NYU

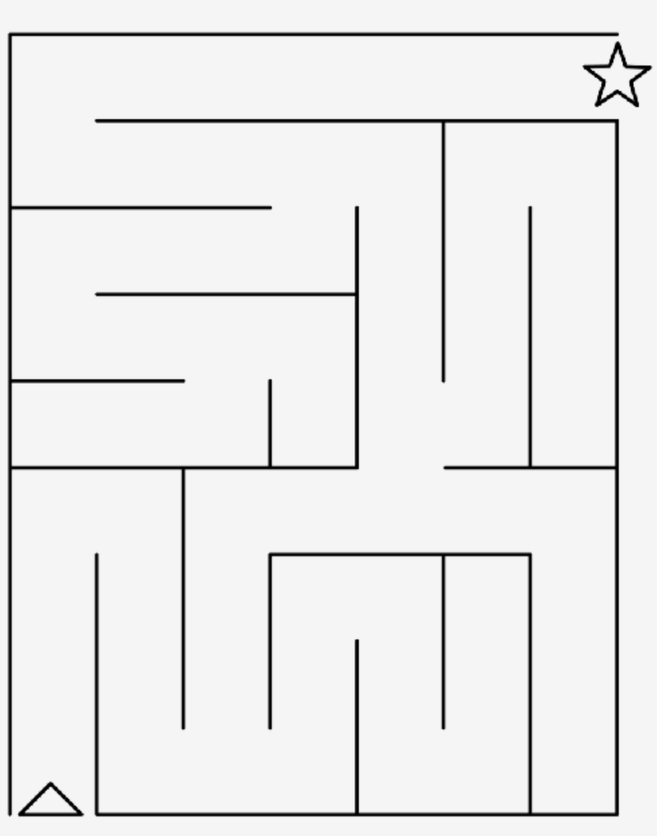
Introduction to Robot Intelligence
[Spring 2023]

Planning and A* search

April 27, 2023

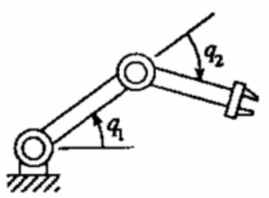
Lerrel Pinto

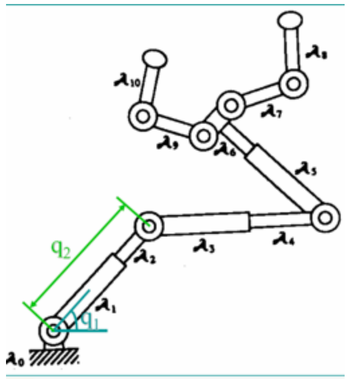
A 2D maze solving problem

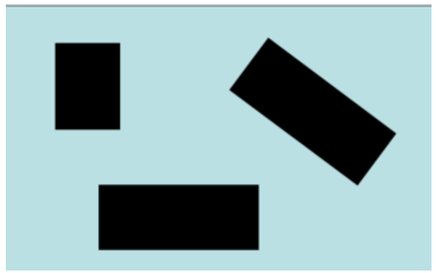


High dimensional planning examples

Fundamental Concept: C-space



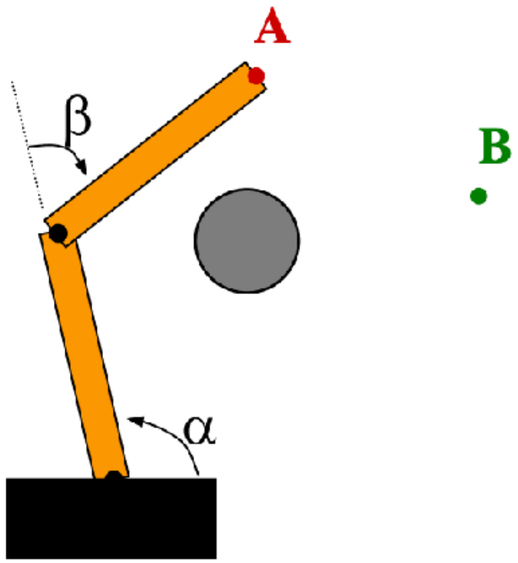




Configuration Space

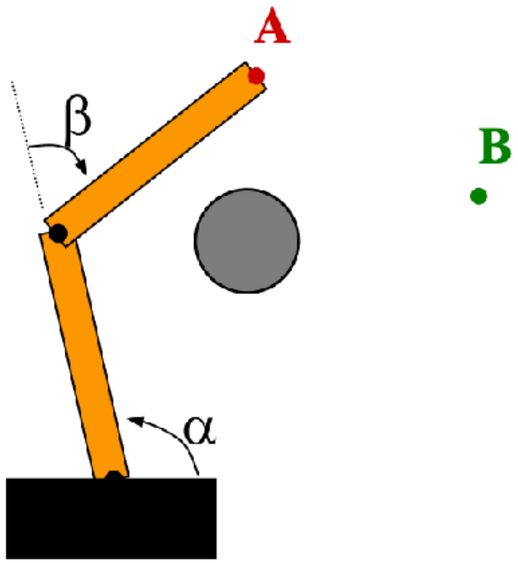
Planning Algorithms

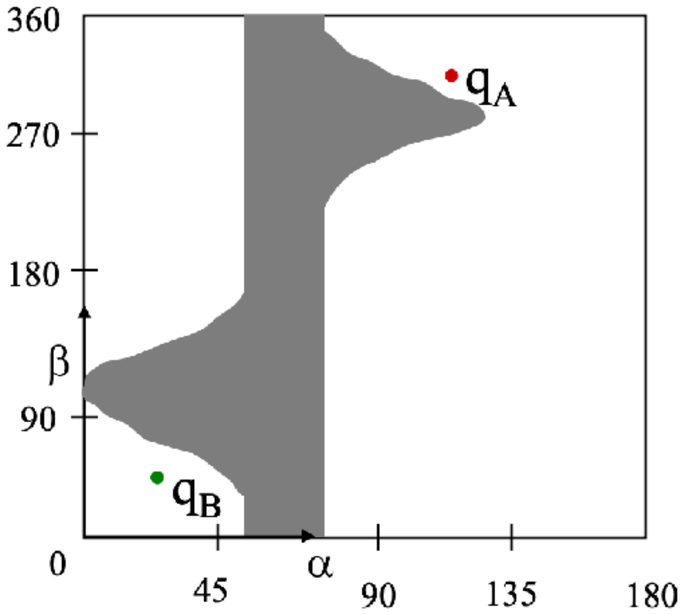
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

