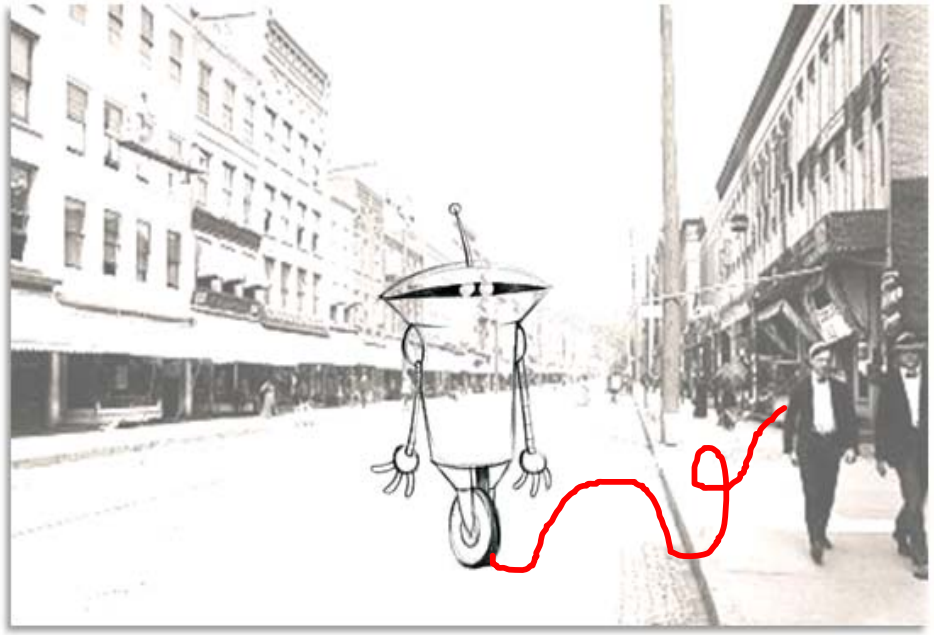
Lecture 17: Motion planning (4)
Roadmaps and Graph search

Topics:

- Roadmaps
- Voronoi decomposition
- Graph search algorithms
  - Breadth first
  - Depth first
  - Dijkstra
  - A*



Reading:

- Choset: 5, H
- LaValle: 2,6

Motion planning

Given: ability to track $q_{1:t}$

q_{goal}

Sometimes: map, q_0

Find: $u_{1:t}$ s.t. $q_t = q_{goal}$

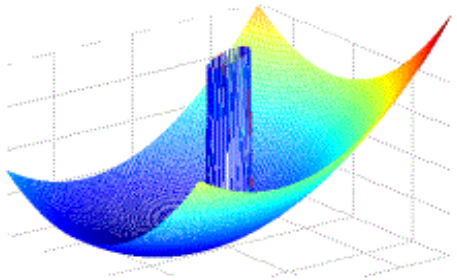
Assuming: in this class

2D motion

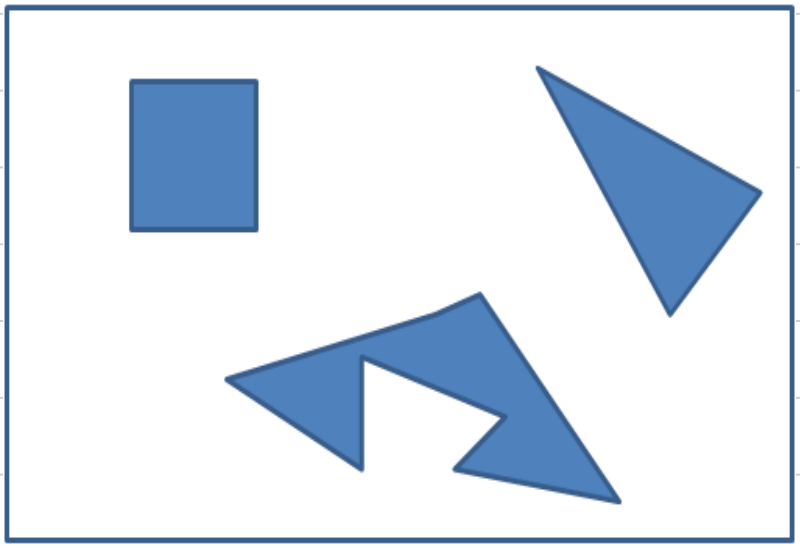
$\dot{q} = u$

holonomic

Localization		Motion Planning
Dead Reckoning	"Missing info"	
grid based	"Discrete"	cell decomposition, Roadmaps
EkF / kf	"Continuous"	potential / Navigation Functions
PF	"Samples"	



