GLOBAL NAVIGATION

GLOBAL NAVIGATION

- Examples
  - http://www.youtube.com/watch?v=ABJjdpxeMtE&noredirect=1
  - http://www.youtube.com/watch?v=tro-fjsBs9g

ENVIRONMENT REPRESENTATION

GLOBAL NAVIGATION

- Navigation in an environment where local navigation techniques are insufficient
  - “Local”
    - Walk straight to goal
    - Always turn such that direction is most toward goal as possible
  - Local Minima
    - Local techniques can lead to globally inefficient choices

ENVIRONMENT REPRESENTATION

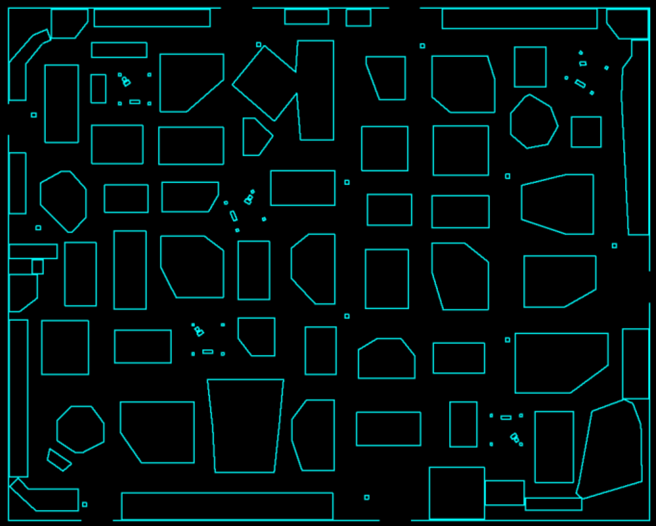
- Visual representation more detailed than necessary
  - Very common for dynamics simulation
  - Typically true for navigation as well
- The more complex the representation, the more expensive

ENVIRONMENT REPRESENTATION

- Full 3D polygonal representation
  - Quite expensive
  - Details smaller than ~0.2 m probably don't matter.
  - Floor plan matters more than vertical space
    - (vertical clearance)

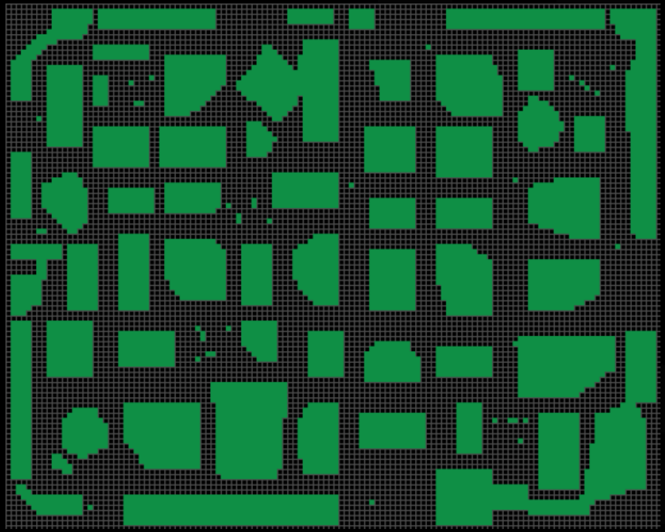
ENVIRONMENT REPRESENTATION

- 2D footprint
  - Saving an entire dimension
  - How much detail?
    - Coarse bounding volumes
      - Visually clear regions are no longer clear



ENVIRONMENT REPRESENTATION

- Keep polygons or rasterize to grid?
  - Grid offers simple “is colliding” query
  - (Compatible with potential field methods)



GLOBAL NAVIGATION

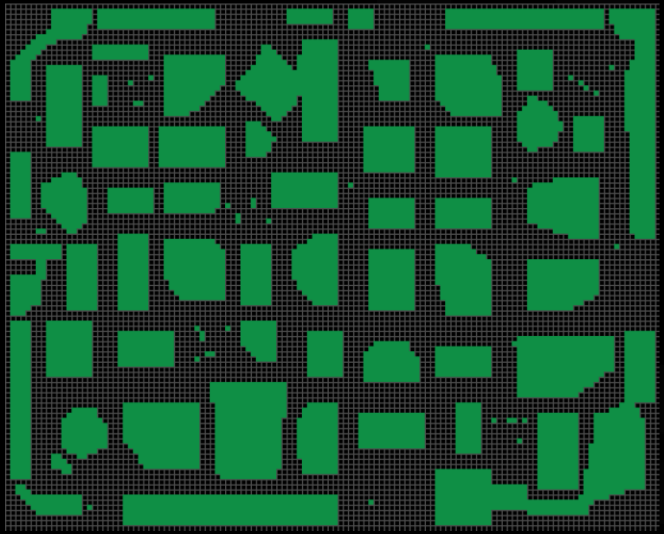
- Solving requires two things
  - Represent the navigable space and its relationships
  - Search the navigable space for optimal paths

NAVIGATION GRID

- Various names
  - Guidance field
  - Potential field

NAVIGATION GRID - DEFINITION

- Discretization of space
  - Cells don’t have to be uniform or square
    - Rectangle, hex, etc.
  - Cells are either marked as free or occupied
    - Non-boolean values possible



NAVIGATION GRID - USAGE

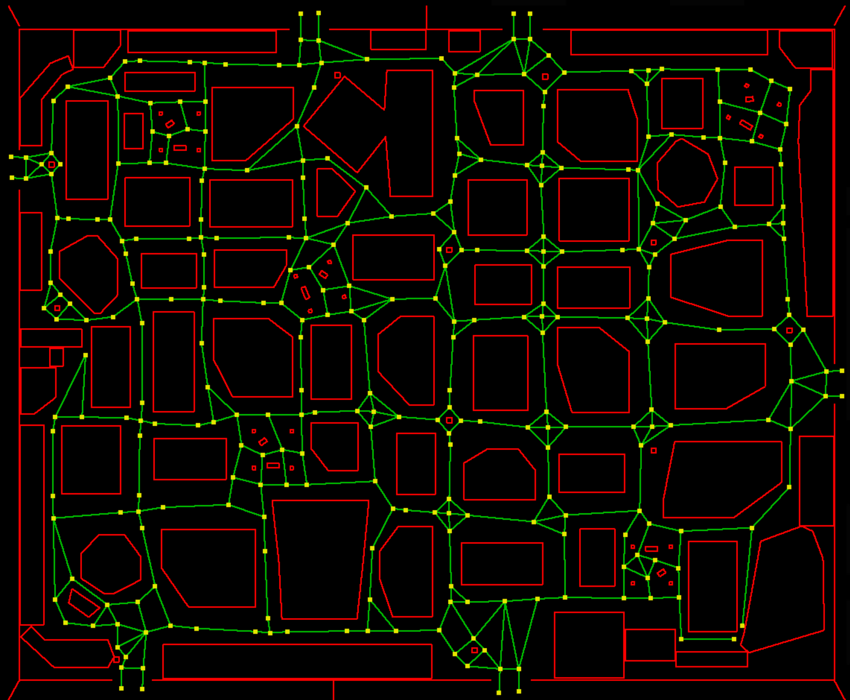
- Select a goal point
- Each cell contains the direction of travel along the *shortest* path from that cell to the goal point
- Compute:
  - Compute shortest path distance to goal from each cell center
    - Solve using front propagation algorithms
      - (e.g. https://www.ceremade.dauphine.fr/~peyre/teaching/manifold/tp2.html)
  - Compute gradient of the field – gradient is the direction of the shortest path

NAVIGATION GRID - ANALYSIS

- Pros
  - O(1) preferred direction computation
    - (even with bi-linear interpolation of the grid)
- Cons
  - Expensive creation
    - Pre-computation or created by hand
  - Suffers from discretization errors
  - One field per goal
  - Requires planar topology – can’t walk over and under a bridge