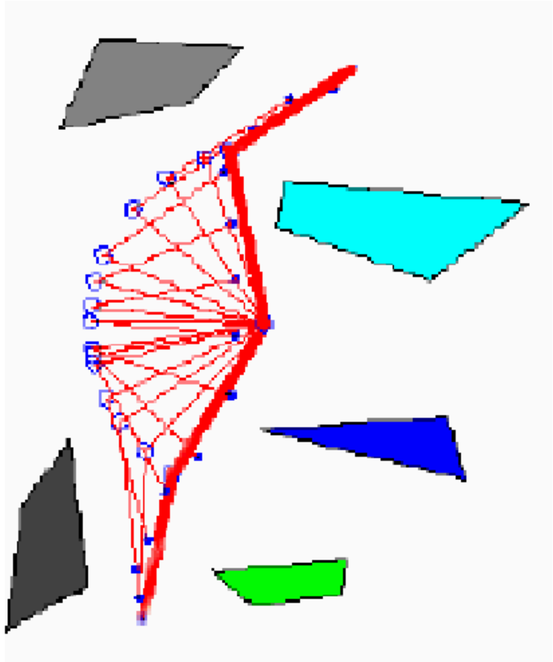
Fundamental Concept: C-space





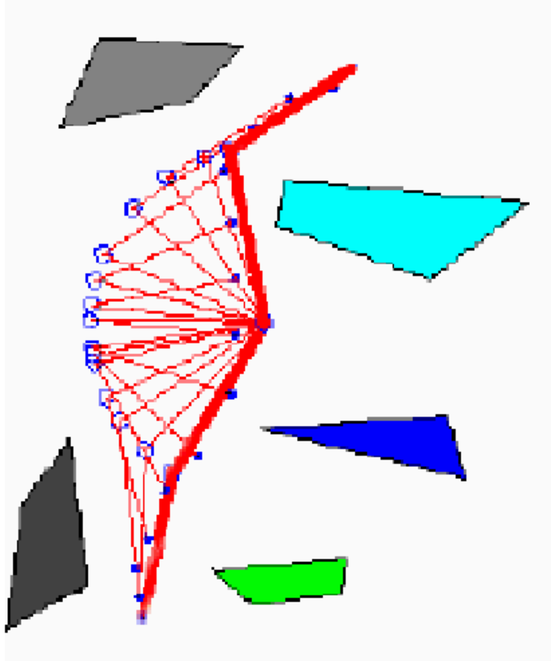
Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

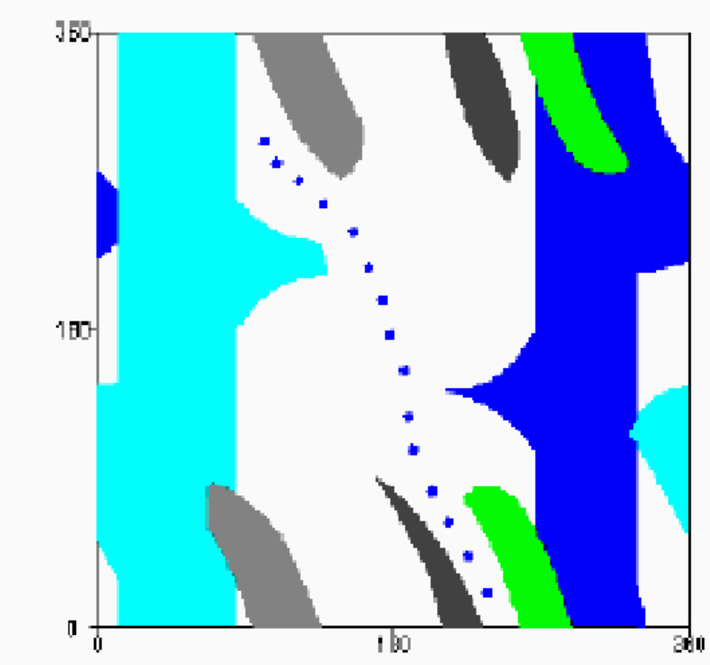
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

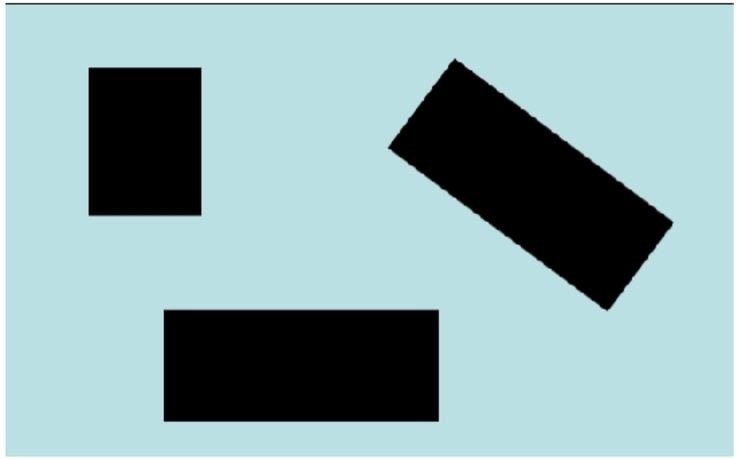
Fundamental Concept: C-space





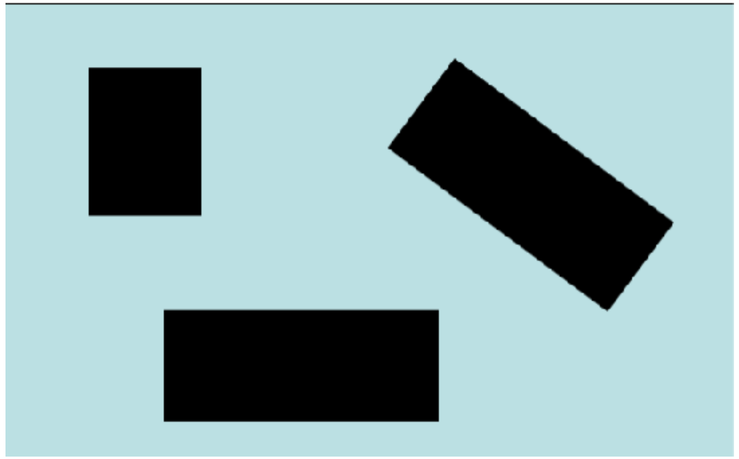
Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

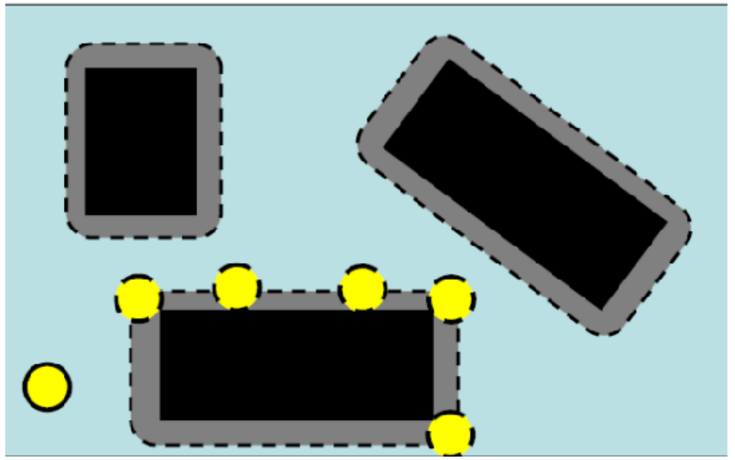
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