$G = (V, E)$



	cell decomposition	Roadmap
V	cell	points in Q_{free}
E	Adjacency	paths in Q_{free} between $v_i, v_j \in V$

Roadmap

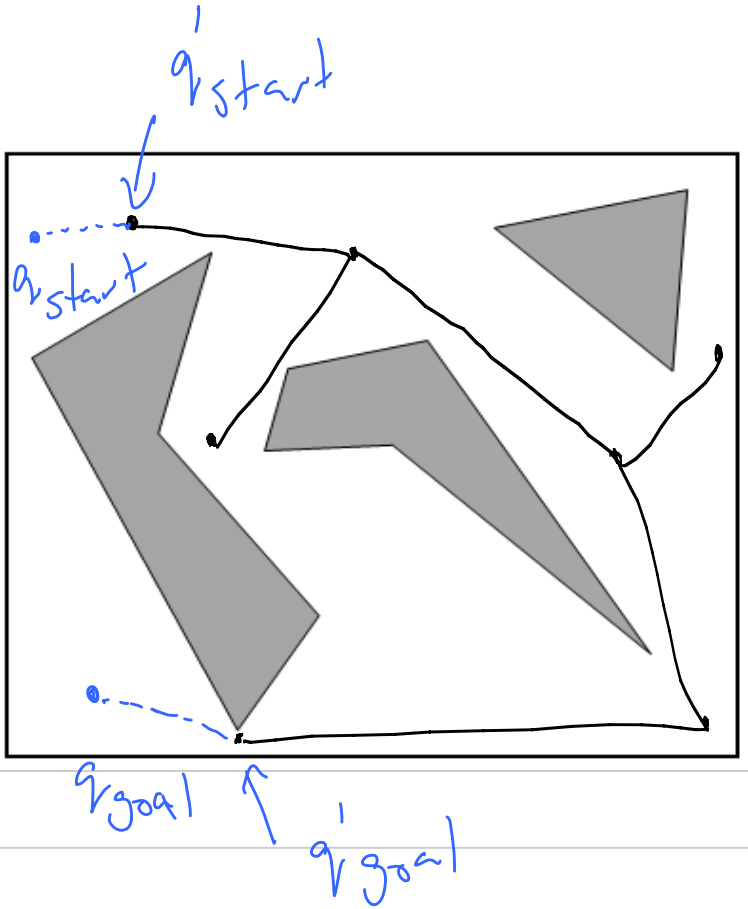
$G = (V, E)$ is a roadmap if

$\forall q_{start}, q_{goal} \in Q_{free}$:

