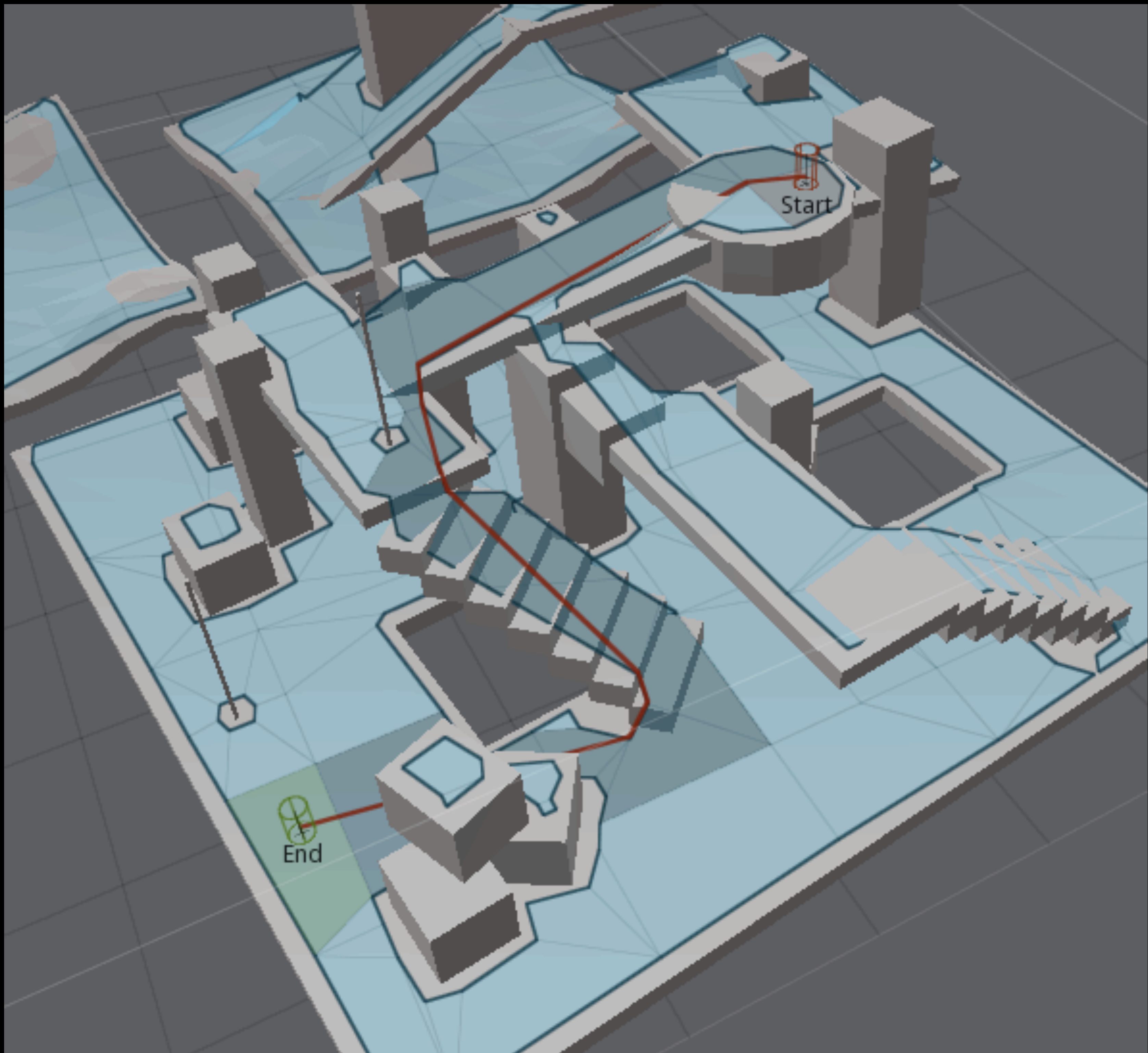
- Assume you have a line in the mesh between points:
  - (x_1, y_1) and (x_2, y_2)
- And a point: (x_p, y_p)
- Compute: $(x_2 - x_1) \cdot (y_p - y_1) - (y_2 - y_1) \cdot (x_p - x_1)$
- If:
  - result < 0, inside line
  - result $= 0$, on line
  - result > 0, outside line