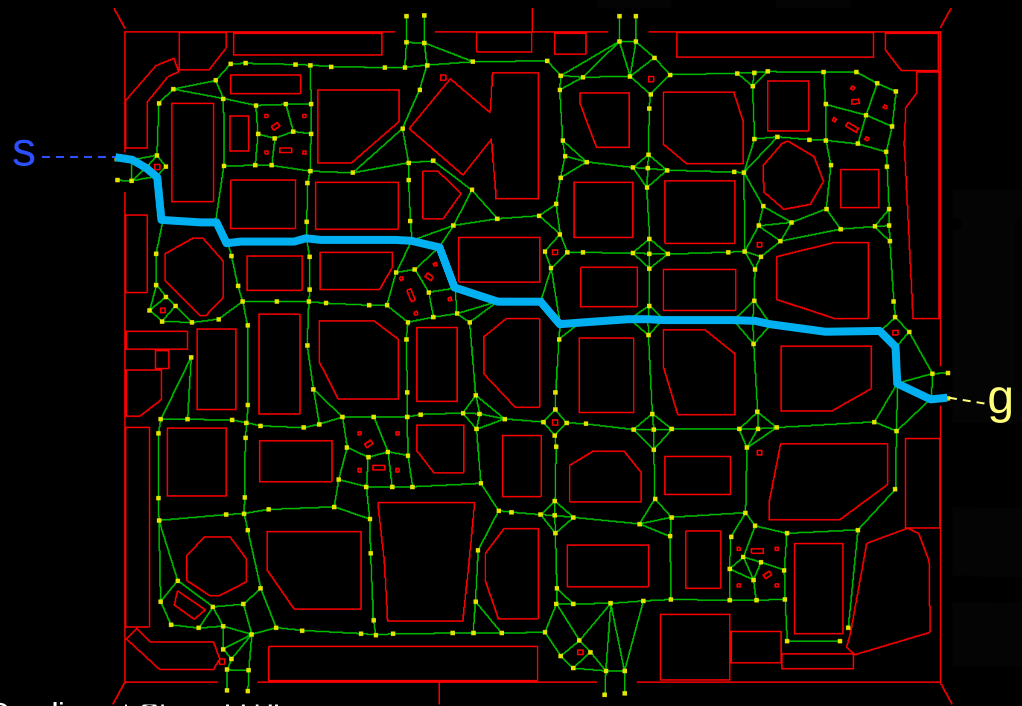
ROAD MAP - DEFINITION

- A discrete *sampling* of free space
- Each sample is guaranteed to be collision free
- Links between samples is guaranteed to be a collision free trajectory



ROAD MAP - USE

- Given start (s) and goal (g) positions
  - Link to roadmap
  - Find path on roadmap



ROAD MAP - USE

- Path
  - $P = [p_1, p_2, p_3, \ldots, p_n, g]$
    - Ordered list of waypoints
  - Preferred direction is direction toward "next" waypoint – the *target* waypoint
  - When do you change which waypoint is the target waypoint?
  - What if the target waypoint is lost?

ROAD MAP - USE

- When do you advance the target waypoint?
  - Simply measure distance (d) – d < D → reached
    - D – threshold
      - Big enough to be robust
      - Small enough that the next waypoint is reachable
  - What if the crowd keeps me from reaching the waypoint?
  - What if the crowd sweeps me PAST the waypoint along my path, but I don’t get close?

ROAD MAP - USE

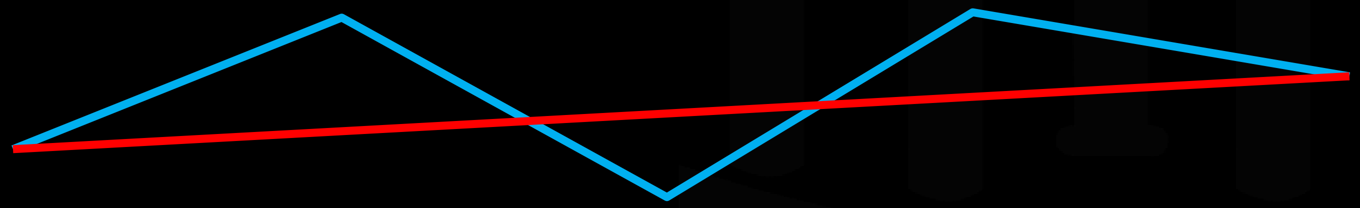
- When do you advance the target waypoint?
  - Visibility tests
    - Set the target waypoint to be the most advanced waypoint that is *visible*
    - This keeps the waypoint as far in “front” as possible
    - Also detects if the agent is pushed from the path

ROAD MAP - USE

- What if you lose sight of the target waypoint (pushed off the path)?
  - Replan
    - Create a new path
  - Rewind
    - Try testing previous waypoints (or successive)
    - Replan if all else fails
  - Remember
    - Remember where you were when you last could see it and work toward that

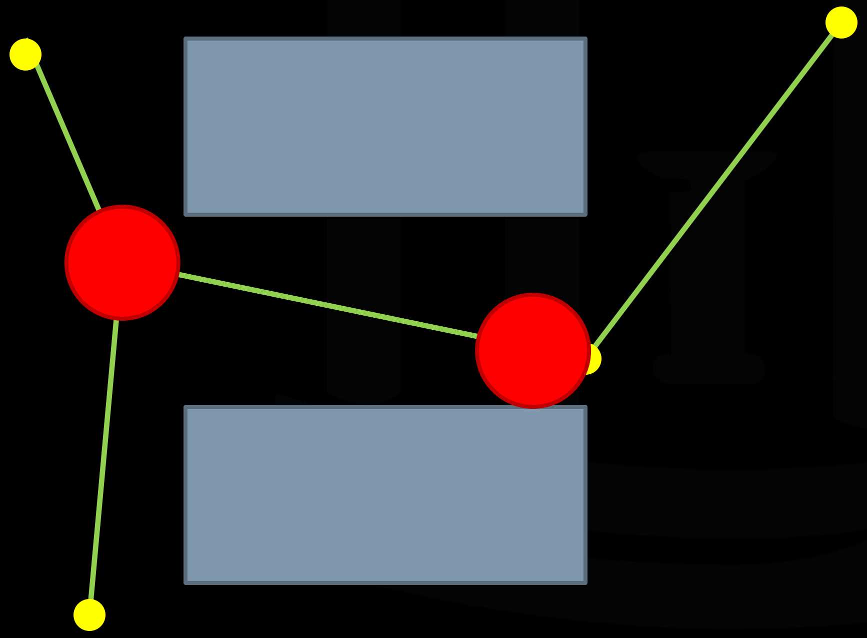
ROAD MAP - ANALYSIS

- Paths are dependent on sampling and connectivity
  - Path is only "optimal" w.r.t. the graph – not the environment
  - "Smoothing" the path helps
  - Earlier visibility query implicitly smooths the path
    - All but the last visible nodes are culled



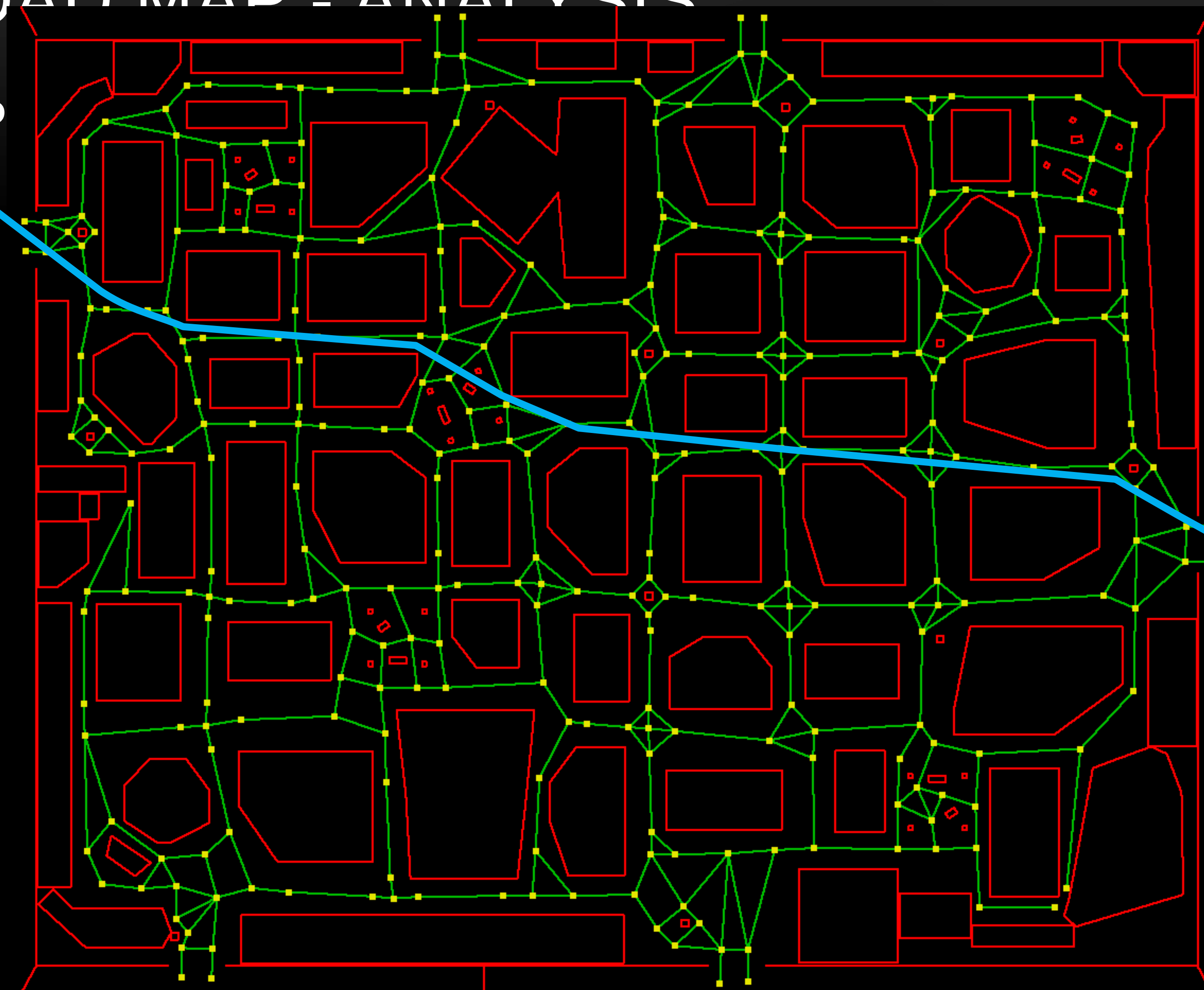
ROAD MAP - ANALYSIS

- That form of smoothness depends on the roadmap



ROAD MAP - ANALYSIS

- P ty
  - se
    h
  -



ROAD MAP - ANALYSIS

- Clearance
  - Roadmaps are computed with one clearance in mind
    - What if there are entities of varying size?
    - Big agents will attempt to travel links with insufficient clearance on a small-agent map
    - Small agents will skip valid paths when using big-agent maps
  - Encode each link with maximum clearance

