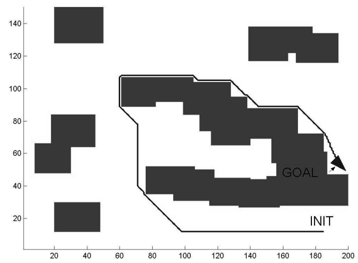
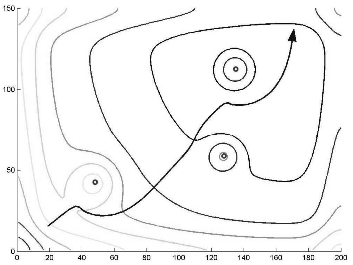
Robot Motion Planning





CSE398/498-011
23 Jan 05

Supporting References

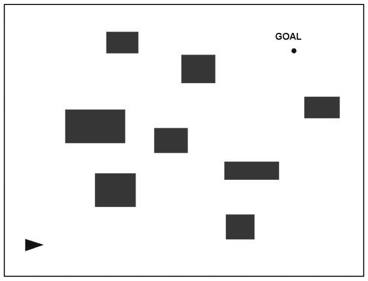
- *"Motion Planning Using Potential Fields,"* R. Beard & T. McClain, BYU, 2003
- You should download this from the course page and *read* it

Let’s Assume…

- We have an *a priori* map of the environment OR
- We have sufficient sensor information to reconstruct the environment

The Basic Motion Planning Problem

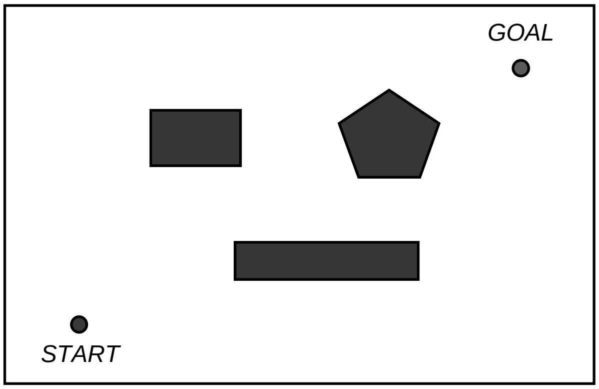
- Given an initial robot position and orientation in a potentially *cluttered workspace **C***, how should the robot move in order to reach an objective pose?



Class Objectives

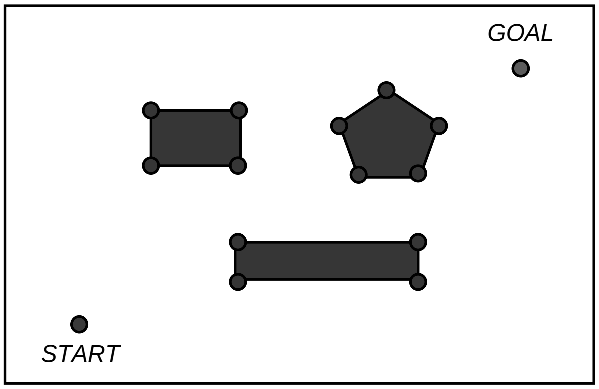
- Examine alternate approaches to motion planning
- Roadmap Approach:
  - Visibility Graph Methods
- Cell Decomposition:
  - Exact Decomposition
  - Approximate: Uniform discretization & quadtree approaches
- Potential Fields

1. The Visibility Graph Method



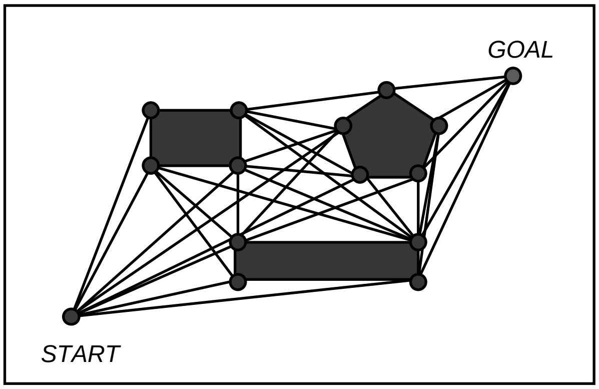
1) Model obstacles as polygons

The Visibility Graph Method (cont’d)