(1) $\exists$ path $\in Q_{free}$ from q_{start} to $q'_{start} \in V$

(2) $\exists$ path $\in Q_{free}$ from $q'_{goal} \in V$ to q_{goal}

(3) $\exists$ path in G from q'_{start} to q'_{goal}



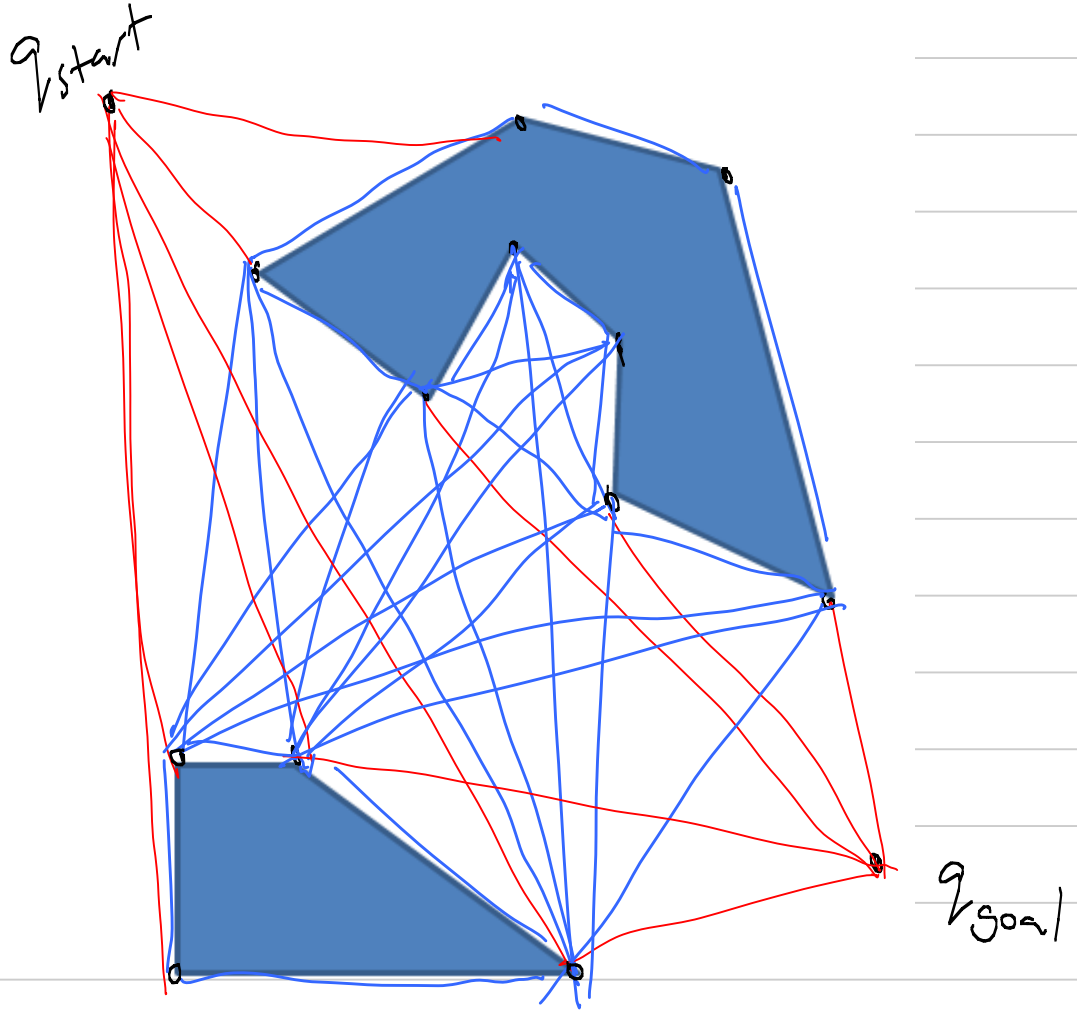
Visibility Graph

workspace: polygonal

idea: Line of sight

V = all vertices of polygons + q_{start} + q_{goal}

E = line of sight segments



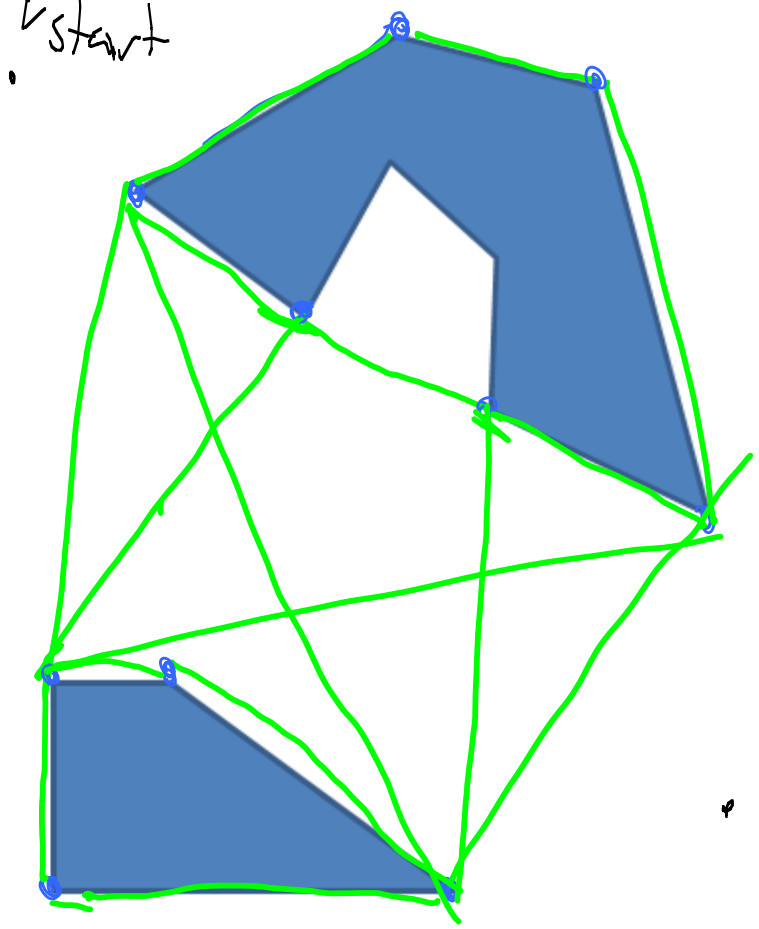
$e_{ij} \in E$ iff $s v_i + (1-s) v_j \in Q_{free} + Boundary \quad \forall s \in [0,1]$

Reduced Visibility Graph

Remove edges/vertices

