Day 27

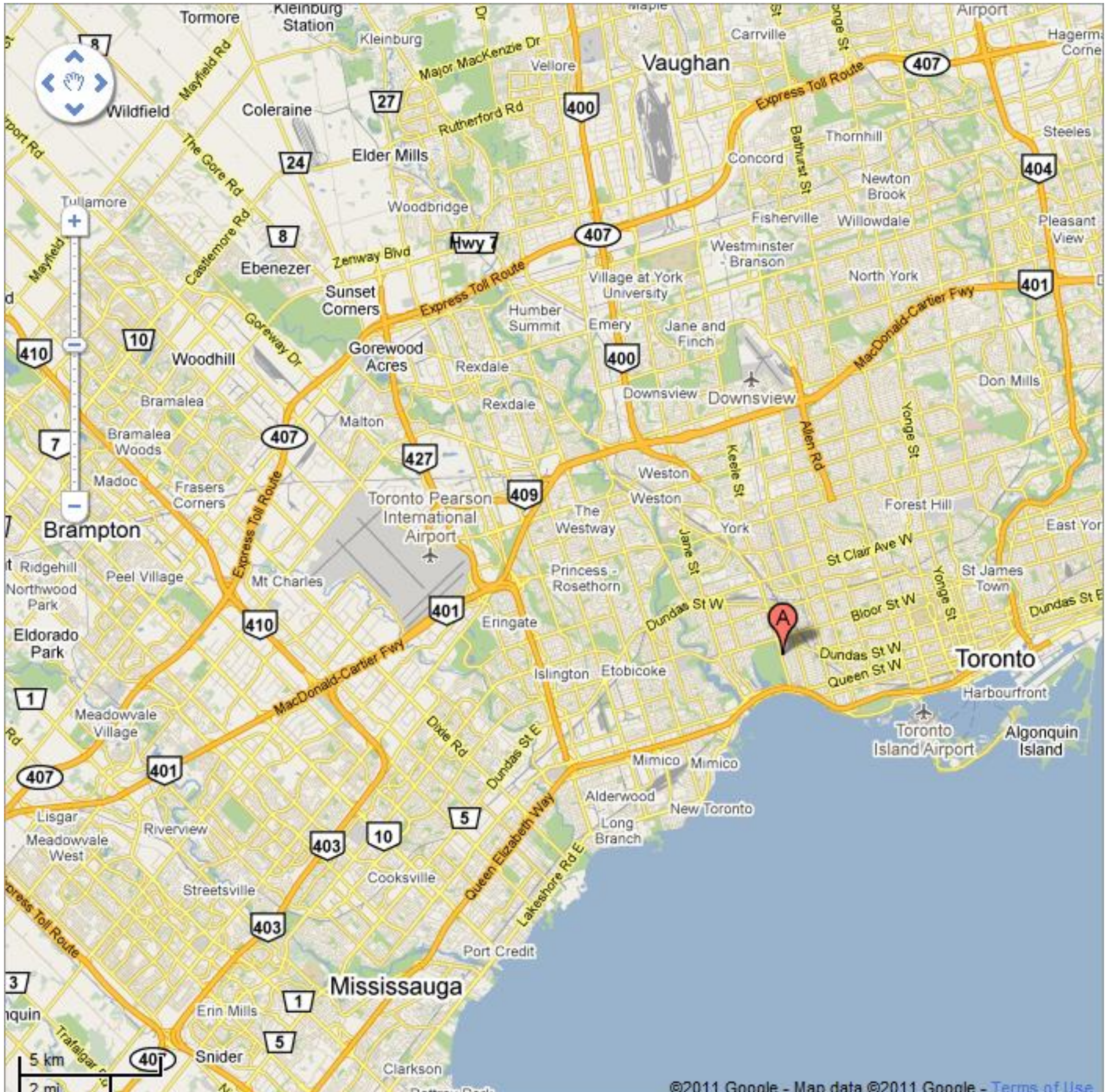
Visibility Graphs

Introduction

- if we have to frequently plan paths on a static environment then it makes sense to use a data structure that supports efficient planning of subsequent paths
  - e.g., visibility graph
    - nodes correspond to vertices of polygonal obstacles
    - edges correspond to paths between nodes