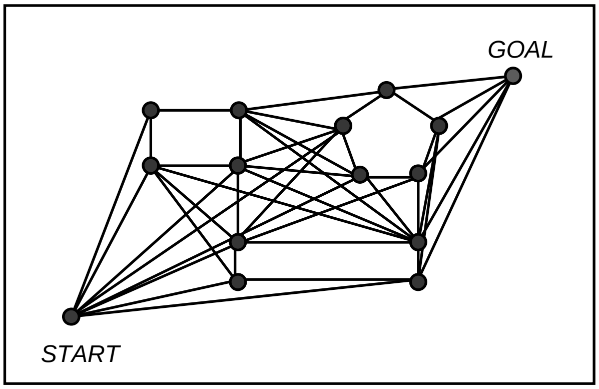
2) Construct a graph G(V,E). All polygon vertices are added to V, as are the start and goal positions.

The Visibility Graph Method (cont’d)



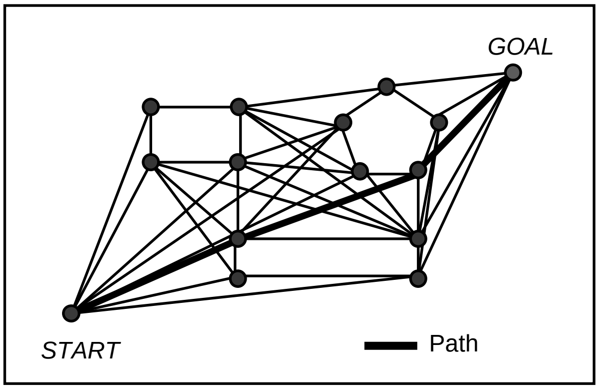
3) All vertices that are visible to one another are connected with an edge. These edges are added to E.

The Visibility Graph Method (cont'd)



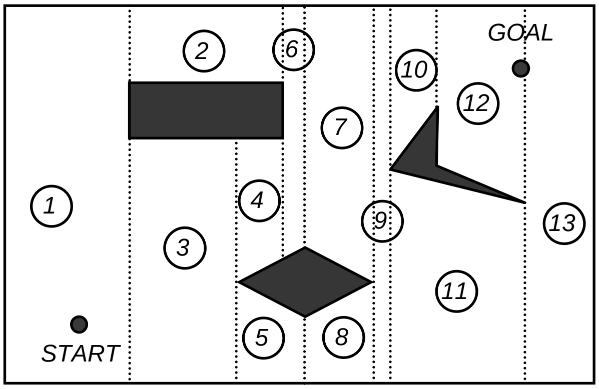
4) Polygon edges are also added to E. Then we only need to find the shortest path from the start vertex to the goal vertex in G. How can we find this???

The Visibility Graph Method (cont'd)



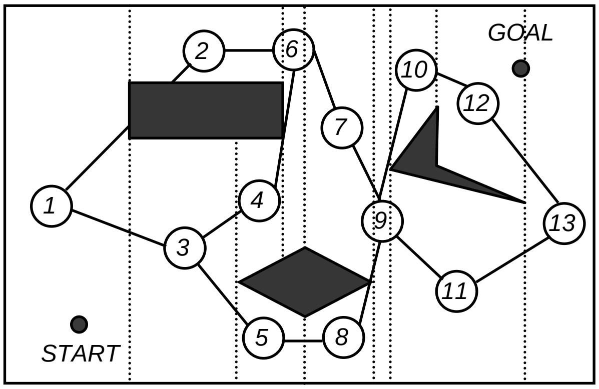
We can find the shortest path using Dijkstra's Algorithm!!!