V = reflex vertices + q_{start} + q_{goal}
= angle in $Q_{free} > \pi$

$e_{ij} \in E$ iff V_i, V_j are consecutive reflex vertices OR V_i, V_j are connected by a bitangent line (line of sight + tangent to obs.)



Voronoi decomposition

maximum clearance

$\{S_i\}$ points in Q_{free}

$$Vor(S_i) = \{ q \in Q_{free} \mid d(q, S_i) < d(q, S_j) \ \forall j \neq i \}$$

http://www.cs.cornell.edu/Info/People/chew/Delaunay.html

Generalized Voronoi Diagram (GVD)

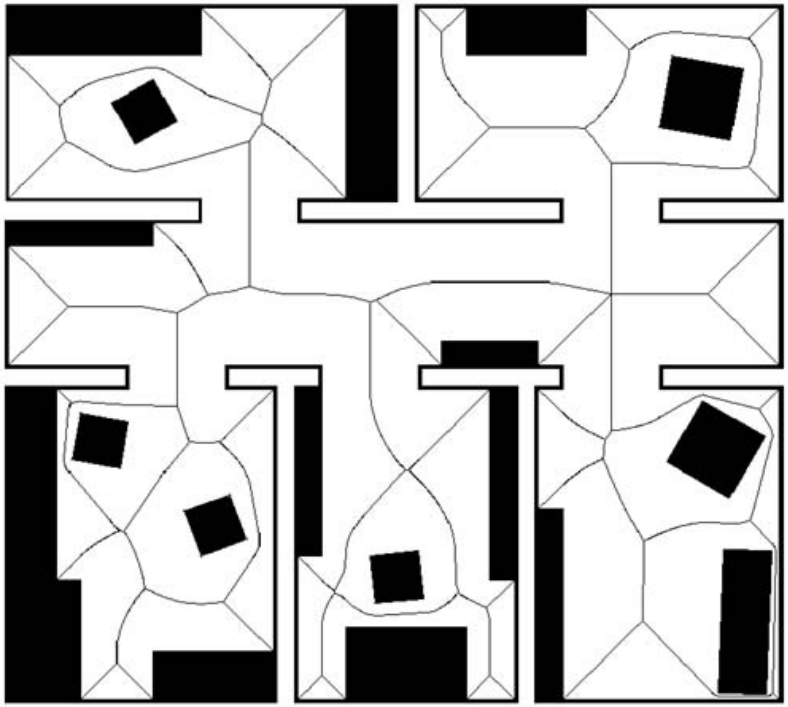


Image from: http://www.sfbtr8.uni-bremen.de/project/r3/HGVG/hierarchicalVGraphs.html