ROAD MAP - ANALYSIS

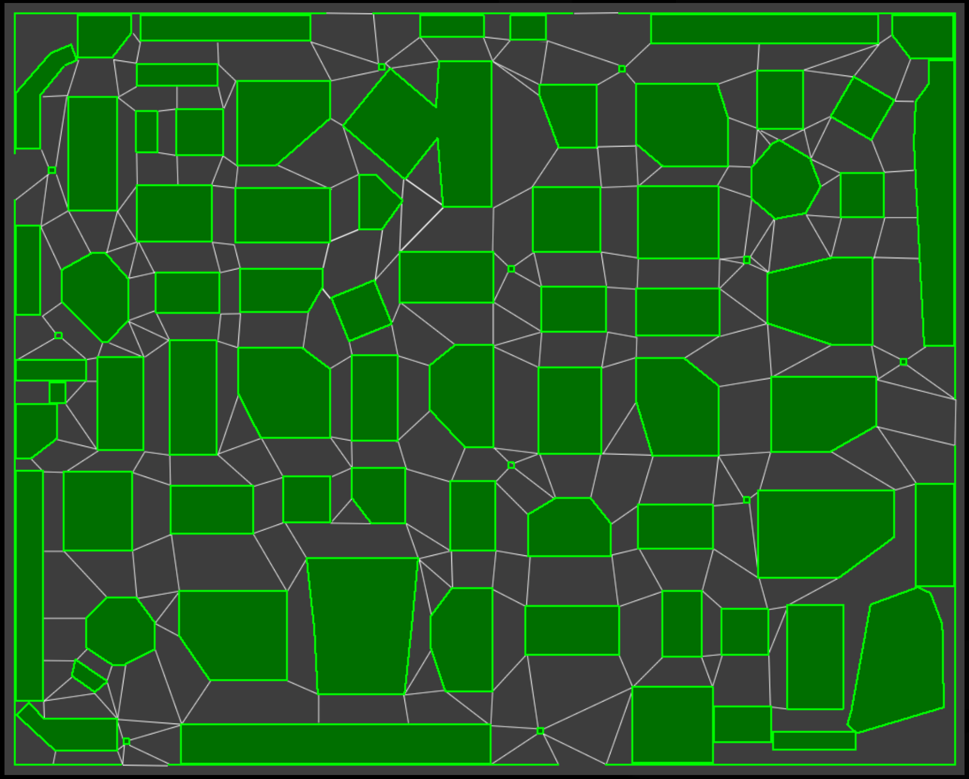
- More choices → more complexity
  - The only way to give agents more paths to reach their goal is to increase the complexity of the map
  - Search algorithms are worse than linear in the length of the optimal path (length = # of links)
    - Double the # of links, more than double the computation time
  - Also increase memory footprint

ROAD MAP - ANALYSIS

- Pros
  - Easy to create
  - Graph search straight-forward and generally effective
  - Pre-computed
  - Allows for non-planar topologies
- Cons
  - Hard to create a *good* roadmap
  - Paths non-optimal and non-smooth
  - Requires acceleration structure and visibility query to link to the graph

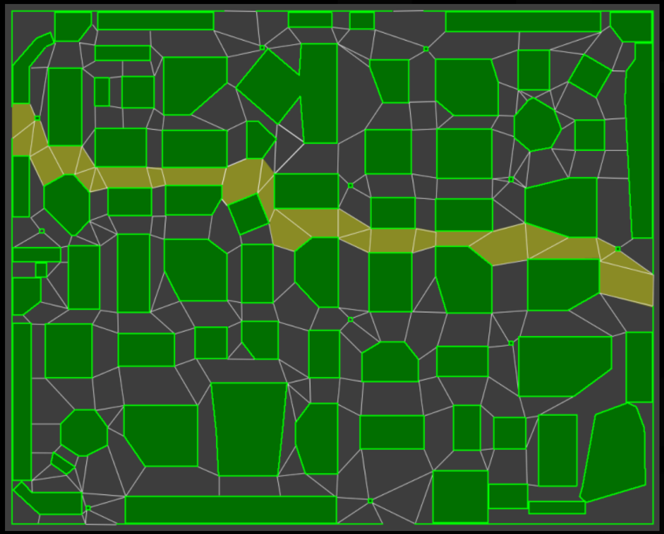
NAVIGATION MESH - DEFINITION

- Discretization of free region into a mesh of convex polygons



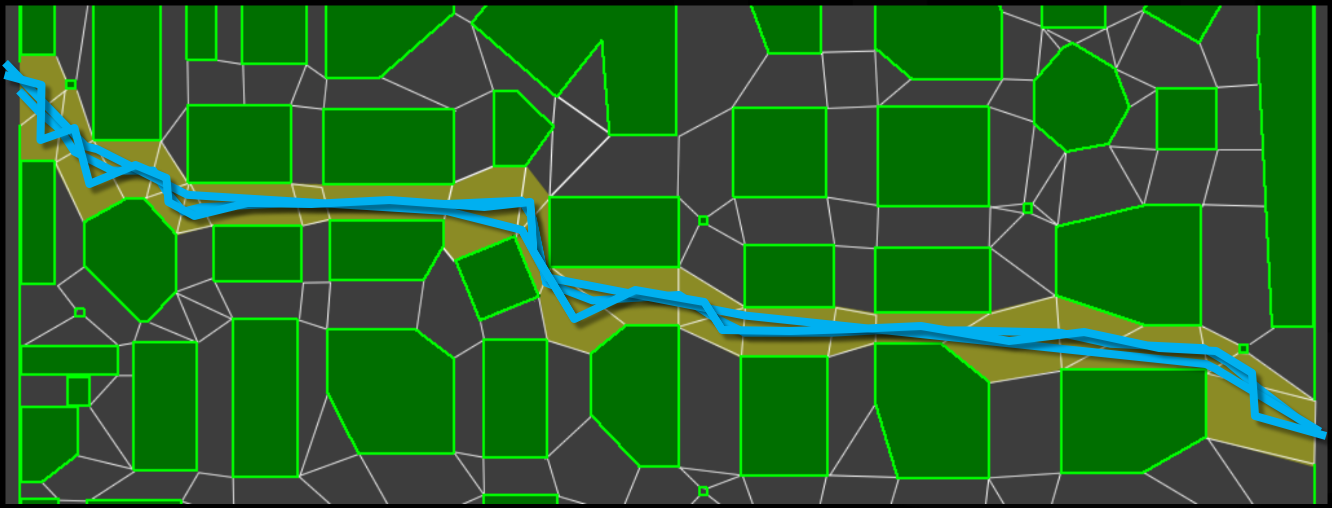
NAVIGATION MESH - USE

- Discretization of free region into a mesh of convex polygons
  - Graph search the mesh for an *envelope*
  - Compute path in the envelope