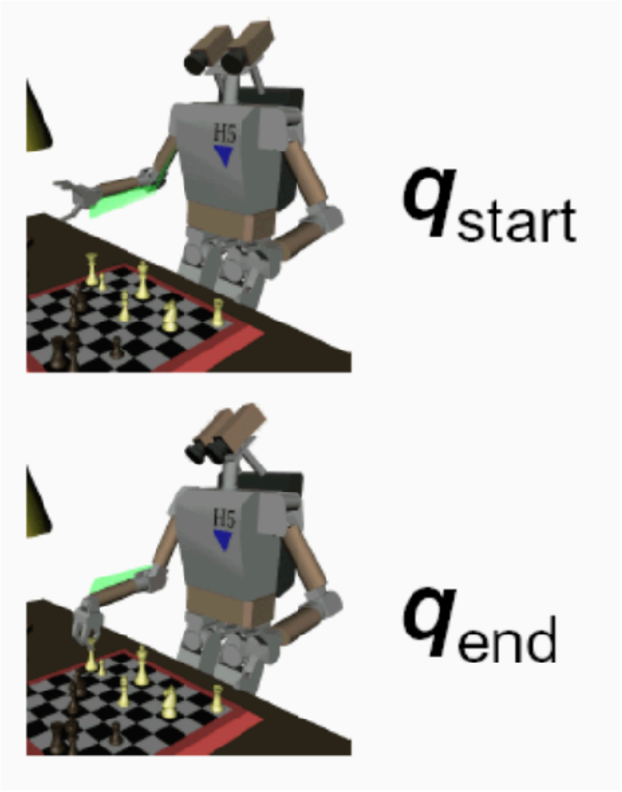
Fundamental Concept: C-space

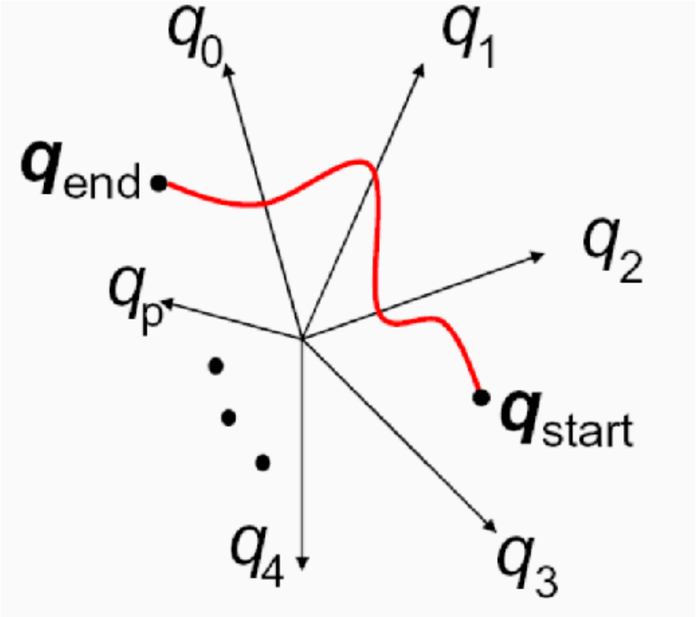




Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

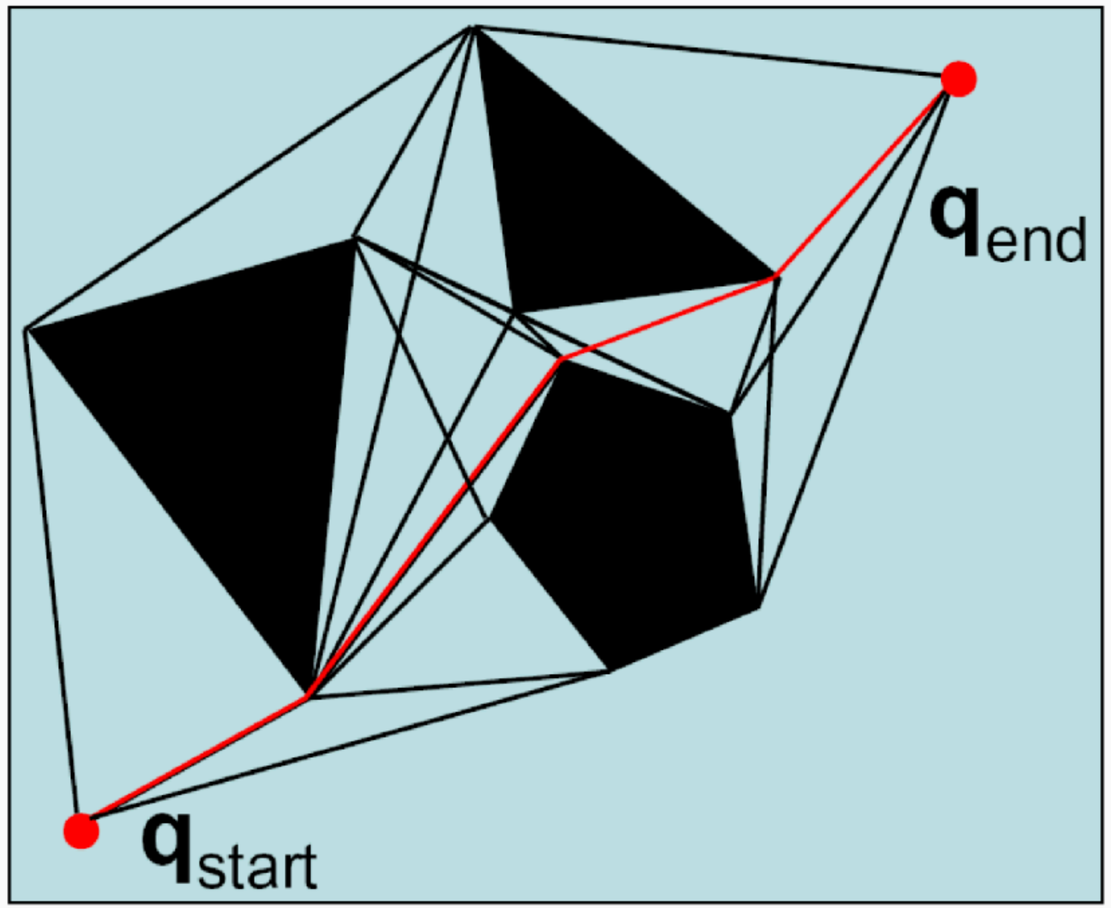
The motion planning problem





Slide credits: Howie Choset

Visibility Maps

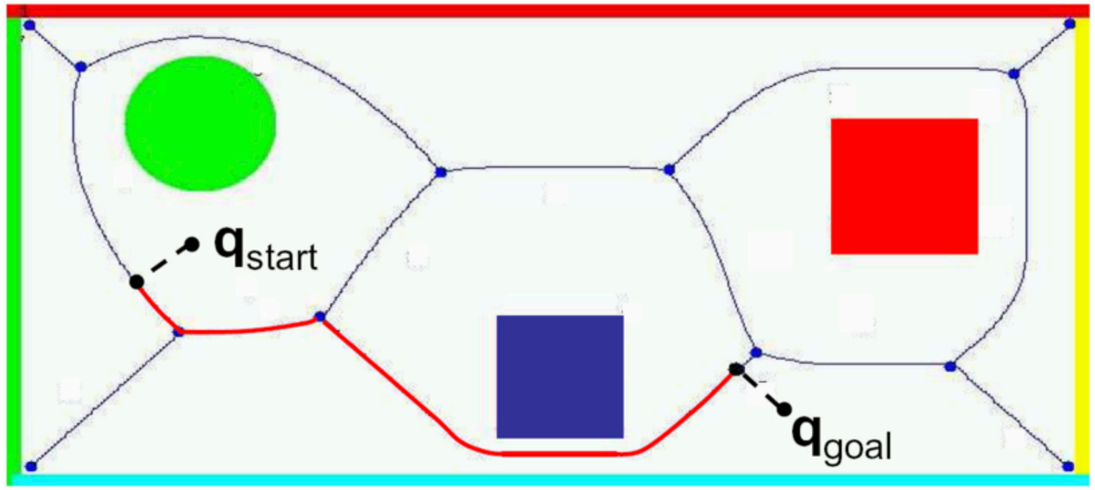


General Strategy:

1. Convert C-space to a graph.
2. Each polygonal vertex is a node.
3. Edges can be determined by ‘visibility’.
4. Once converted to graph… this is a graph search problem.