Roadmaps

- consider the highways (in orange) in the GTA



Roadmap

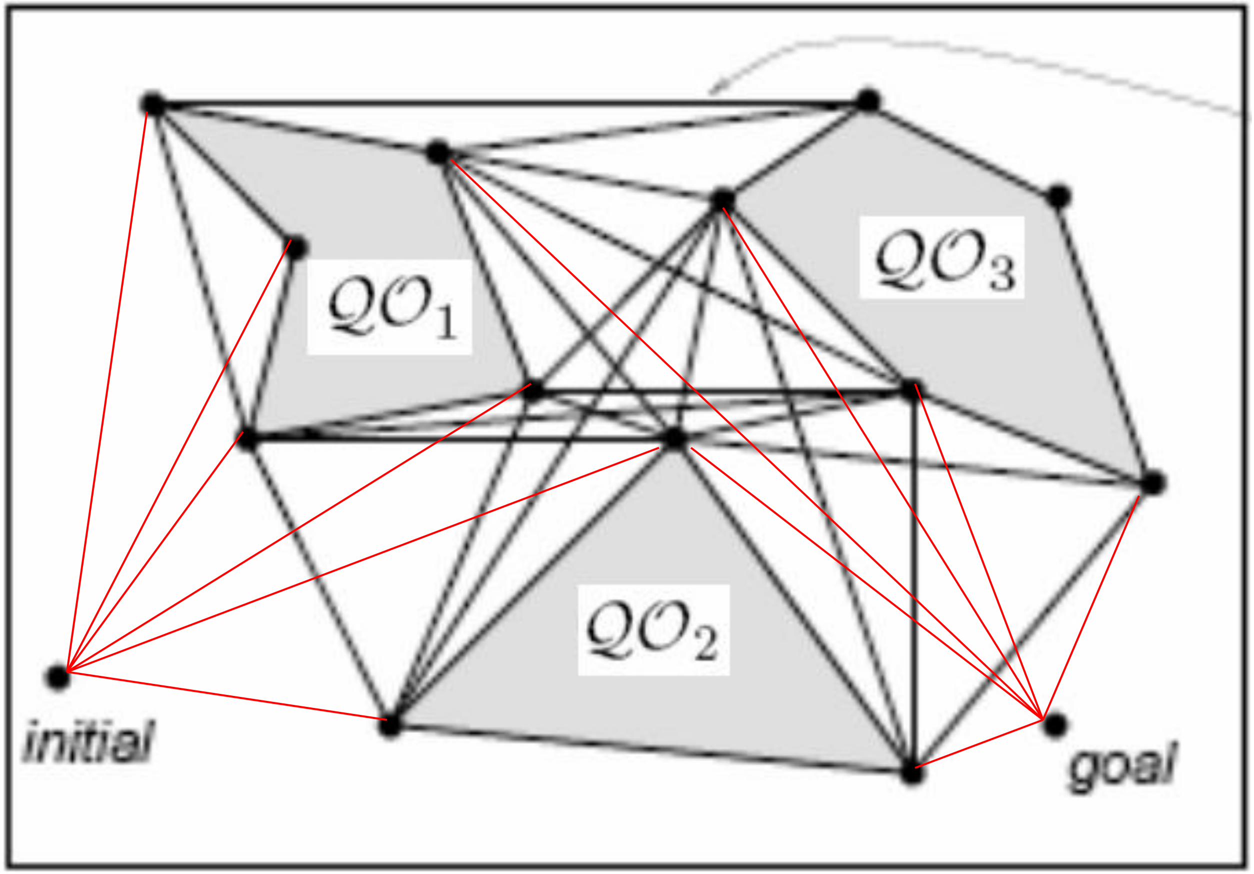
- definition:
  - A union of one-dimensional curves is a roadmap (RM) if for all starting points (q_{start}) and goal points (q_{goal}) in freespace (Q_{free}) that can be connected by a path the following properties hold:
  - Accessibility
    - there exists a path from q_{start} to some $q'_{start} \in RM$
      - i.e., the robot can reach the roadmap from the start point
  - Departability
    - there exists a path from $q'_{goal} \in RM$ to q_{goal}
      - i.e., the robot can depart the roadmap to reach the goal point
  - Connectivity
    - there exists a path in RM from q'_{start} to q'_{goal}
      - i.e., there is a path on the roadmap connecting the start and depart points

Visibility Graph

- defined for 2D space with polygonal obstacles
- a graph where
  - nodes
    - {q_{start}, q_{goal}, and vertices of all obstacles }
  - edges
    - connect all pairs of nodes n_i and n_j that are visible to one another