NAVIGATION MESH - USE

- Envelope Path
  - Centroid path
  - Edge center path
  - “Optimal” path

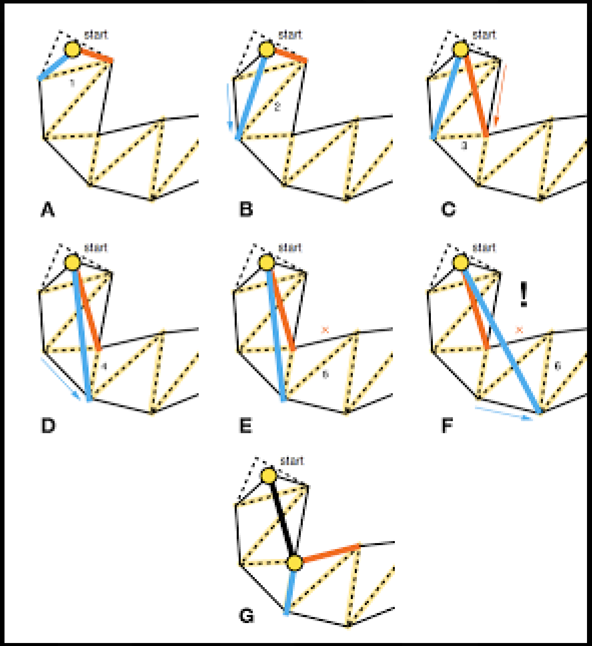


NAVIGATION MESH - USE

- Funnel algorithm (approximate)
  - How we select the “optimal” path

NAVIGATION MESH - USE

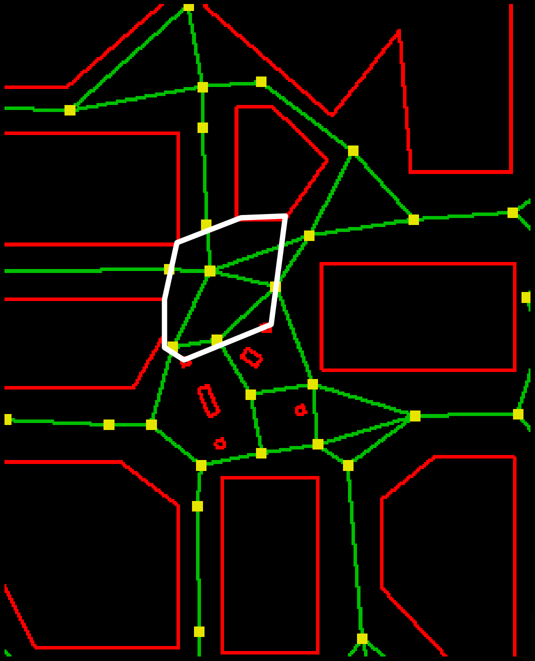
- Define an origin: o
- Define the cone of visibility spanning the first portal
- For each successive portal
  - Contract the funnel
  - If funnel collapses, create a waypoint on that portal vertex
  - Reset the origin to that waypoint

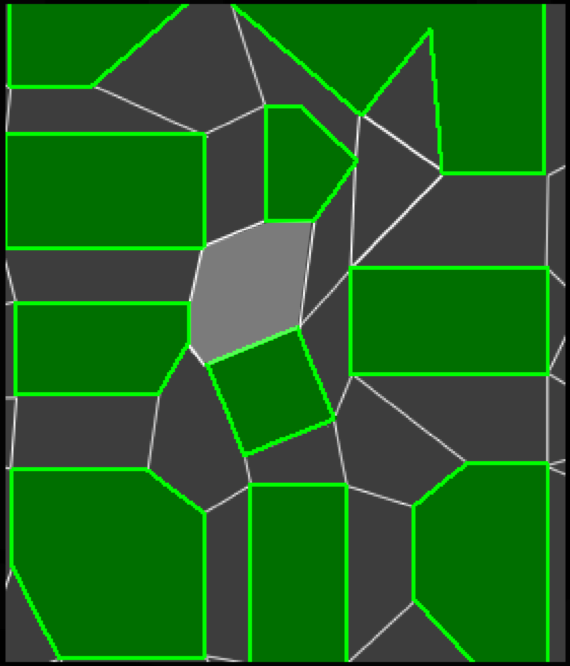


http://cs.brown.edu/courses/cs195u/lectures/06.pdf

NAVIGATION MESH - ANALYSIS

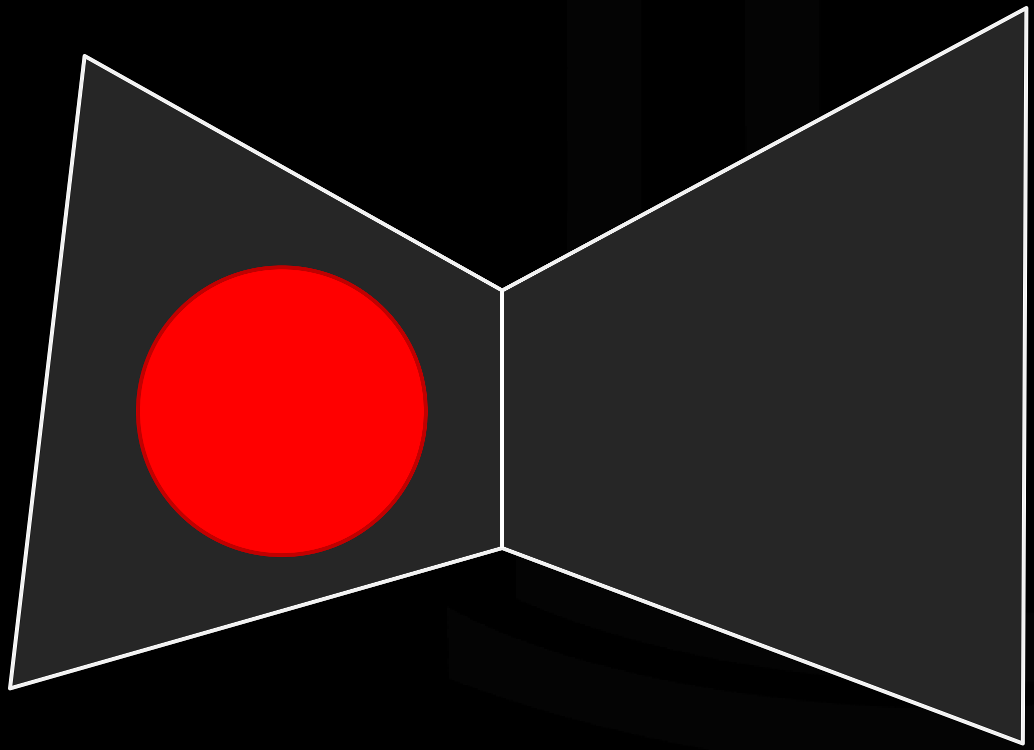
- Implicit connectivity





NAVIGATION MESH - ANALYSIS

- Clearance for range of sizes
  - In the graph – make edge weight depend on clearance

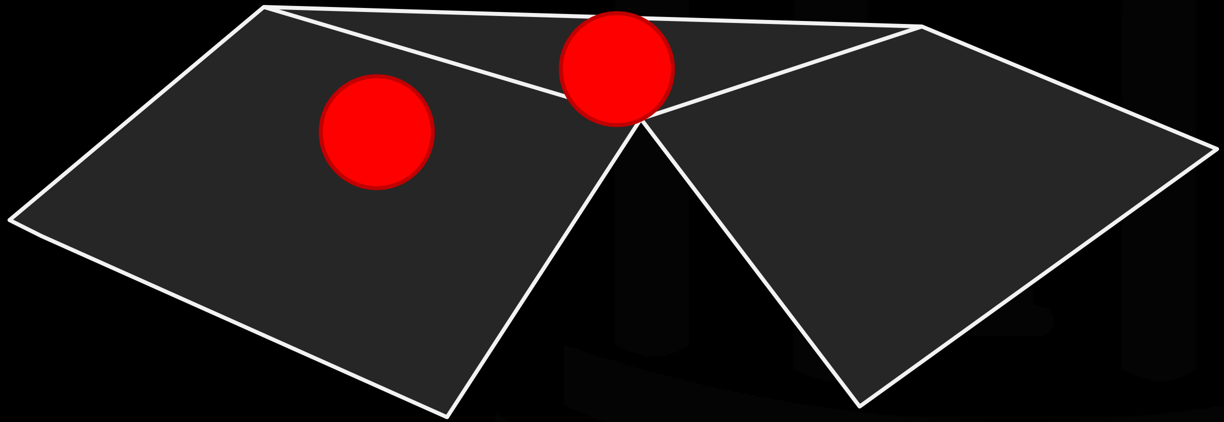


NAVIGATION MESH - ANALYSIS

- Convexity is good
  - Any two points inside a convex polygon are "linkable"
  - Progress easy to track
    - Given target portal, as long as I'm in the polygon, I can move to a point on the portal

NAVIGATION MESH - ANALYSIS

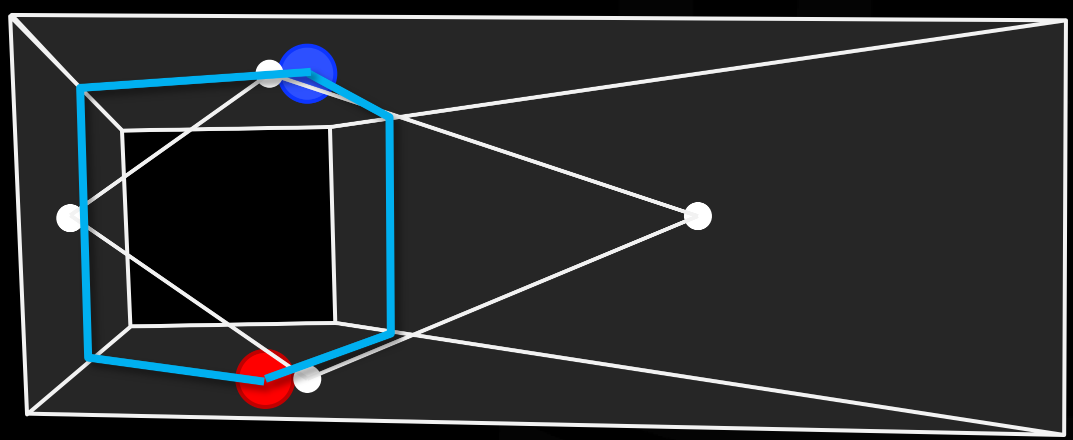
- If the edges are wide enough, is the mesh clear?
  - Not necessarily
  - Further classification needs to be done
  - Clearance *can* depend on which way one travels



“A Generalized Exact Arbitrary Clearance Technique for Navigation Meshes.” R. Oliva, N. Pelechano ACM SIGGRAPH conference on Motion in Games (MIG'2013). November 7-9. Dublin (Ireland). 2013.