Graph search

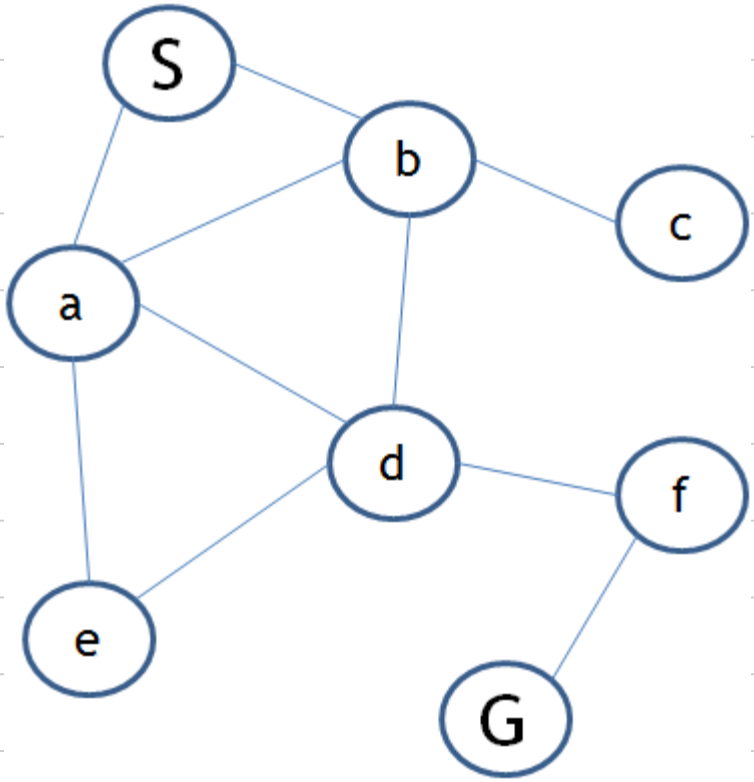
Input: $G = (V, E)$

V_{start}

V_{goal}

Output: path from V_{start} to V_{goal}

(sequence of nodes / edges)



$v_{start} = x_I$

$v_{goal} = x_G$

Graph search

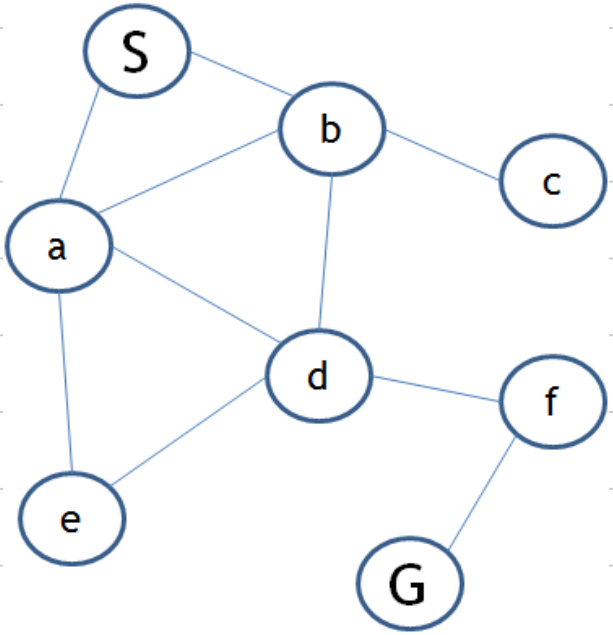
```
FORWARD_SEARCH
1   Q.Insert(x_I) and mark x_I as visited
2   while Q not empty do
3       x ← Q.GetFirst()
4       if x ∈ X_G
5           return SUCCESS
6       forall u ∈ U(x)
7           x' ← f(x, u)
8           if x' not visited
9               Mark x' as visited
10              Q.Insert(x')
11          else
12              Resolve duplicate x'
13  return FAILURE
```

choosing a node to explore

if it is the goal → done

possible actions

adding x' to Q



LV Fig 2.4

successor nodes

maintain 3 sets of nodes: unvisited

active (Q)