Visibility Graph

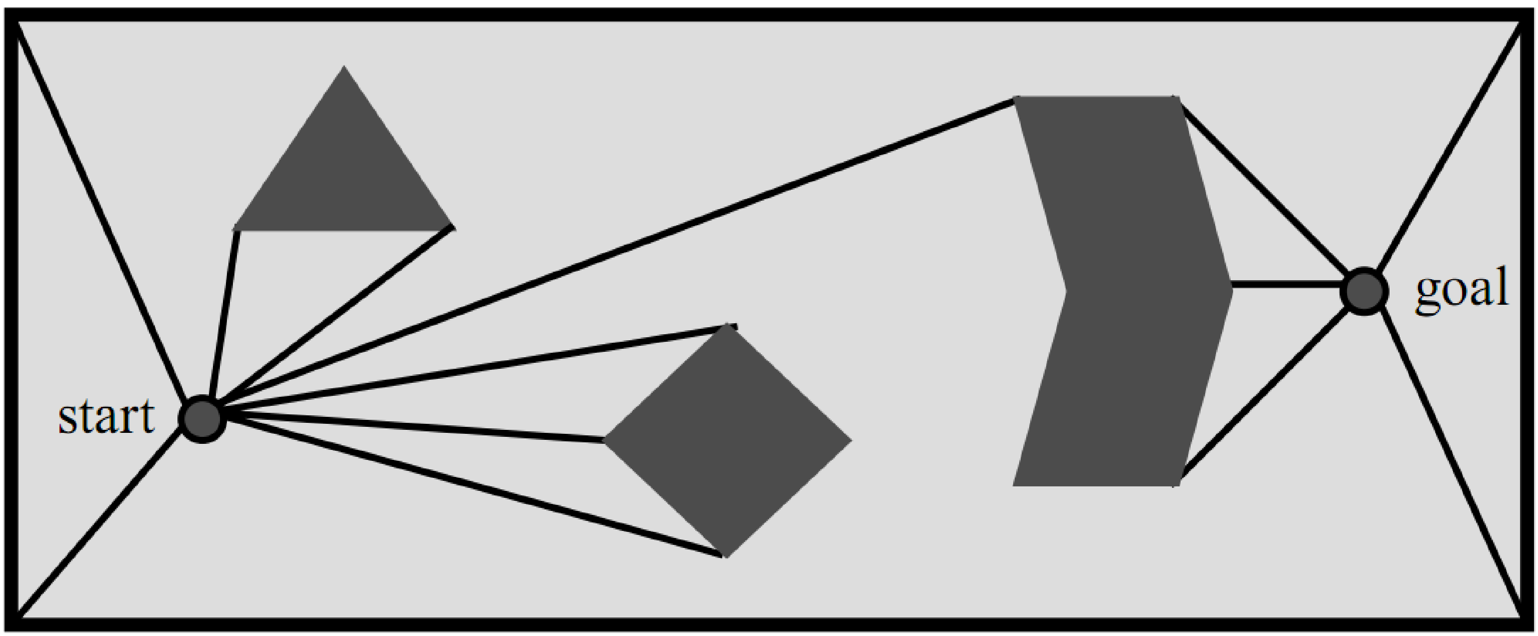
- contains the shortest path between the start and goal



* normally the start and goal points are included in the graph

Visibility Graph Construction

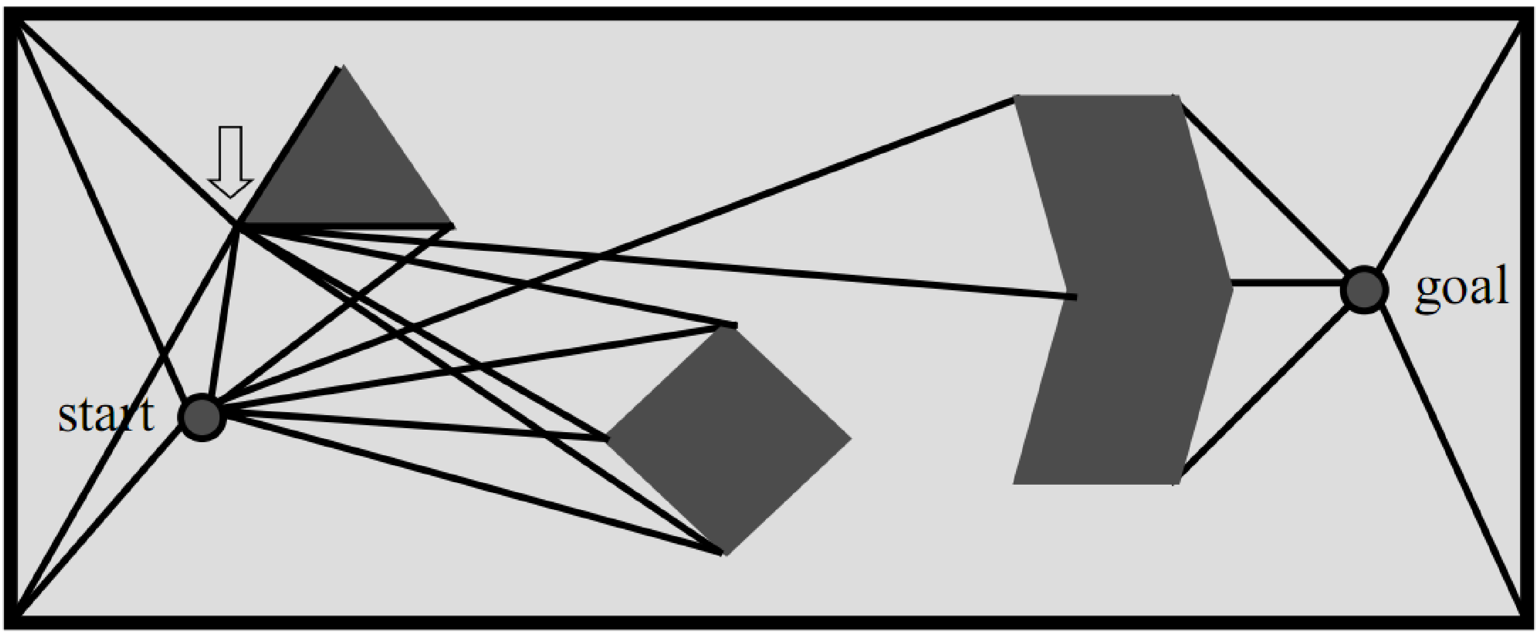
- start by drawing lines of sight from start and goal to all visible vertices