2. Exact Cell Decomposition Method



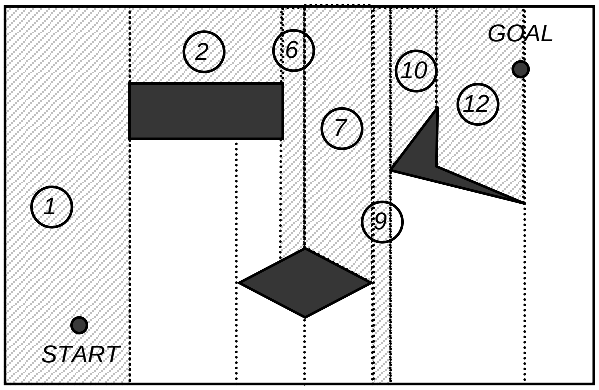
1) Decompose Region Into Cells

Exact Cell Decomposition Method (cont'd)



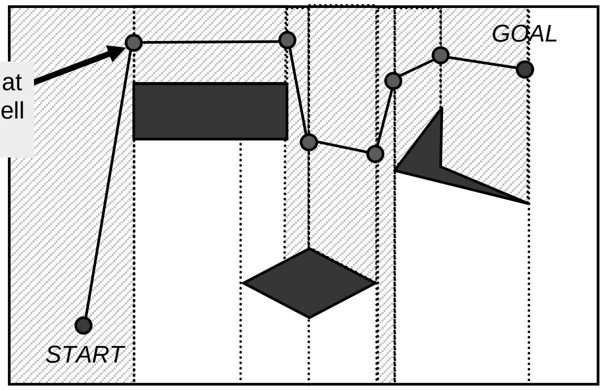
2) Construct Adjacency Graph

Exact Cell Decomposition Method (cont'd)



3) Construct Channel from shortest cell path (via Depth-First-Search)

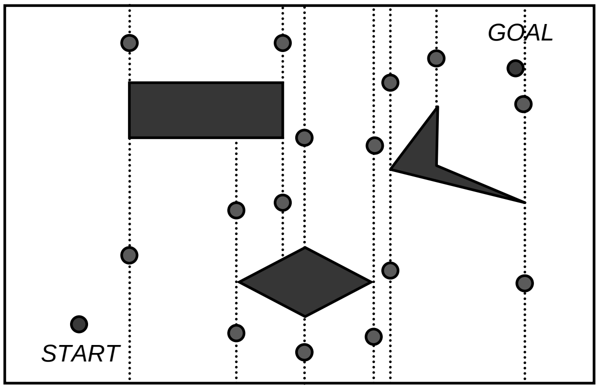
Exact Cell Decomposition Method (cont'd)



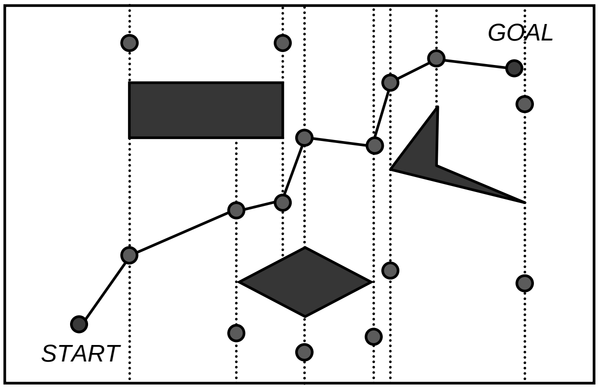
Nodes placed at the center of cell boundary.