NAVIGATION MESH - ANALYSIS

- What is the path distance between two polygons for graph search?
  - Moving from red to blue
  - Correcting this brings back graph density

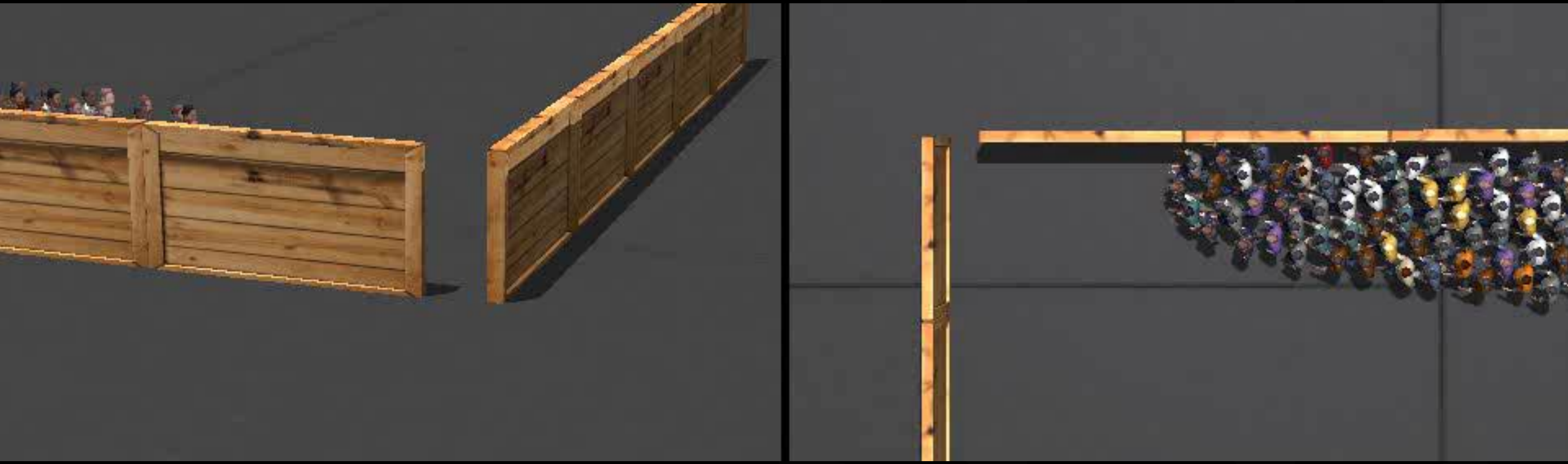


NAVIGATION MESH - ANALYSIS

- Pros
  - Generally more compact than equivalent graphs
  - Envelopes of trajectories encoded
- Cons
  - VERY difficult to produce
  - Properly handling clearance is tricky

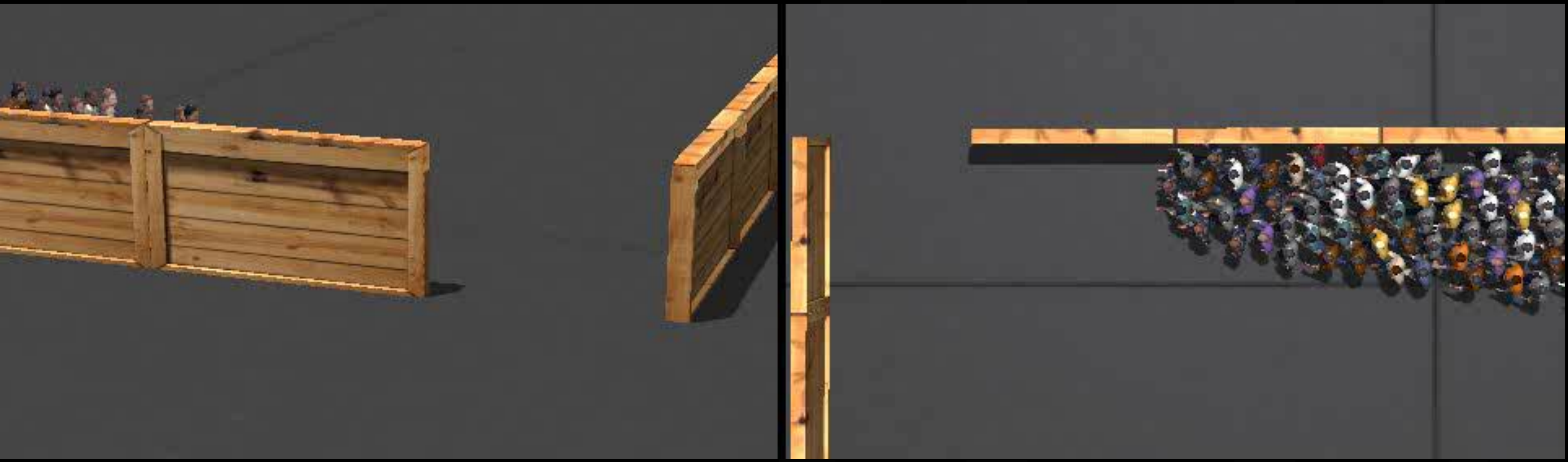
WAYPORTALS

- Narrow passages



WAYPORTALS

- Wide passages

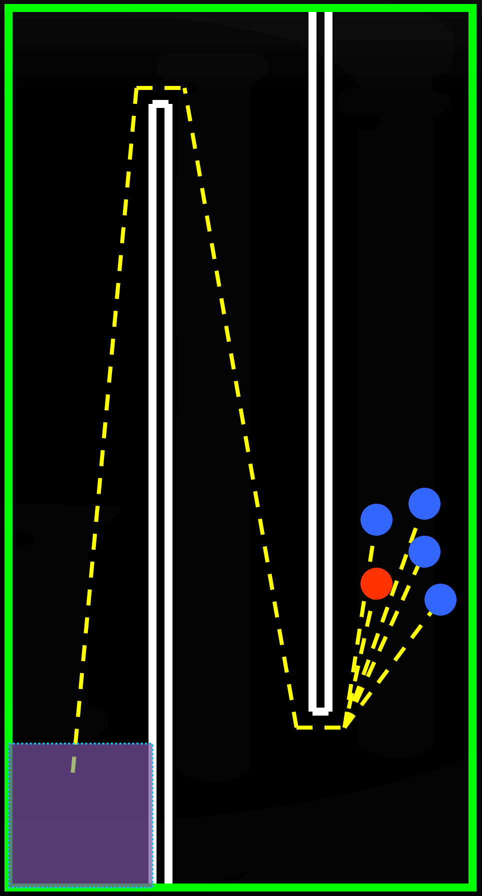


WAYPORTALS

- Wide passages

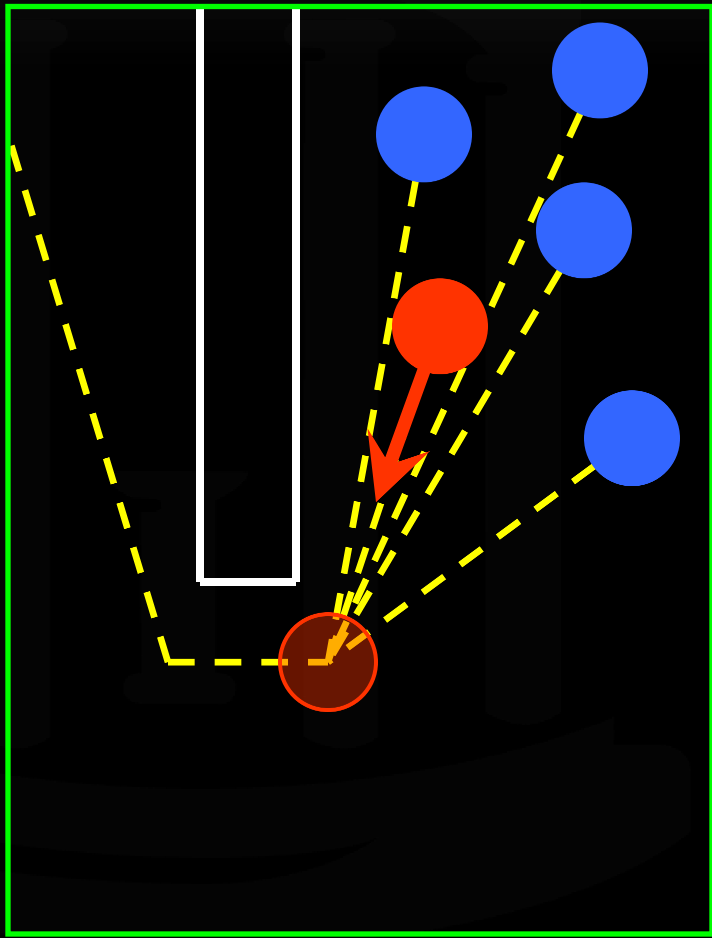
WAYPORTALS

- Global Planning
  - Understands full domain
  - For agent and goal:
    - Find "optimal" path to goal
    - Only consider static obstacles
  - Nearby agents have similar paths