Localization example



Navigation mesh: Generation

- Nav. meshes can be put down by hand by designers
  - Common approach
- Many middleware tools exist to auto-build nav meshes
  - Recast is open source
  - http://code.google.com/p/recastnavigation/


Start
End

Navigation mesh: Dynamic changes

- Difficult to modify mesh
  - Not worth inserting moving objects
- Must intersect polygons
- See also:
  - http://digestingduck.blogspot.com/2011/03/temporary-obstacle-processing-overview.html
  - Works on this essentially full time

Navigation mesh: Planning

- Possible actions:
    - Move across each edge of the polygon
    - Difficulty depends on the number of edges of each polygon

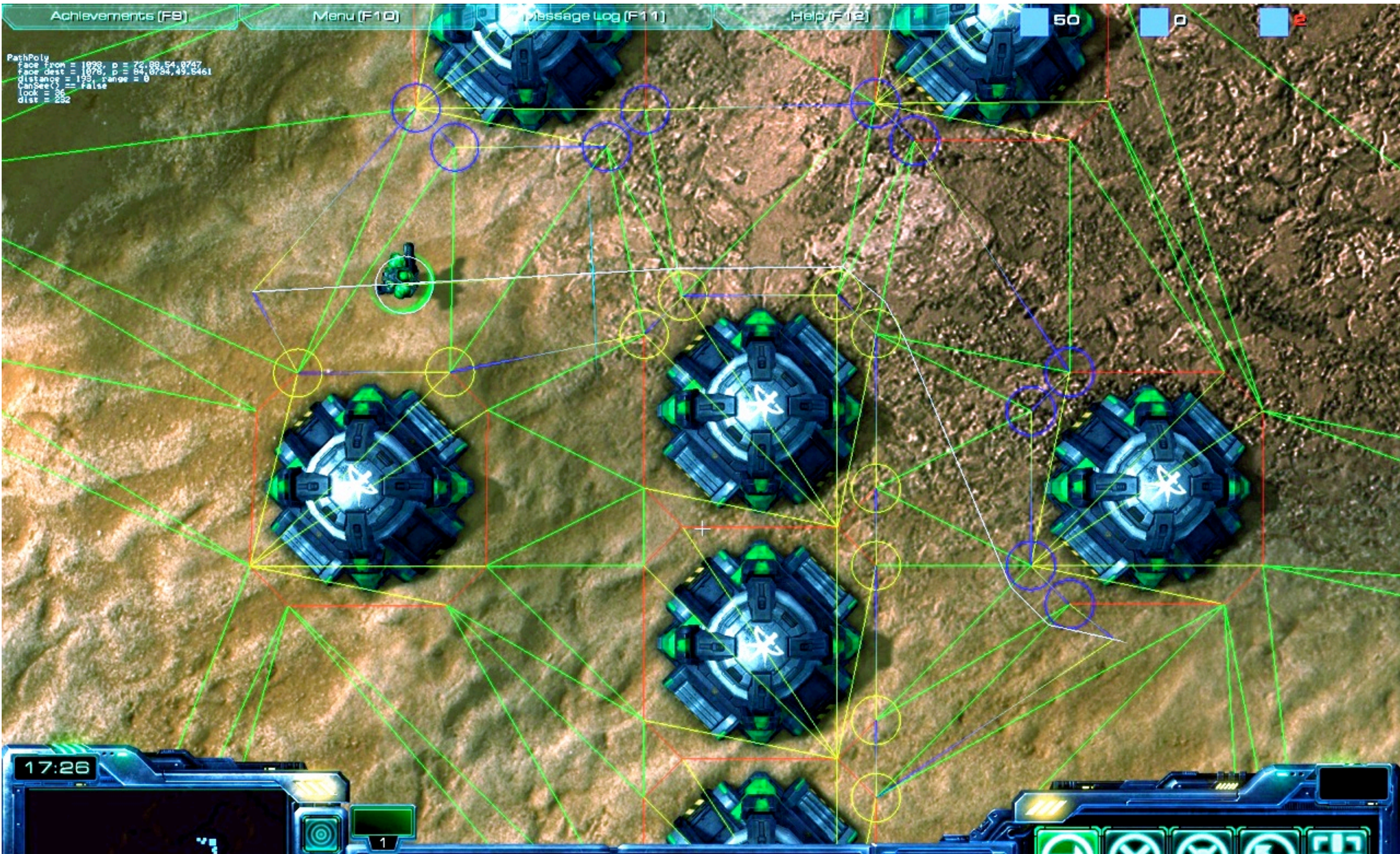
Navigation mesh: Memory

- Representation of the world is very light
  - Large areas represented by a single polygon
  - Less or same data as the level geometry
  - Usually simpler
    - Movement confined to flat surfaces
  - No texture data
- More than required for waypoint graph