Slide credits: Howie Choset

Voronoi Diagram

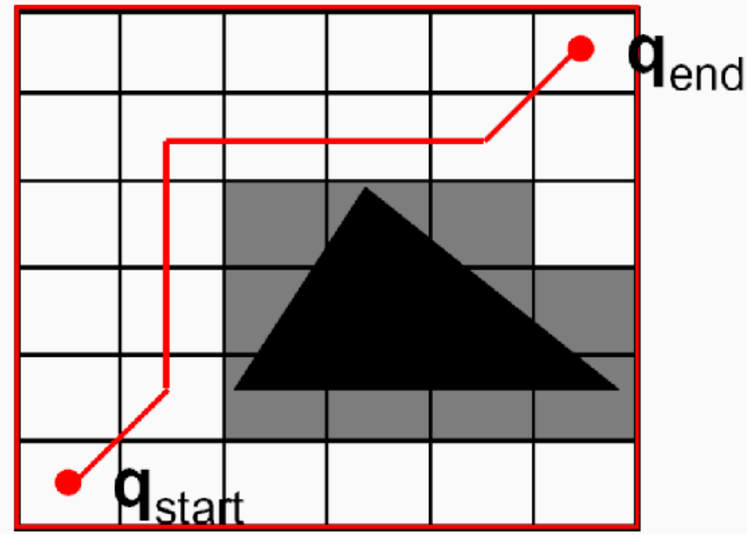


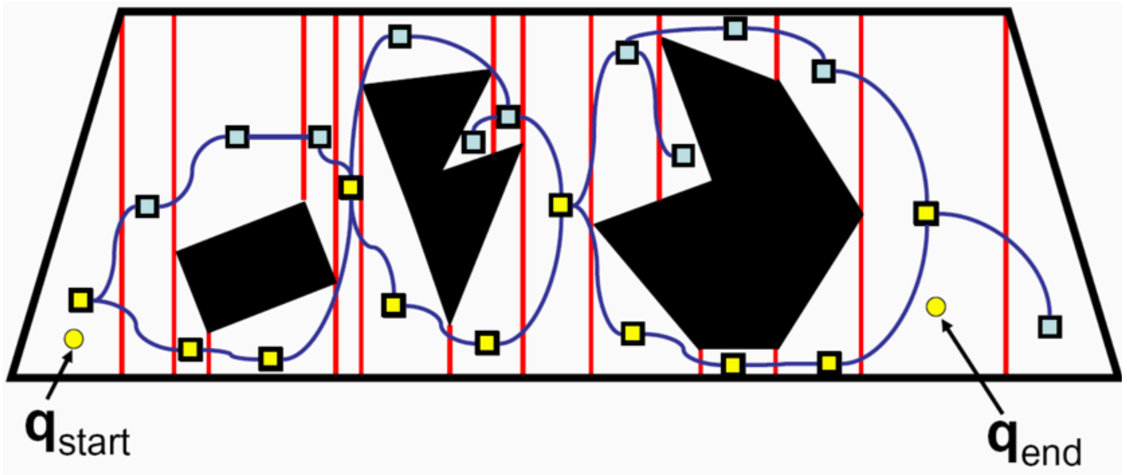
General Strategy:

1. Convert C-space to a Voronoi Diagram.
2. Given a start and end position, find closest points on the roadmap.
3. Solve motion planning as Graph search.

Slide credits: Howie Choset

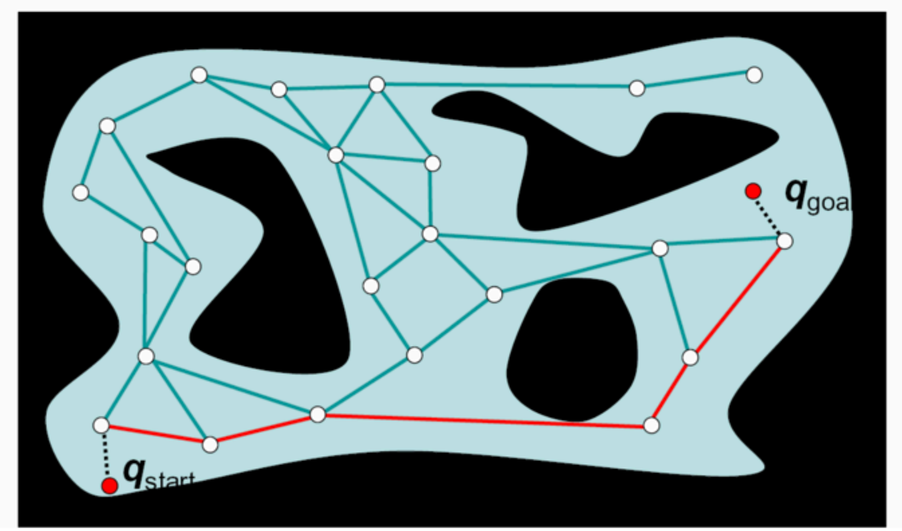
Cell Decomposition





Slide credits: Howie Choset

Sampling based



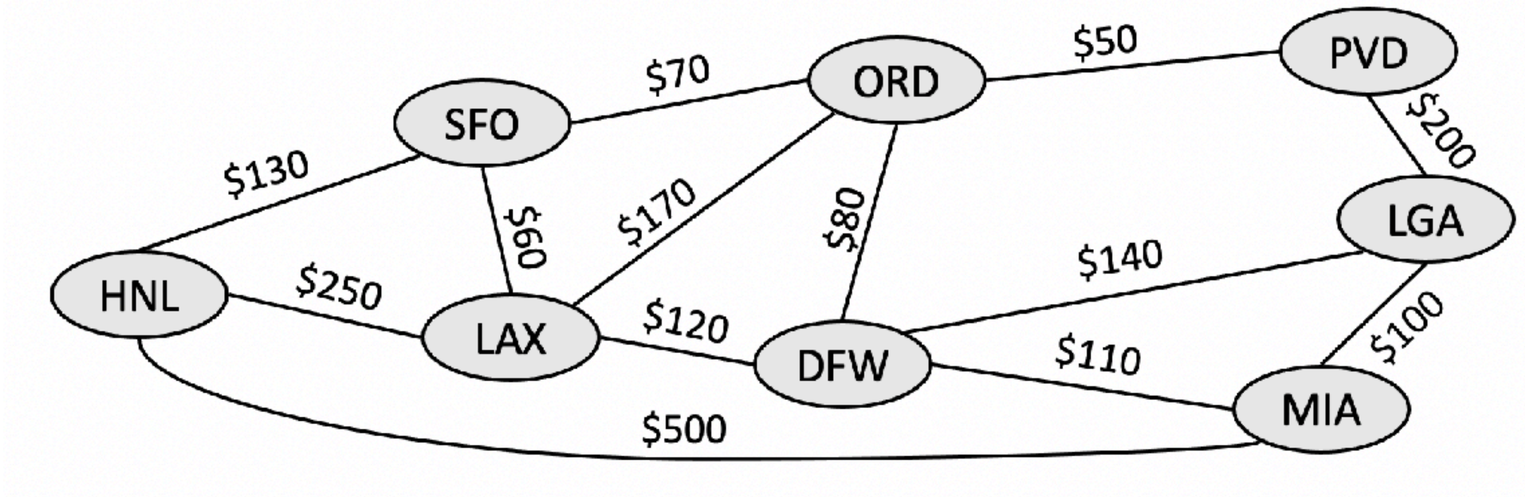
General Strategy:

1. Use random sampling to find points.
2. Create a graph from the sampled points.
3. Solve motion planning as Graph search.