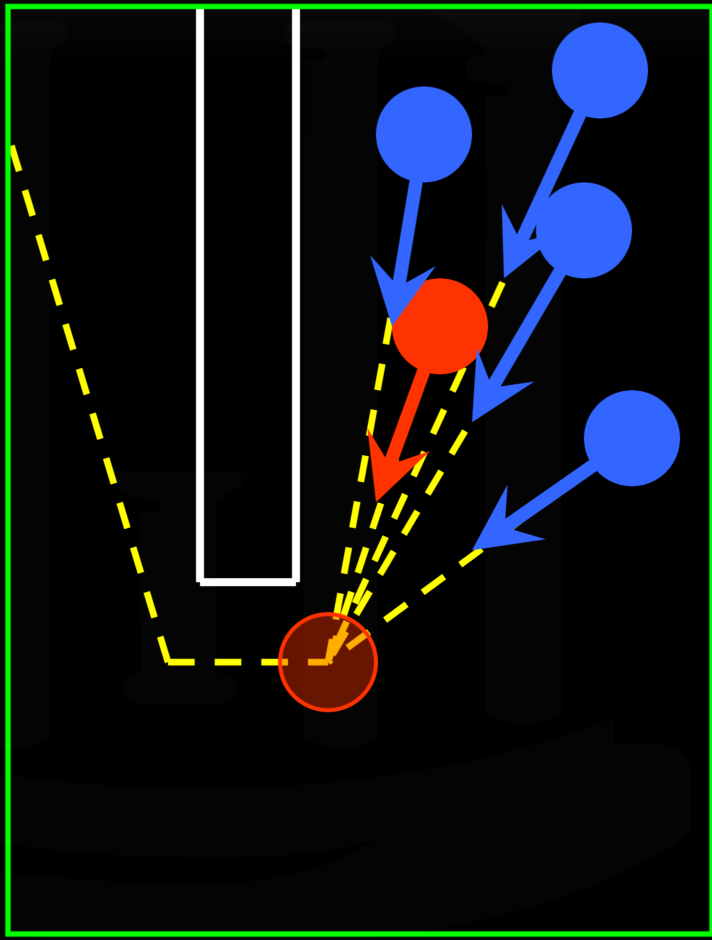
WAYPORTALS

- Local Planning
  - Limited domain knowledge
    - Waypoint
  - Move towards waypoint



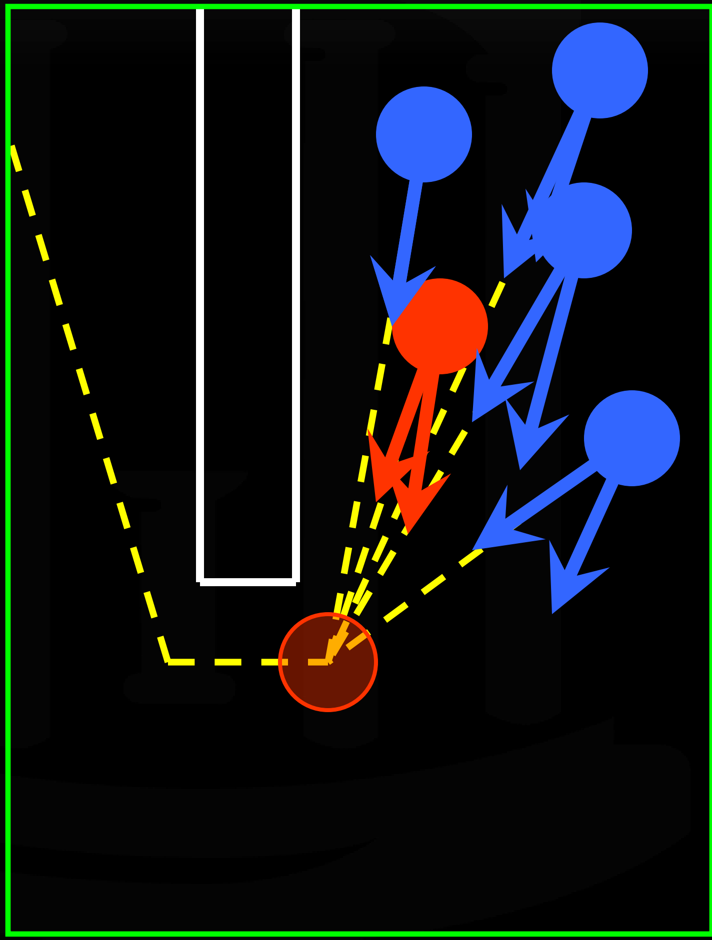
WAYPORTALS

- Local Planning
  - Limited domain knowledge
    - Waypoint
  - Move towards waypoint
  - Avoid collisions



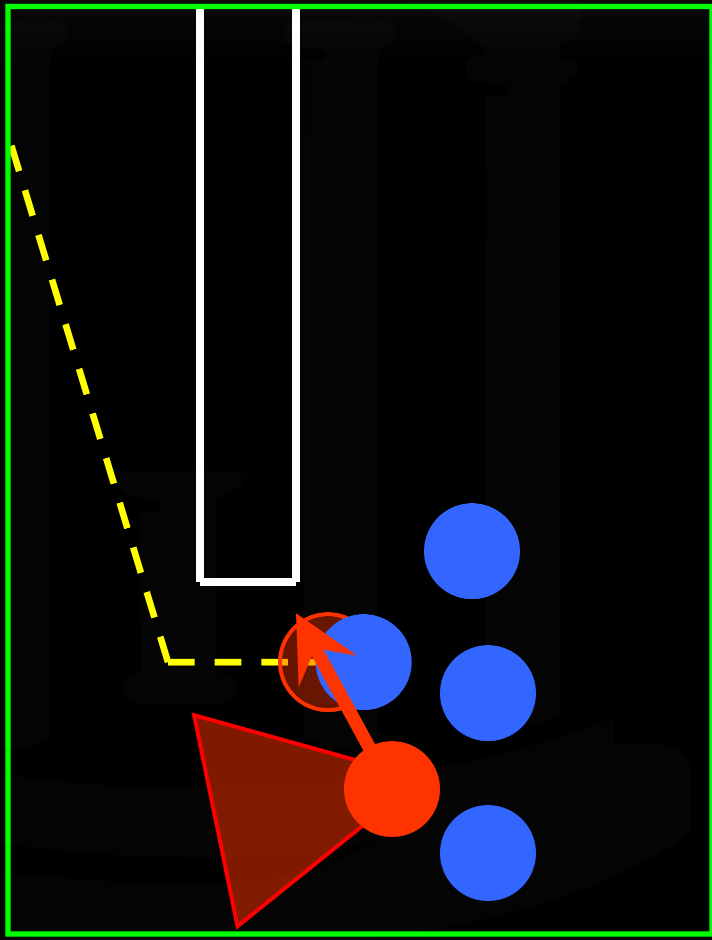
WAYPORTALS

- Local Planning
  - Limited domain knowledge
    - Waypoint
  - Move towards waypoint
  - Avoid collisions



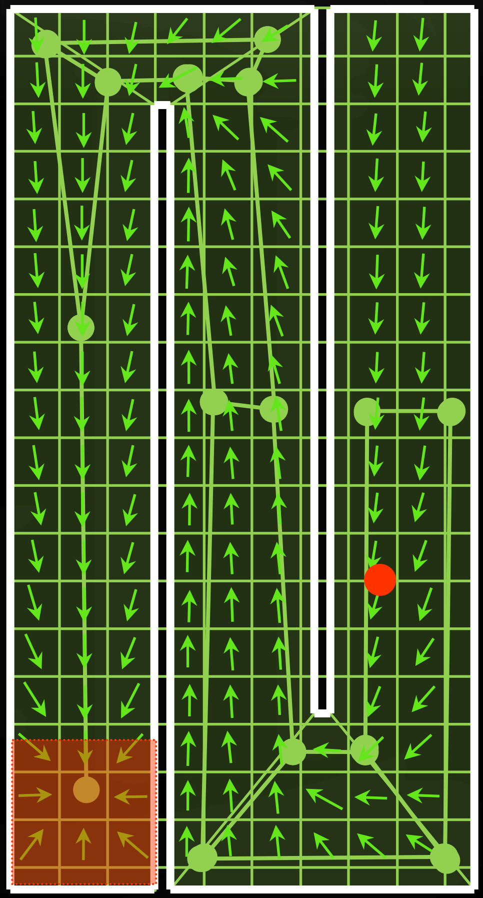
WAYPORTALS

- Local Planning
  - Only knows waypoint
  - Unable to exploit additional space
  - Solution:
    - Small change to global planner to communicate more semantics
    - Extend local planner to use new information