Constrained Delaunay Triangulation

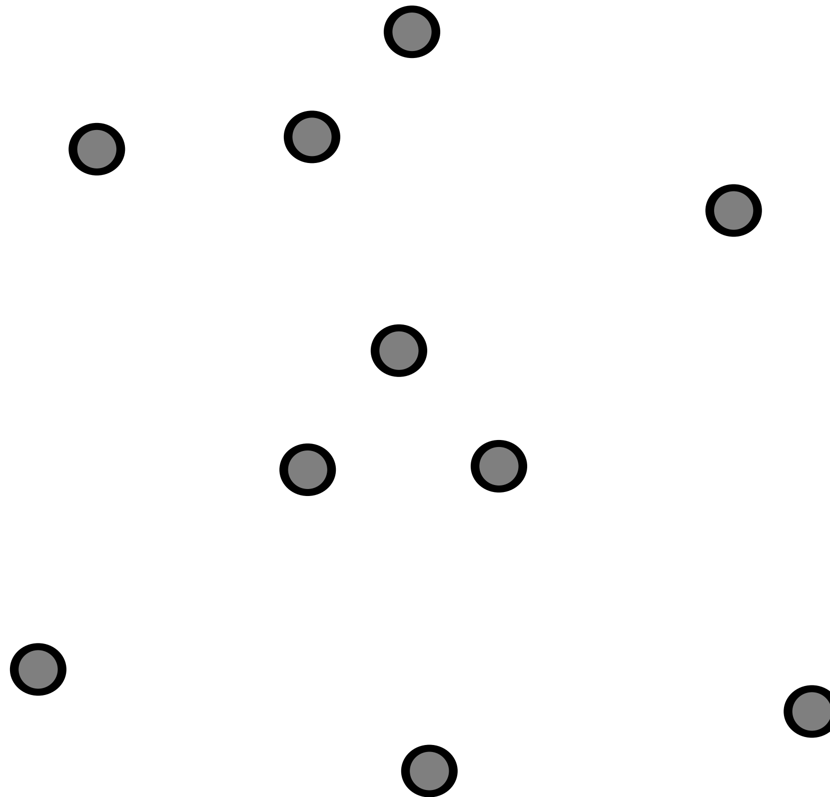
- Somewhat like a navmesh
  - Use exclusively triangles, not polygons
- Maintain the properties of the Delaunay Triangulation
  - Triangulation should not be long and skinny
- Constrained, because not all edges can be controlled