Slide credits: Howie Choset

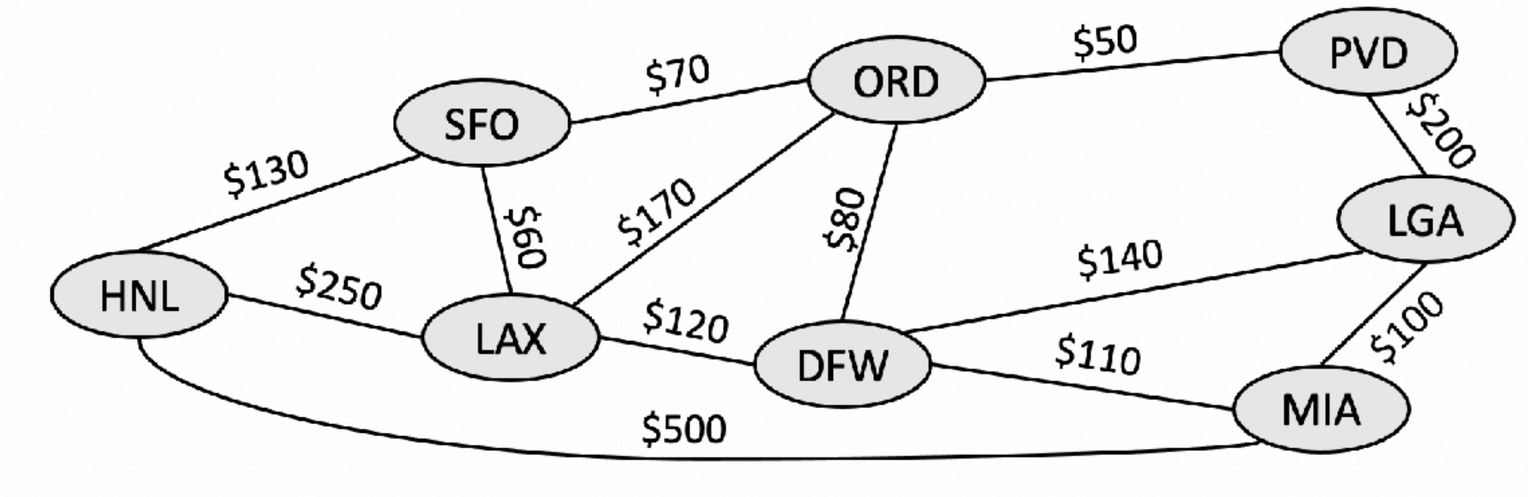
But how do you do graph search?

Example of a graph



Slide credits: Marty Stepp, UW

Example of a graph



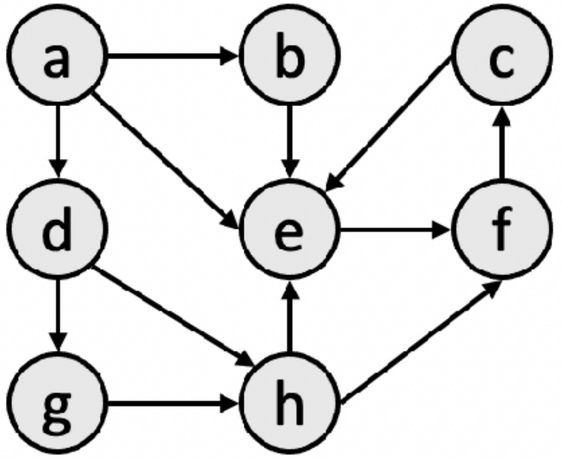
- Sometimes, we just want *any* path (or want to know there *is* a path).
- Sometimes, we want to minimize path *length* (# of edges).
- Sometimes, we want to minimize path *cost* (sum of edge weights).

Slide credits: Marty Stepp, UW

Depth First Search

depth-first search (DFS): Finds a path between two vertices by exploring each possible path as far as possible before backtracking.

- Often implemented recursively.
- Many graph algorithms involve *visiting* or *marking* vertices.

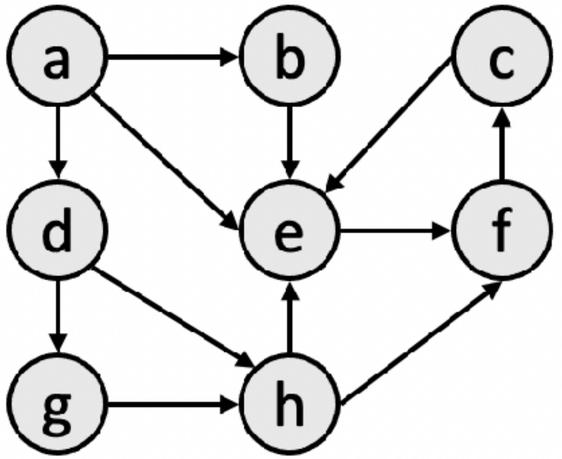


Slide credits: Marty Stepp, UW

Depth First Search

depth-first search (DFS): Finds a path between two vertices by exploring each possible path as far as possible before backtracking.

- Often implemented recursively.
- Many graph algorithms involve *visiting* or *marking* vertices.



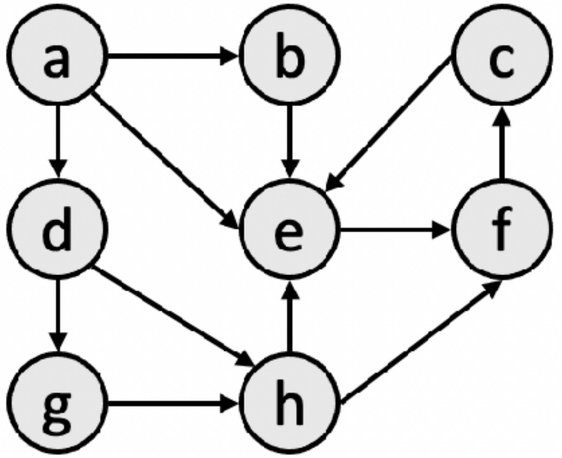
Potential Issue?

Slide credits: Marty Stepp, UW

Breadth First Search

breadth-first search (BFS): Finds a path between two nodes by taking one step down all paths and then immediately backtracking.

- Often implemented by maintaining a queue of vertices to visit.

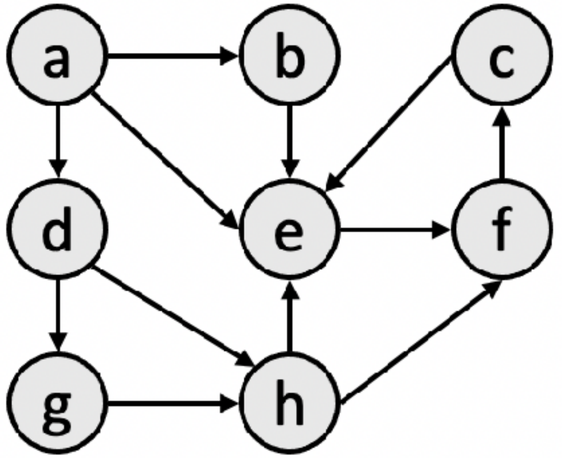


Slide credits: Marty Stepp, UW

Breadth First Search

breadth-first search (BFS): Finds a path between two nodes by taking one step down all paths and then immediately backtracking.

- Often implemented by maintaining a queue of vertices to visit.



Potential Issue?