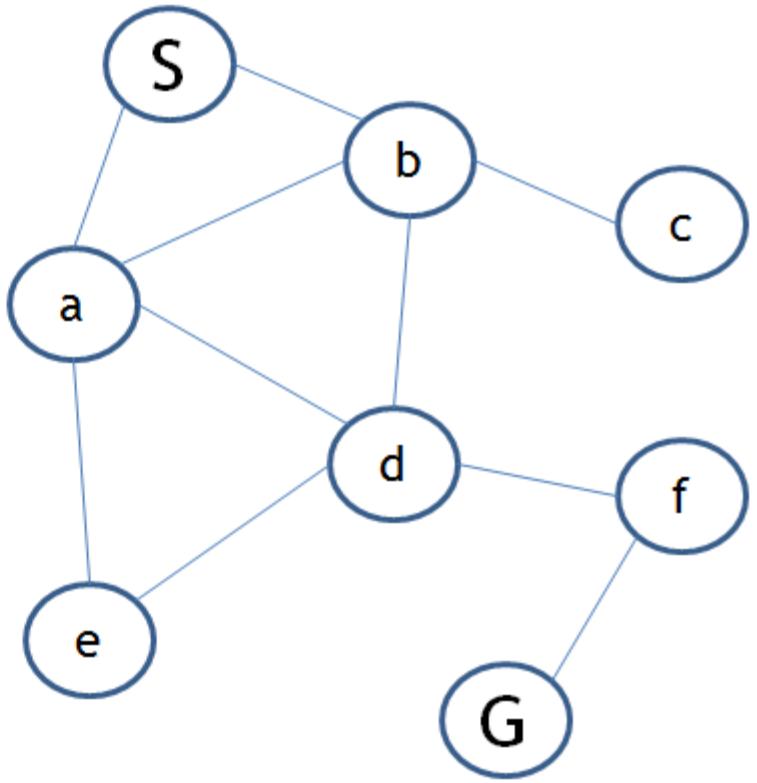
dead

Graph search

- Breadth first

exhustive

Q.GetFirst : FIFO



step	Q
0	S
1	a, b
2	b, e, d
3	e, d, c
4	d, c
5	c, f
6	f
7	G

nodes	parents
a, b	S
e, d	a
c	b
f	d
G	f

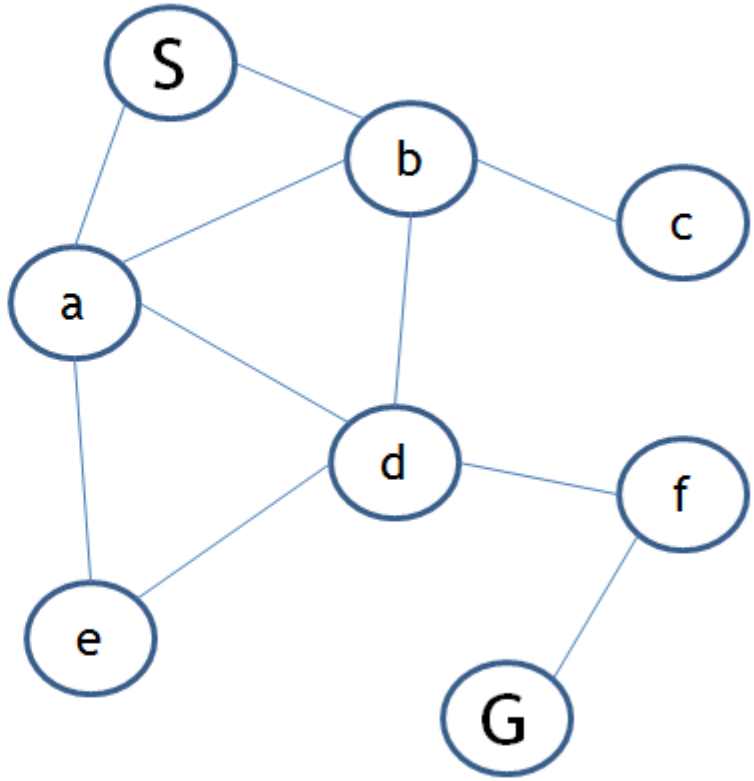
Graph search

- Depth first

greedy

Q.get first : LIFO

step	Q
0	s
1	a, b
2	a, c, d
3	a, c, e, f
4	a, c, e, g



nodes	parent
a, b	s
c, d	b
e, f	d
g	f