Starcraft II CDT



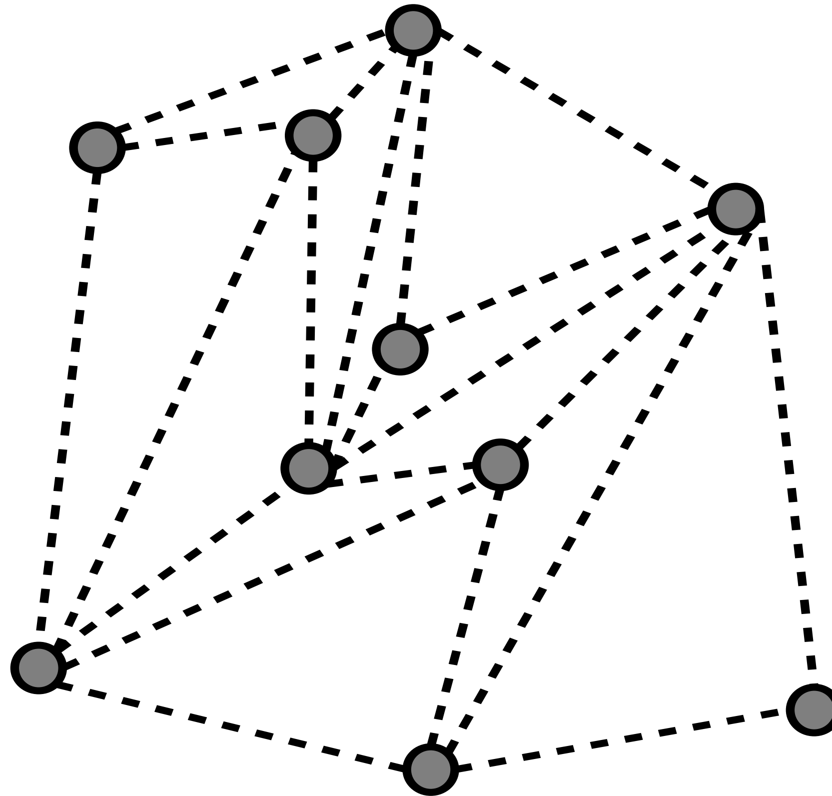
UNIVERSITY OF DENVER

Point set

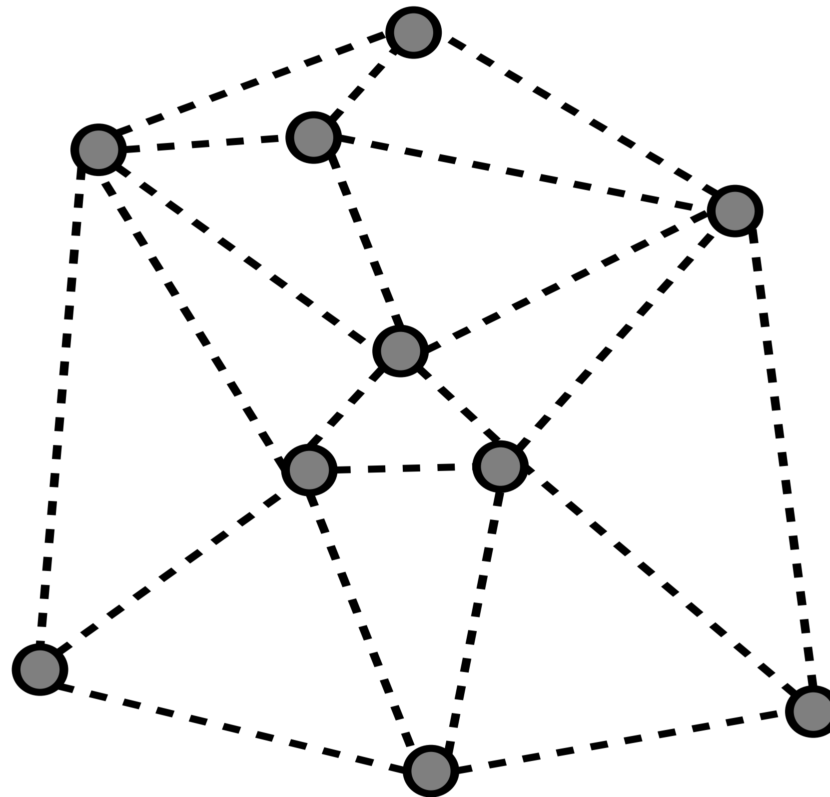


Images from Douglas Deymen's thesis: http://skatgame.net/mburo/ps/thesis_demyen_2006.pdf