4) Construct Motion Path **P** from channel cell borders

2b. Exact Cell Decomposition Method Using Euclidean Metric



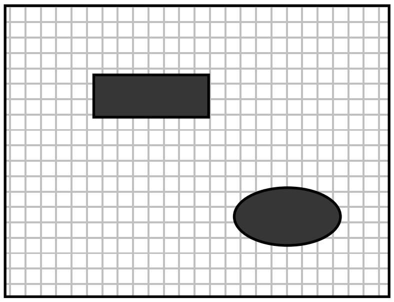
Exact Cell Decomposition Method Using Euclidean Metric



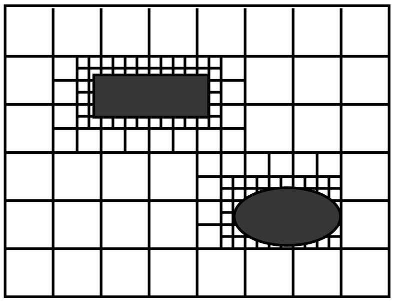
3. Approximate Cell Decomposition

1. Perform cell tessellation of configuration space ***C***
   - Uniform or quadtree
2. Generate the cell graph *G(V,E)*
   - Each cell $v \in C_{free} \subseteq C$ corresponds to a vertex in ***V***
   - Two vertices $v_i, v_j \in$ ***V*** are connected by an edge e_{ij} if they are adjacent (8-connected for exact)
   - Edges are weighted by Euclidean distance
3. Find the shortest path from $\boldsymbol{v_{init}}$ to $\boldsymbol{v_{goal}}$

Step 1: Cell Decomposition



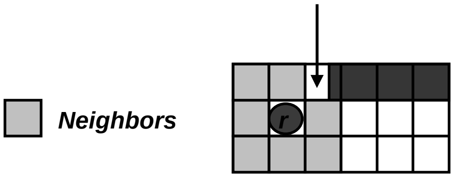
Uniform



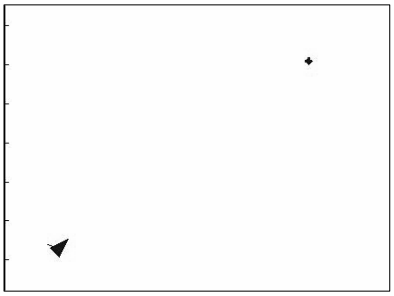
Quadtree

Cell Decomposition Issues

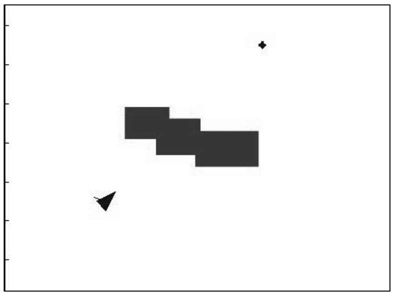
1. Continuity of trajectory a function of resolution
2. Computational complexity increases dramatically with resolution
3. Inexactness. Is this cell an obstacle or in ***C**$_{free}$*?



Cell Decomposition Simulations

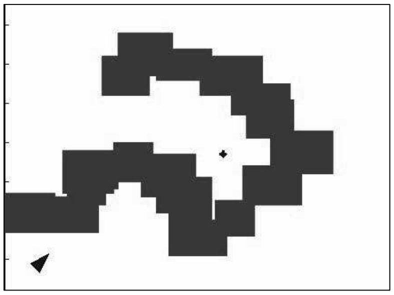


No Obstacles

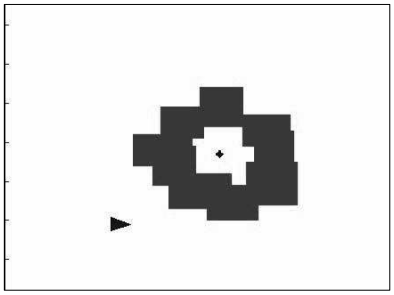


Single Obstacle

Cell Decomposition Simulations



Multiple Obstacles



No Path

Approximate Cell Summary

- PROS
  - Applicable to general obstacle geometries
  - Provides shorter paths than exact decomposition
- CONS
  - Performance a function of discretization resolution (δ)
    - Inefficiencies
    - Lost paths
    - Undetected collisions
  - Number of graph vertices |V| grows with δ^2 and Dijkstra's runs in O(|E| lg |V|) (an A* implementation in O(V^2))