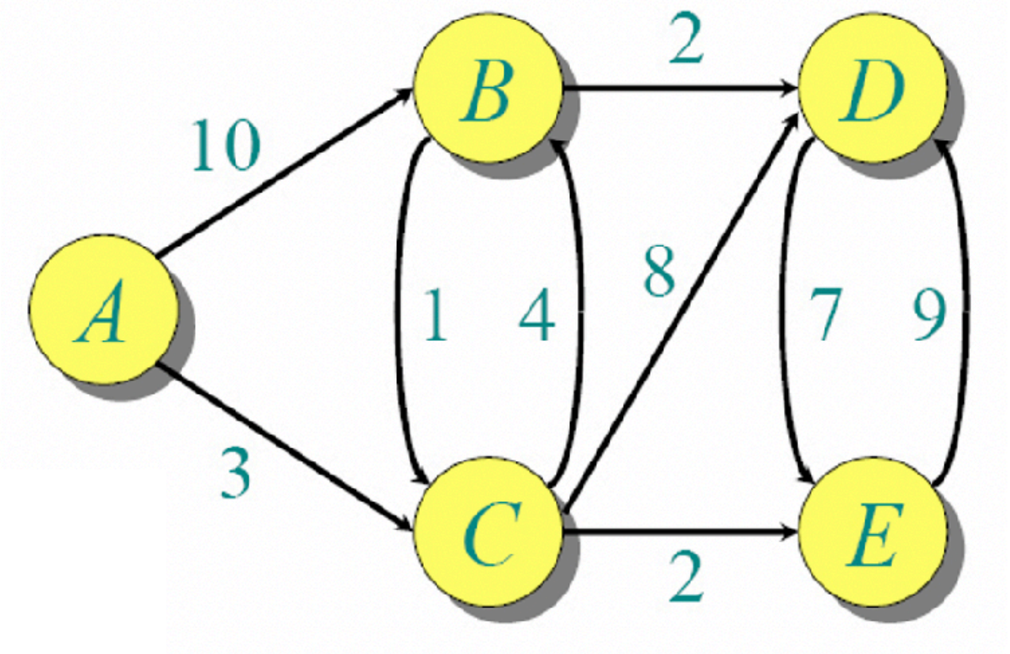
Slide credits: Marty Stepp, UW

Dijkstra's Algorithm

"Computer Science is no more about computers than astronomy is about telescopes."

Slide credits: Lakshman Veeravagu and Luis Barrera

Dijkstra's Algorithm



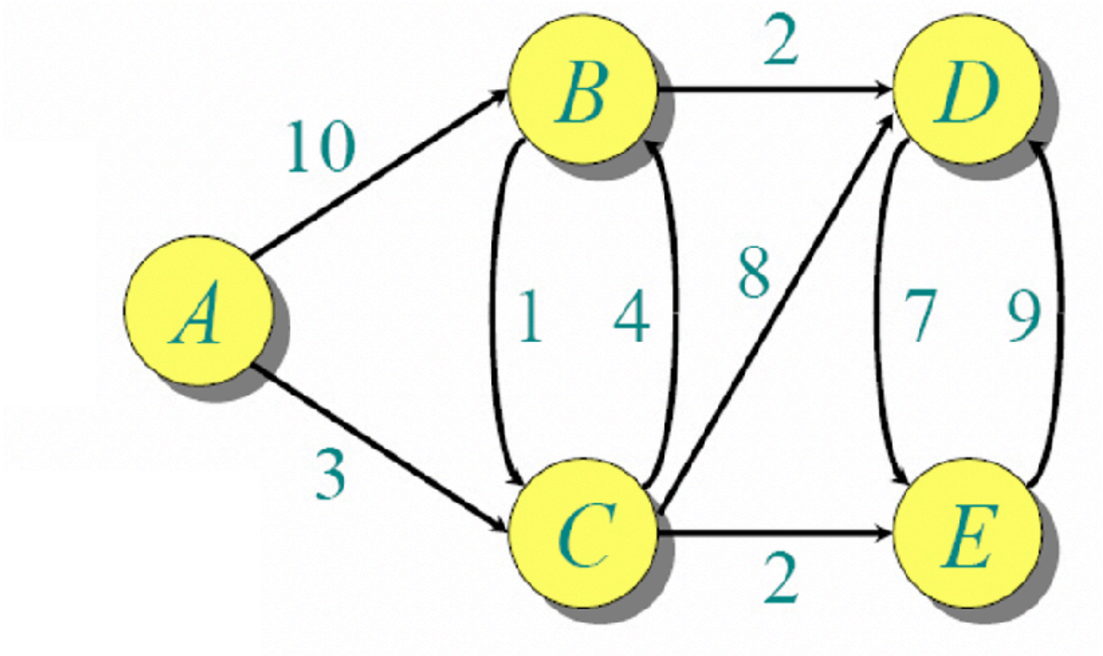
Slide credits: Lakshman Veeravagu and Luis Barrera

Dijkstra's Algorithm

```
dist[s] ←o                          (distance to source vertex is zero)
for all v ∈ V–{s}
      do dist[v] ←∞                 (set all other distances to infinity)
S←∅                                 (S, the set of visited vertices is initially empty)
Q←V                                 (Q, the queue initially contains all vertices)
while Q ≠∅                          (while the queue is not empty)
do  u ← mindistance(Q,dist)         (select the element of Q with the min. distance)
    S←S∪{u}                         (add u to list of visited vertices)
     for all v ∈ neighbors[u]
          do if  dist[v] > dist[u] + w(u, v)        (if new shortest path found)
                then     d[v] ←d[u] + w(u, v)      (set new value of shortest path)
                  (if desired, add traceback code)
return dist
```

Slide credits: Lakshman Veeravagu and Luis Barrera

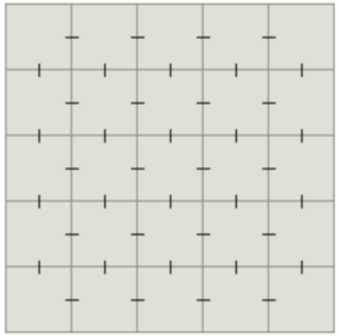
Dijkstra's Algorithm

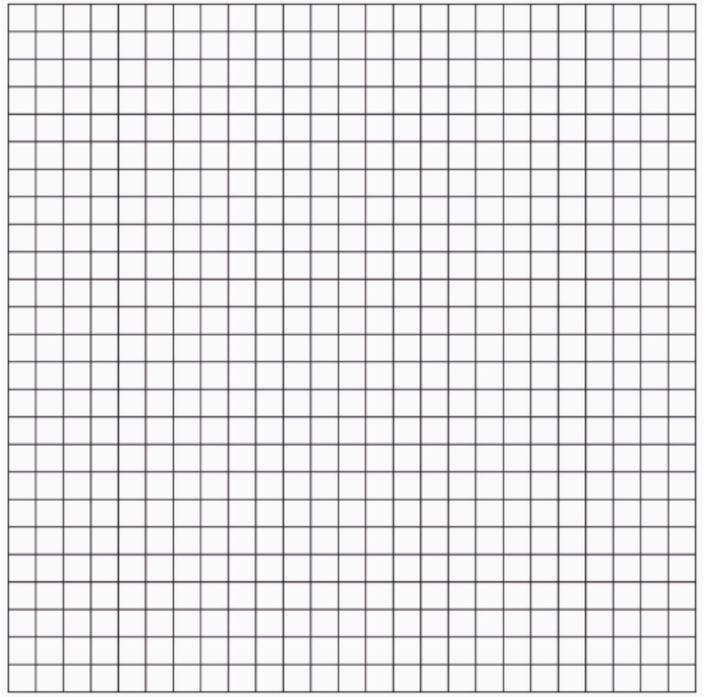


Potential Issue?