Triangulation

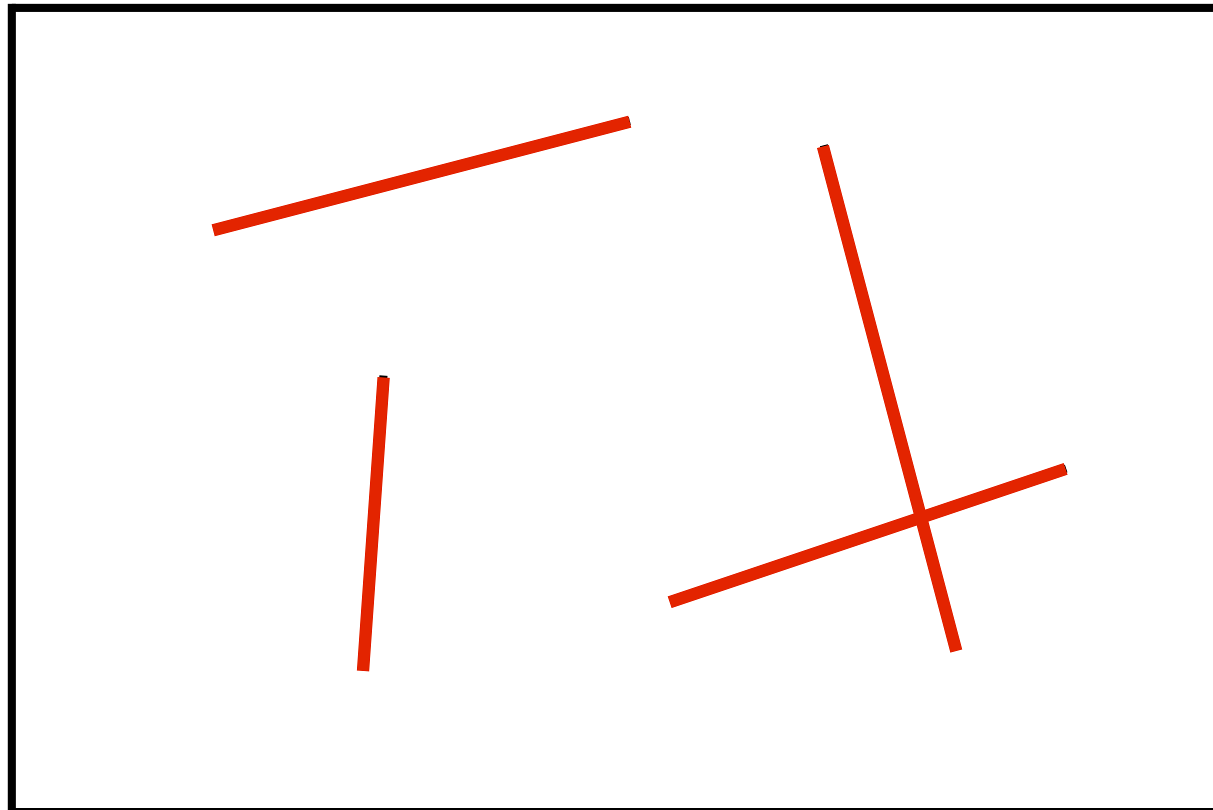


Delaunay Triangulation

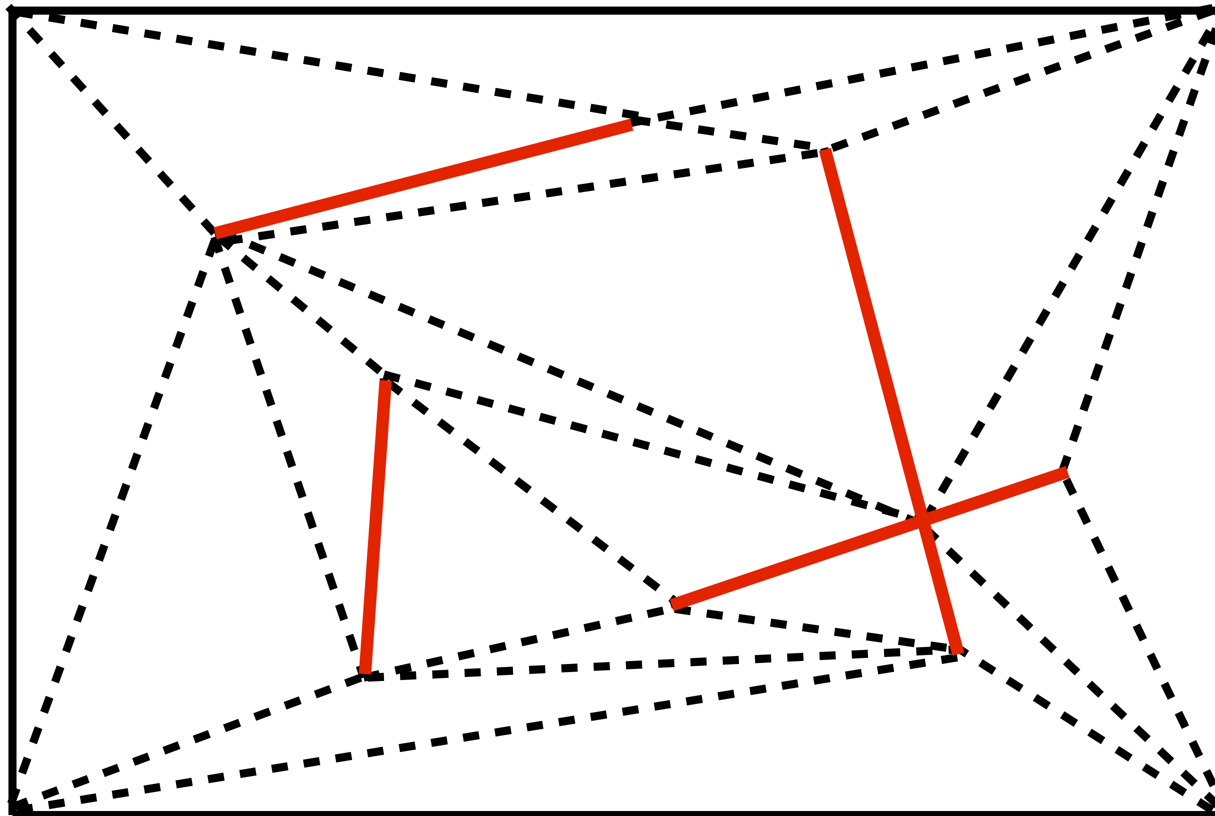


UNIVERSITY OF DENVER

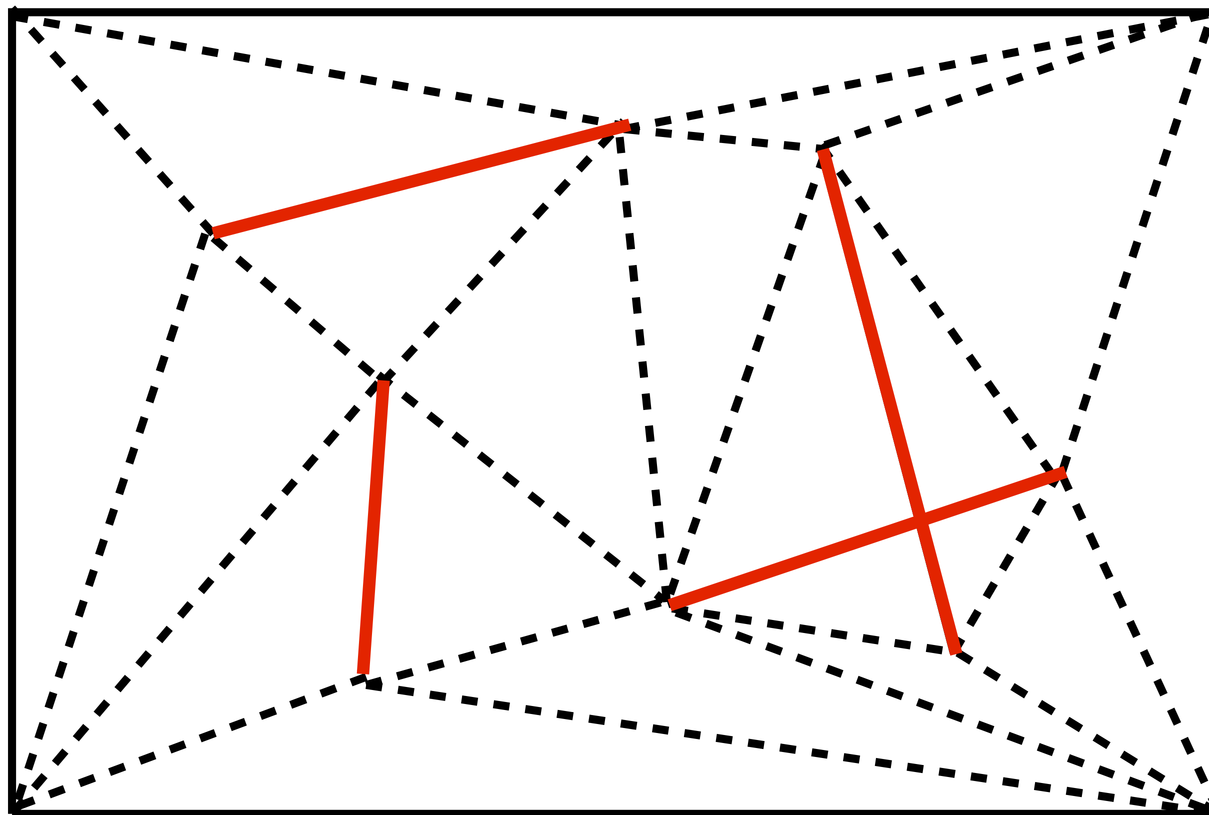
Line segments



Triangulation



Delaunay Triangulation



Geometry

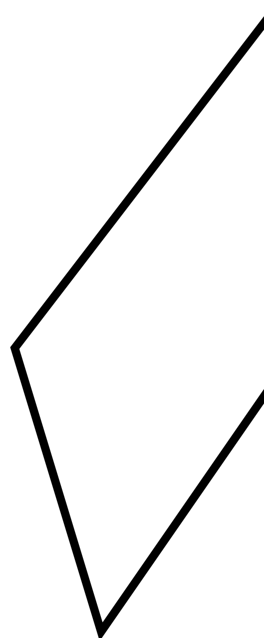
- Sum of interior angles in a triangle: 180°
- Sum of interior angles in a quadrilateral: 360°
  - A quadrilateral contains 2 triangles