The Potential Field Approach

Some Background

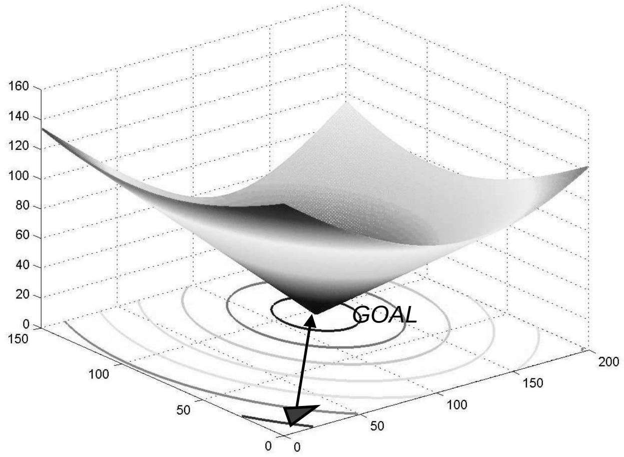
- Introduced by Khatib [1986] initially as a real-time collision avoidance module, and later extended to motion planning
- Robot motion is influenced by an artificial *potential field* – a force field if you will – induced by the goal and obstacles in the robot's configuration space ***C*** (all of the possible positions of the robot)
- The field is modeled by a *potential function* ***E(x,y)*** over ***C***

Some Background (cont'd)

- Motion policy control law is akin to gradient descent on the potential function
- A shortcoming of the approach is the lack of performance guarantees: the robot can become trapped in local minima
- Koditschek's extension introduced the concept of a "navigation function" - a local-minima free potential function

Example Application: Target Tracking with Obstacle Avoidance

The Idea…



Flashback: What is the Gradient?

- In 2D, the gradient of a function *f* is defined as

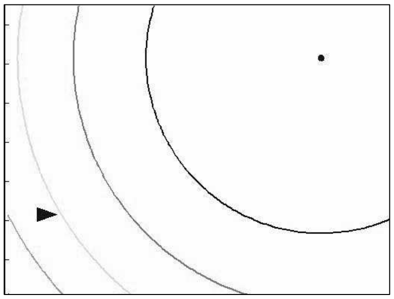
$$\nabla f = \frac{\partial f}{\partial x}\hat{x} + \frac{\partial f}{\partial y}\hat{y}$$

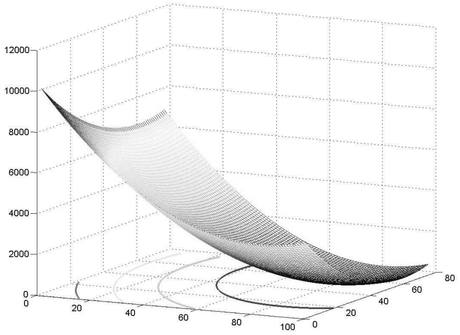
- The gradient points in the direction where the derivative has the largest value (the greatest rate of increase in the value of *f*)
- The *gradient descent* optimization algorithm searches in the *opposite* direction of the gradient to find the *minimum* of a function
- Potential field methods employ a similar approach

Some Characteristics of Potential Field Approaches

- Potential fields can live in continuous space
  - No cell decomposition issues
- Local method
  - Implicit trajectory generation
  - Prior knowledge of obstacle positions not required
- The bad: Weaker performance guarantees