16-735, Howie Choset, with significant copying from G.D. Hager who loosely based his notes on notes by Nancy Amato

Visibility Graph Construction

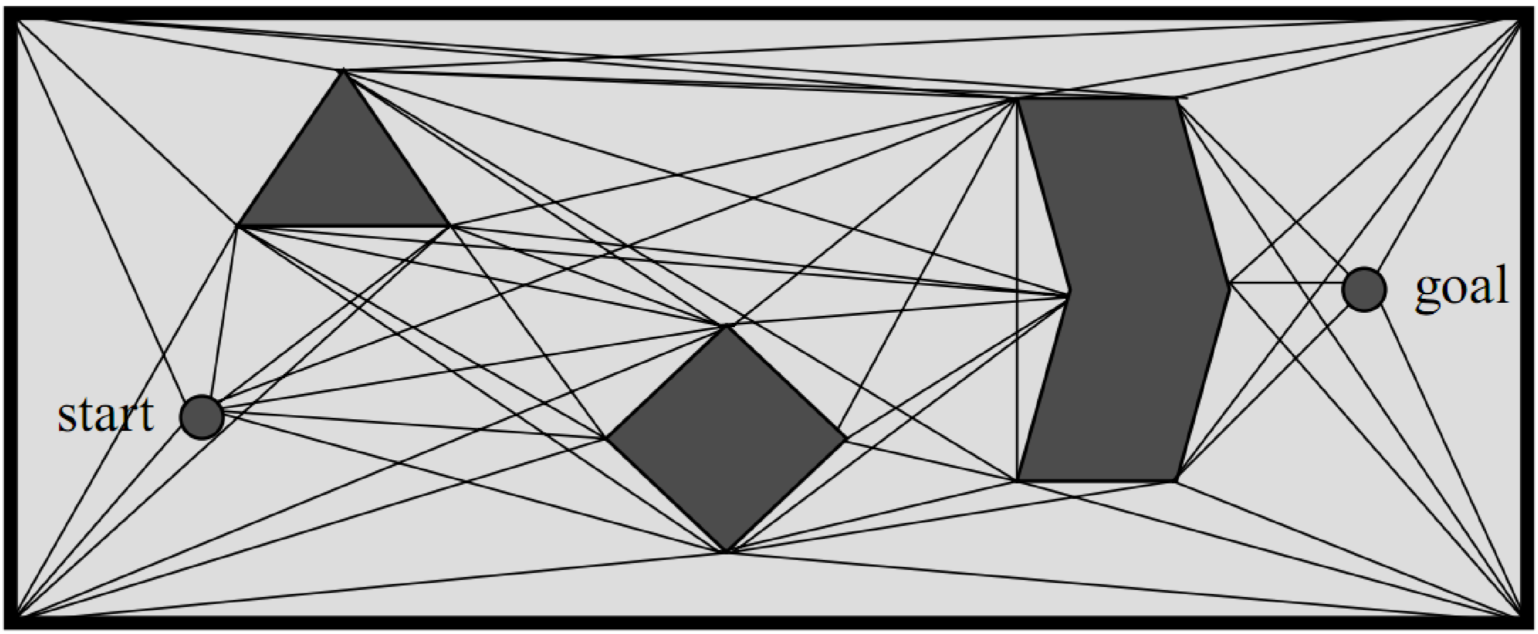
- next, draw lines of sight from every vertex of every obstacle
  - edges of obstacles are also lines of sight