Slide credits: Lakshman Veeravagu and Luis Barrera

Dijkstra's Algorithm

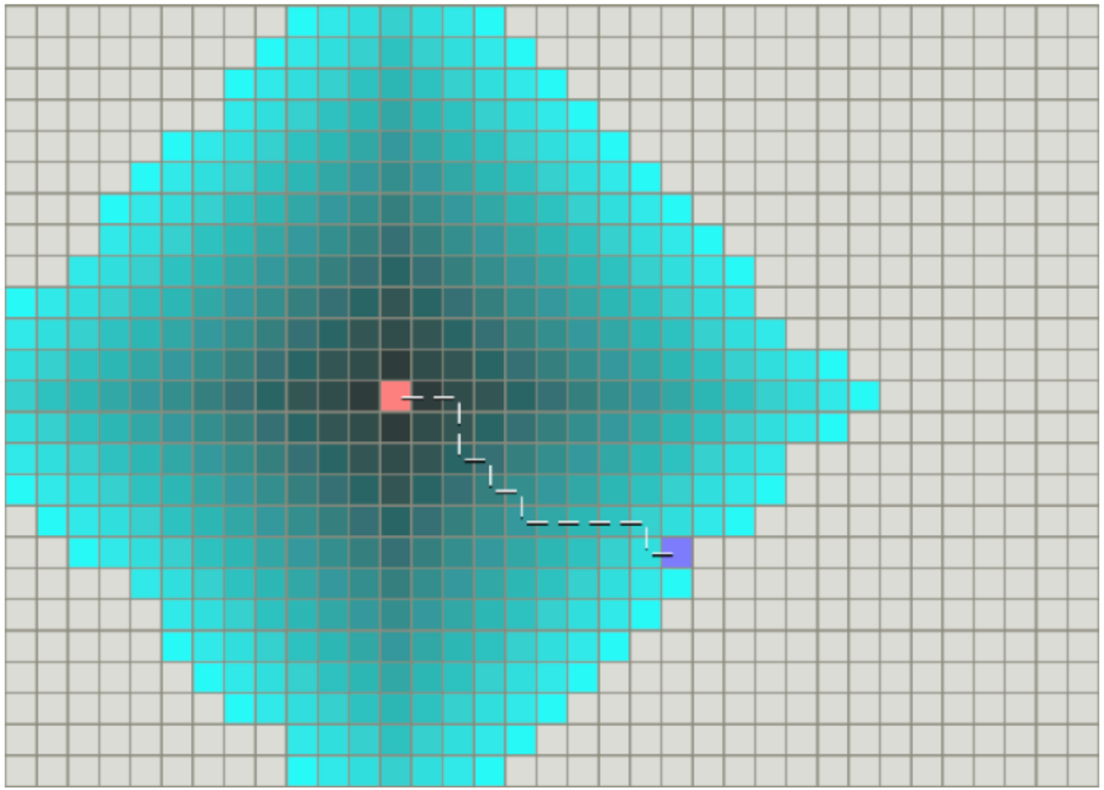




Potential Issue?