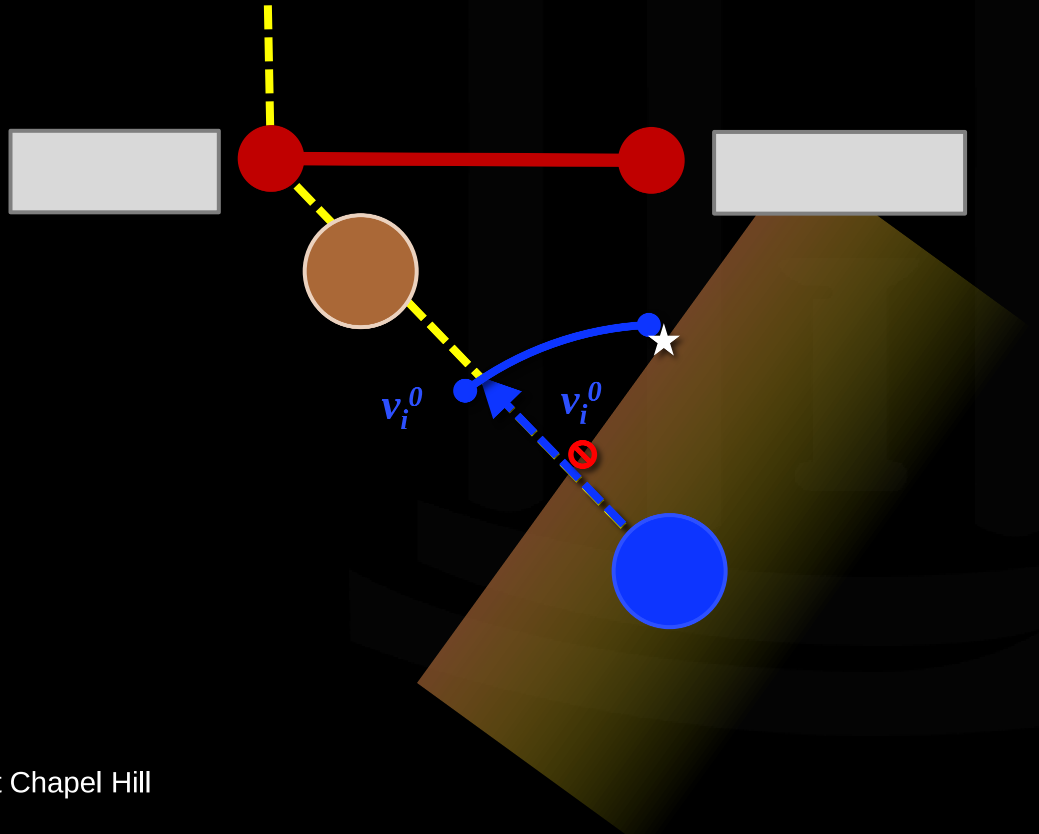
WAYPORTALS

- Previous work in Global Planning
  - Roadmaps
    [Latombe, 1991], [LaValle, 2006]
  - Navigation Mesh
    [Hertel and Mehlhorn, 1985], [Tozour, 2003], [Mononen, 2009], [Snook, 2000], [Kallmann, 2010], [Van Toll et al., 2011]
  - Potential field
    [Khatib, 1986]
  - Dynamic adaptation
    [Jaillet and Simeon 2004; Kallman and Mataric 2004; Ferguson et al. 2006, Zucker et al. 2007], [Sud et al. 2007; Yang and Brock 2007], [Kretz et al, 2012]



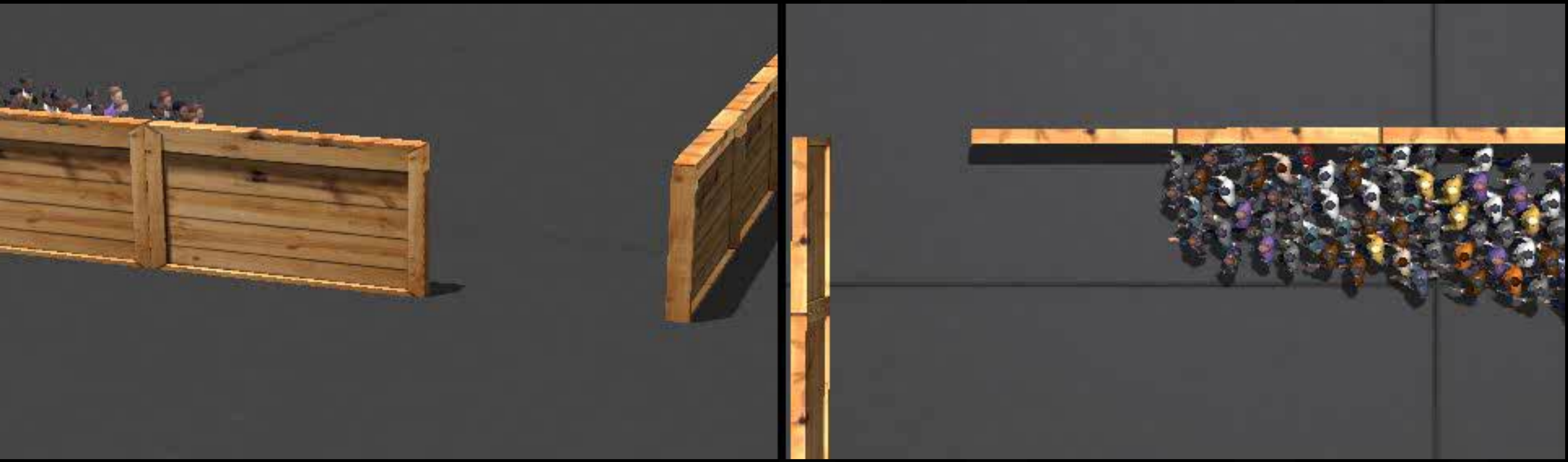
WAYPORTALS

- Limited knowledge leads to limited response
- Promote 1D waypoint to 2D *wayportal*
- Preferred velocity becomes an arc of velocities