UNIVERSITY OF DENVER

Delaunay Property

- Given two adjacent triangles, which form a quadrilateral
  - The quadrilateral should be divided through the two opposite angles which have the largest sum (≥ 180°)

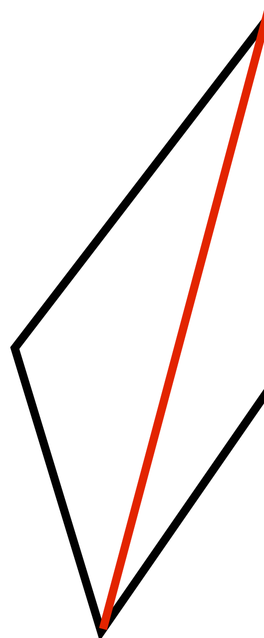
Initial quadrilateral



Delaunay Property

- Given two adjacent triangles, which form a quadrilateral
  - The quadrilateral should be divided through the two opposite angles which have the largest sum (≥ 180°)

Triangulation

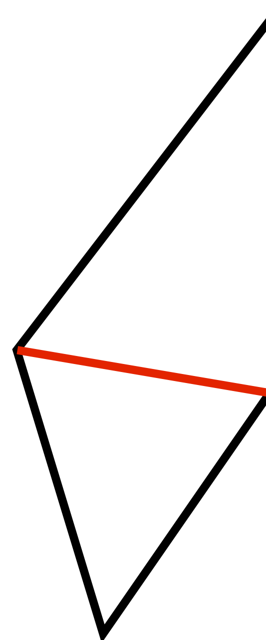


UNIVERSITY OF DENVER

Delaunay Property

- Given two adjacent triangles, which form a quadrilateral
  - The quadrilateral should be divided through the two opposite angles which have the largest sum (≥ 180°)

Delaunay triangulation



Delaunay Triangulation: Localization

- Same as nav mesh
  - Can use similar tricks as waypoint graph to speed the process up

Delaunay Triangulation: Generation

- Relatively simple:
  - Add constrained edges for every obstacle
  - Form triangulation
  - Swap edges in quadrilateral to turn into Delaunay triangulation
    - Can be expensive; there are faster approaches
- But; algorithms can be numerically unstable

Delaunay Triangulation: Dynamic changes

- Easier to change than with nav meshes
- eg insert new constrained edge
    - Add new points if crosses existing constrained edge
    - Remove edges that cross constrained edge
    - Re-triangulate with delaunay property

Delaunay Triangulation: Planning

- Assuming you entered from one edge of triangle:
  - Can exit from two possible edges
  - Cannot exit from constrained edges

Delaunay Triangulation: Memory

- More triangles stored than with nav mesh
    - Although nav meshes are sometimes stored as multiple triangles
- Fairly efficient representation of the world unless obstacles have small jagged edges
    - eg taken from a rasterized grid

UNIVERSITY OF DENVER

Summary

	Localization	Dynamic Changes	Planning Speed	Memory
Grids	easy/fast	easy	slow	high
Waypoint Graph	harder	somewhat easy	fast	lower
Nav Mesh	harder	harder	fast	lower
Delaunay Triangulation	harder	hard	fast	lower

UNIVERSITY OF DENVER

Simple smoothing

- Any path can be represented by line segments
- What algorithms can be performed on line segments to shorten paths that are too long

More complex smoothing

- Funnel Algorithm
  - When a path is defined by a sequence of polygons, the funnel algorithm finds the optimal path through
  - For each additional segment, optionally joint the new points with the previous candidate line segments

Funnel Algorithm

- Pros:
  - Produces high-quality paths
  - No more difficult than many other approaches
- Cons:
  - Need representation amenable to funnels
  - Length of path may be significantly different than when planning