Slide credits: Lakshman Veeravagu and Luis Barrera

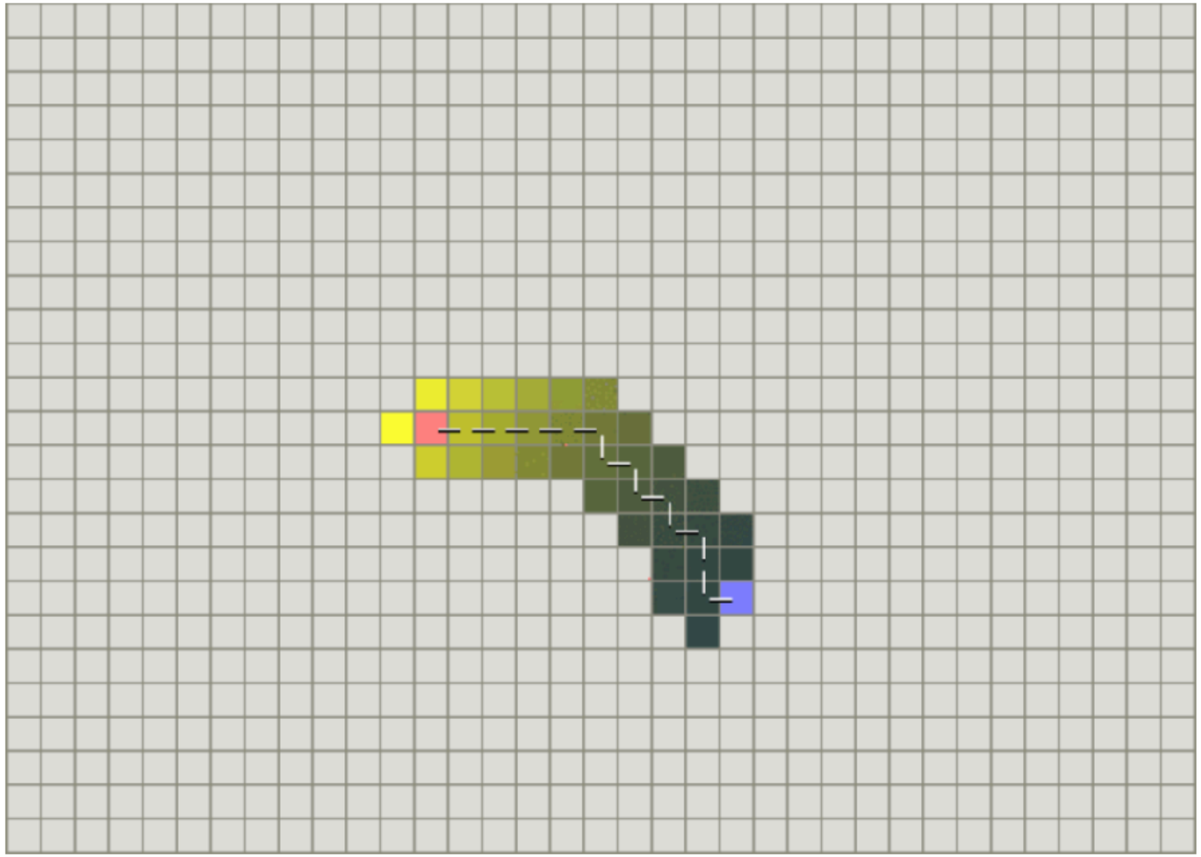
Dijkstra's Algorithm



$$f(n) = g(n)$$

Slide credits: Amit Patel

Greedy Best-First-Search

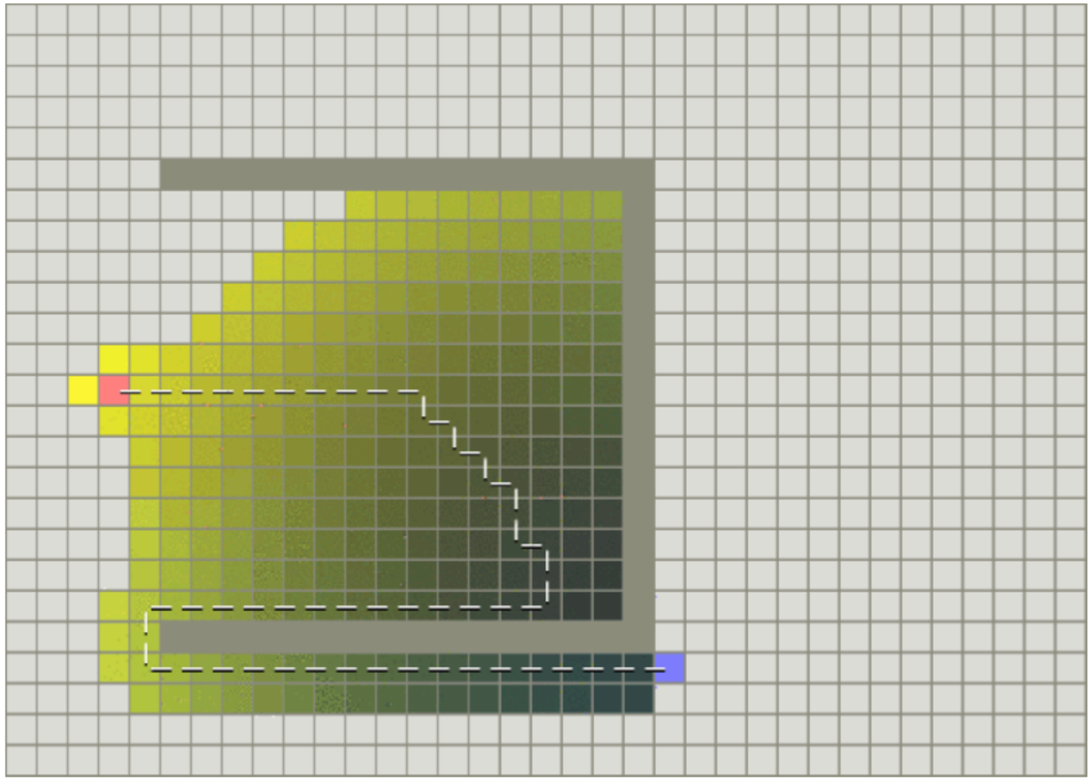


Expand node ‘closest’ to the goal

$$f(n) = h(n)$$

Slide credits: Amit Patel

Greedy Best-First-Search

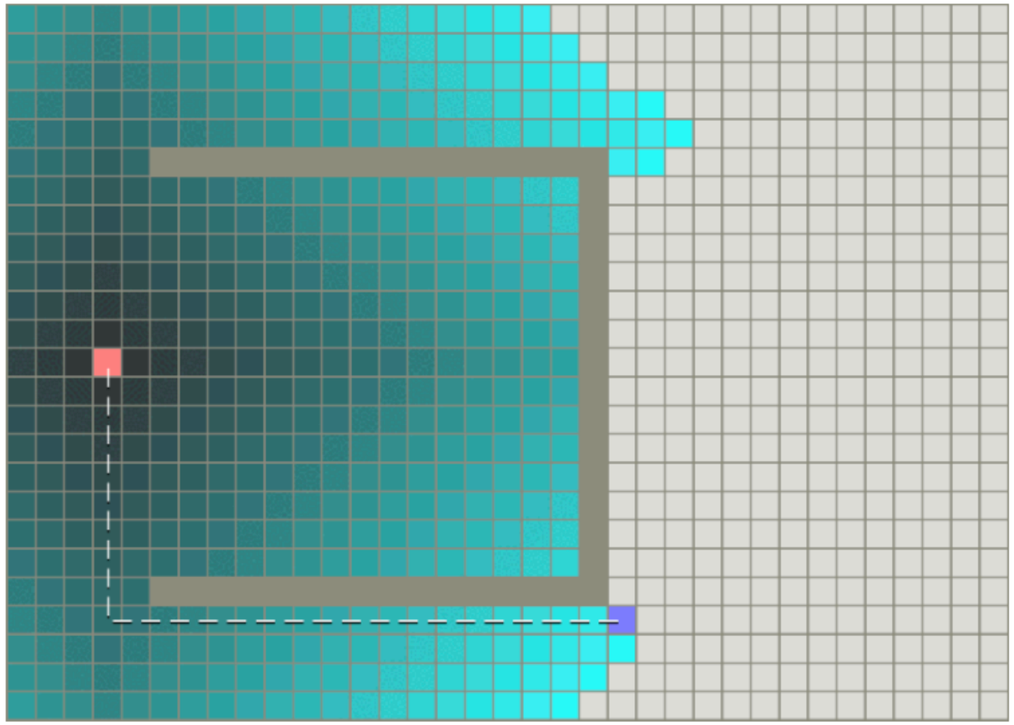


Expand node ‘closest’ to the goal

$$f(n) = h(n)$$

Slide credits: Amit Patel

Dijkstra's



$$f(n) = g(n)$$

Slide credits: Amit Patel

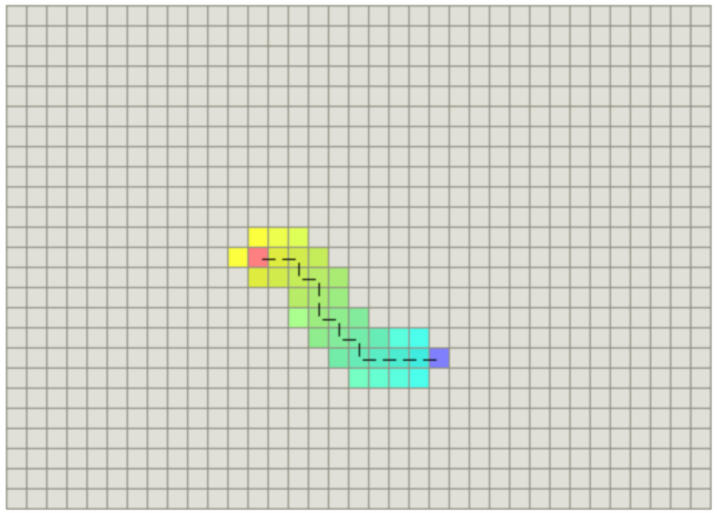
A* = Dijkstra's + Heuristic

$$f(n) = g(n) + h(n)$$

Slide credits: Amit Patel

A* = Dijkstra's + Heuristic

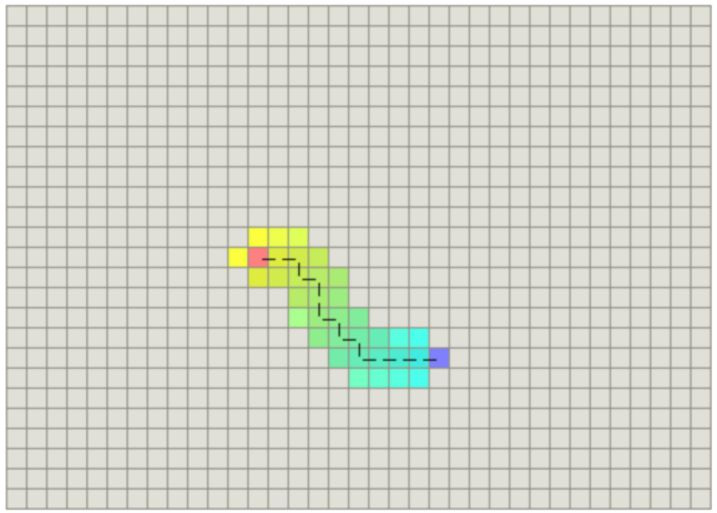
$$f(n) = g(n) + h(n)$$

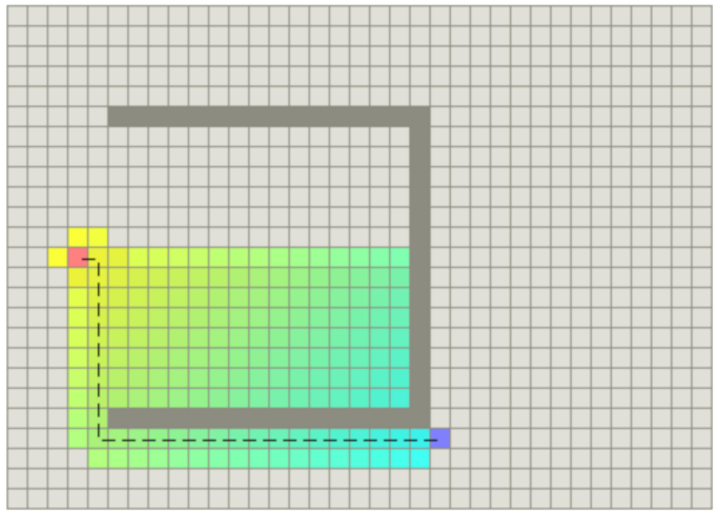


Slide credits: Amit Patel

A* = Dijkstra's + Heuristic

$$f(n) = g(n) + h(n)$$





Slide credits: Amit Patel

Demo

https://qiao.github.io/PathFinding.js/visual/