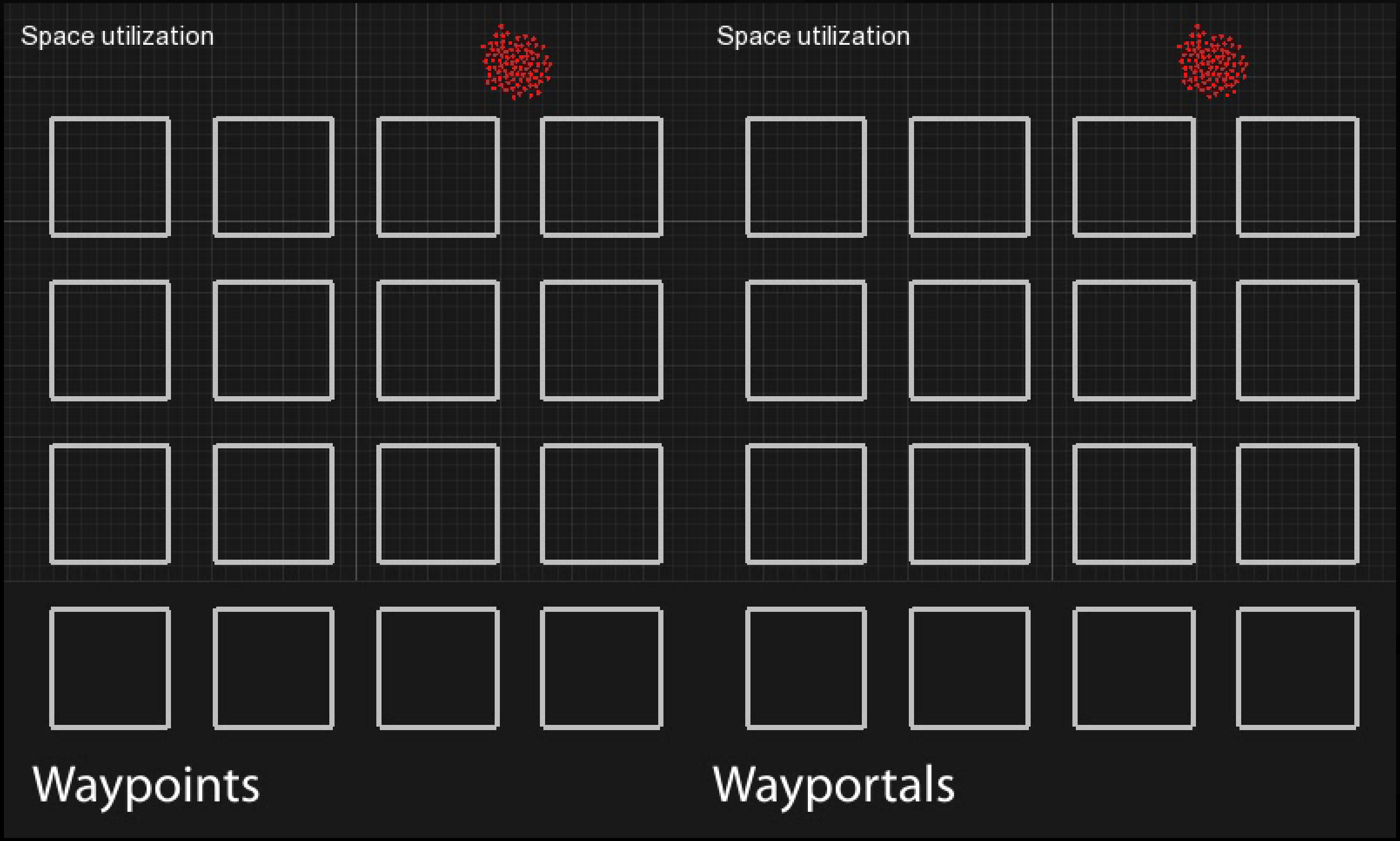
WAYPORTALS

- Using Wayportals



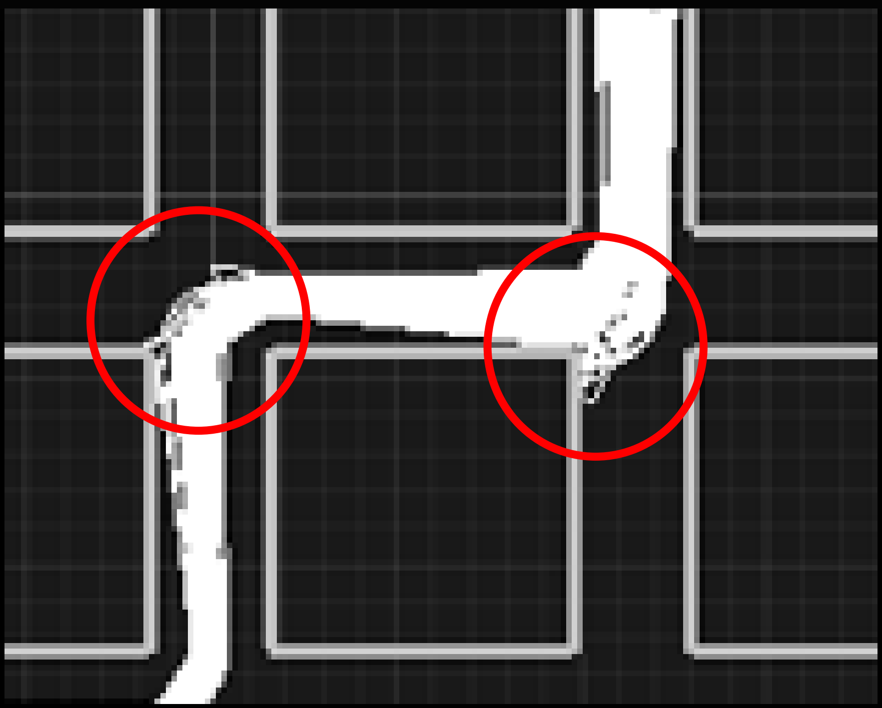
WAYPORTALS

- Improved space utilization and flow

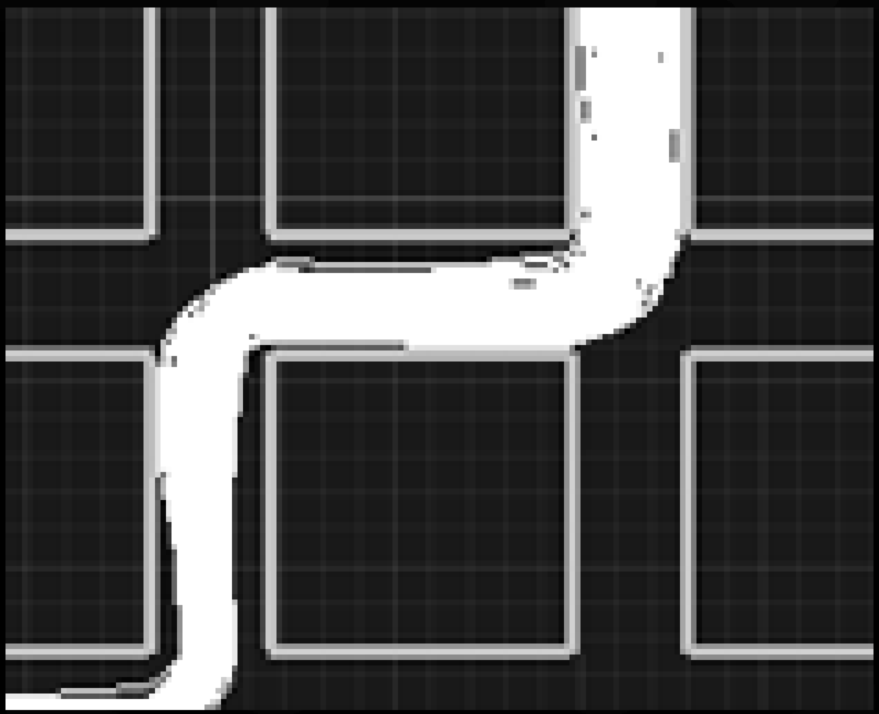


WAYPORTALS

- Improved space utilization and flow



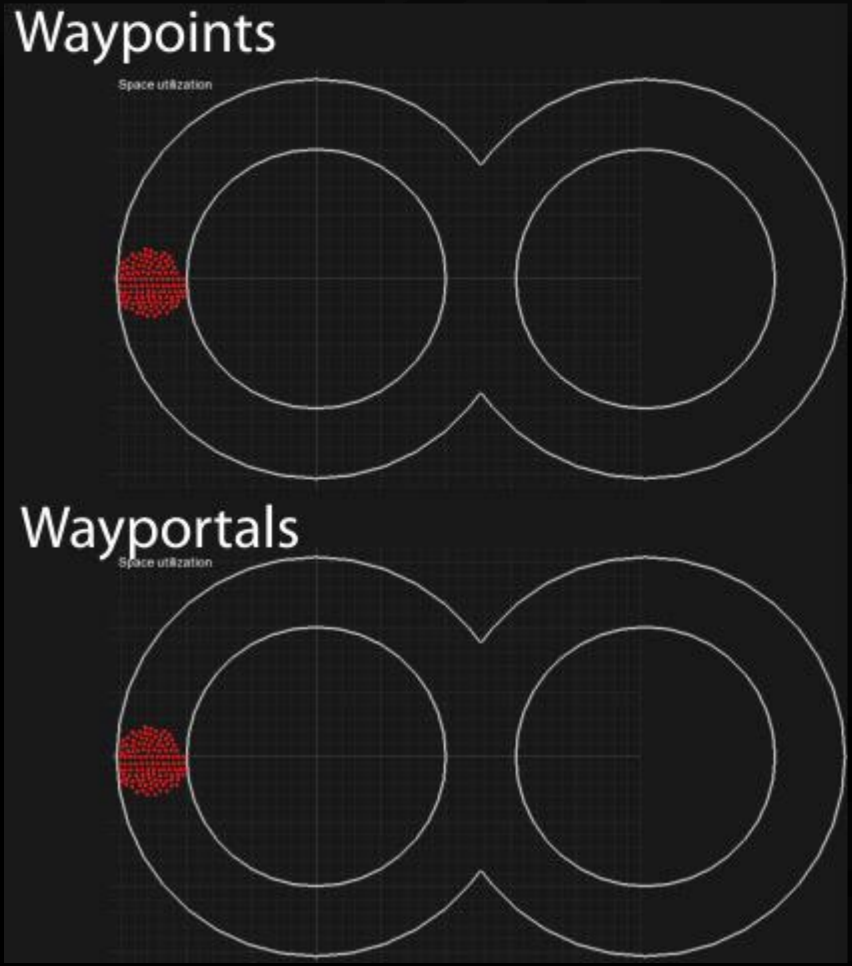
Waypoints



Wayportals

WAYPORTALS

- Improved space utilization and flow



WAYPORTALS

- Summary
  - Formulation for improving space utilization and flow consistent with human behavior
  - Efficiency: minimal increase
    - 10% more expensive over waypoint for 700 agents (from 2.0 μs to 2.2 μs per agent)
  - Correctness: space utilization more consistent with observed human behavior

WAYPORTALS

- Limitations
  - Optimization function is non-convex; approximation constrains the full space of responses

QUESTIONS?