16-735, Howie Choset, with significant copying from G.D. Hager who loosely based his notes on notes by Nancy Amato

Visibility Graph Construction

- repeat until done



16-735, Howie Choset, with significant copying from G.D. Hager who loosely based his notes on notes by Nancy Amato

Visibility Graph Construction

- the algorithm described is $O(N^3)$ where N is the number of vertices

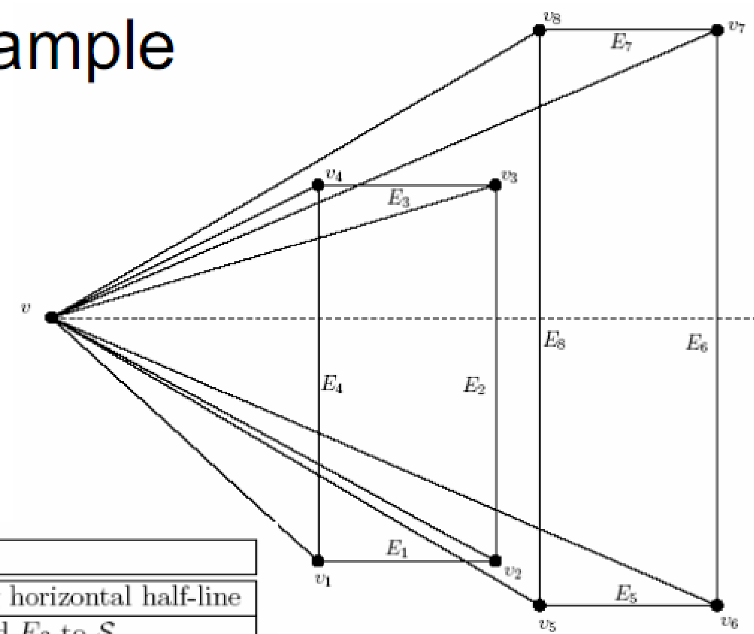
The Sweepline Algorithm

- Goal: calculate the set of vertices v_i that are visible from v
- visibility: a segment v-v_i is visible if
  - it is not within the object
  - the closest line intersecting v-v_i is beyond v_i
- **Algorithm:**
  Initially:
  - calculate the angle α_i of segment v-v_i and sort vertices by this creating list E
  - create a list of edges that intersect the horizontal from v sorted by intersection distance
- For each α_i
  - if v_i is visible to v then add v-v_i to graph
  - if v_i is the "beginning" of an edge E, insert E in S
  - if v_i is the "end" of and edge E, remove E from S

Analysis: For a vertex, n log n to create initial list, log n for each α_i
Overall: n log (n) (or n^2 log (n) for all n vertices

16-735, Howie Choset, with significant copying from G.D. Hager who loosely based his notes on notes by Nancy Amato

Example



Vertex	New $\mathcal{S}$	Actions
Initialization	$\{E_4, E_2, E_8, E_6\}$	Sort edges intersecting horizontal half-line
α_3	$\{E_4, E_3, E_8, E_6\}$	Delete E_2 from $\mathcal{S}$. Add E_3 to $\mathcal{S}$.
α_7	$\{E_4, E_3, E_8, E_7\}$	Delete E_6 from $\mathcal{S}$. Add E_7 to $\mathcal{S}$.
α_4	$\{E_8, E_7\}$	Delete E_3 from $\mathcal{S}$. Delete E_4 from $\mathcal{S}$. ADD (v, v_4) to visibility graph
α_8	$\{\}$	Delete E_7 from $\mathcal{S}$. Delete E_8 from $\mathcal{S}$. ADD (v, v_8) to visibility graph
α_1	$\{E_1, E_4\}$	Add E_4 to $\mathcal{S}$. Add E_1 to $\mathcal{S}$. ADD (v, v_1) to visibility graph
α_5	$\{E_4, E_1, E_8, E_5\}$	Add E_8 to $\mathcal{S}$. Add E_5 to $\mathcal{S}$.
α_2	$\{E_4, E_2, E_8, E_5\}$	Delete E_1 from $\mathcal{S}$. Add E_2 to $\mathcal{S}$.
α_6	$\{E_4, E_2, E_8, E_6\}$	Delete E_5 from $\mathcal{S}$. Add E_6 to $\mathcal{S}$.
Termination		