Generating the Potential Field A Parabolic Well for Attracting to Goal





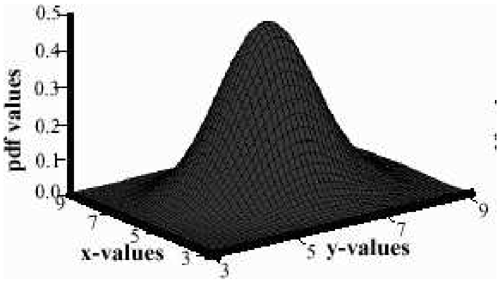
$$E(x) = \frac{1}{2}(x - x_{goal})^T (x - x_{goal})$$

$$\nabla E(x) = x - x_{goal}$$

$$x = \begin{bmatrix} x_w \\ y_w \end{bmatrix}$$

NOTE: x is a vector corresponding to a position in the workspace

Gaussian Obstacle Potential Function

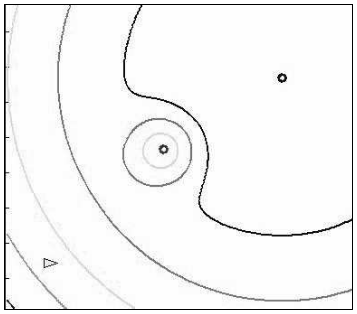


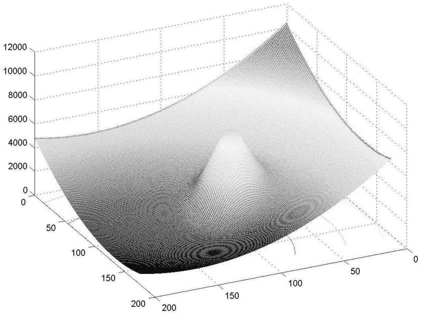
$$f(x) = \frac{1}{2\pi\sqrt{\sigma_x^2\sigma_y^2}} e^{-\left(\frac{(x-\mu)^2}{2\sigma_x^2}+\frac{(y-\mu)^2}{2\sigma_y^2}\right)}$$

For a symmetric
2D Gaussians

$$f(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}\|\bar{x}\|^2} = \beta e^{-\frac{\gamma}{2}\|\bar{x}\|^2}$$

Parabolic Well for Goal
Exponential Source for Obstacle

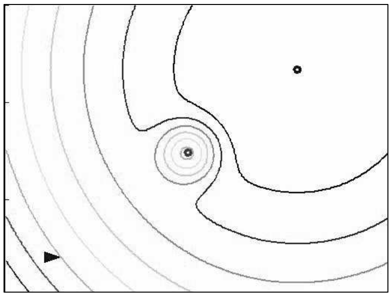


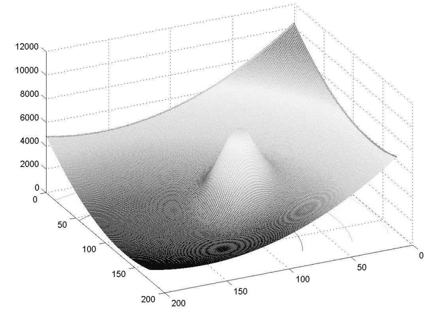


$$E(x) = \frac{1}{2}(x - x_{goal})^T(x - x_{goal}) + \beta e^{\frac{-\gamma}{2}\|x - x_{obs}\|}$$

$$\nabla E(x) = x - x_{goal} - \gamma(x - x_{obs})\beta e^{\frac{-\gamma}{2}\|x - x_{obs}\|^2}$$

Parabolic Well /Exponential Source Unstable Equilibrium Example

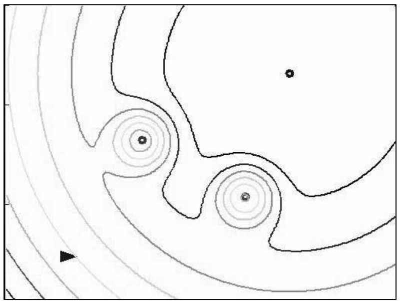


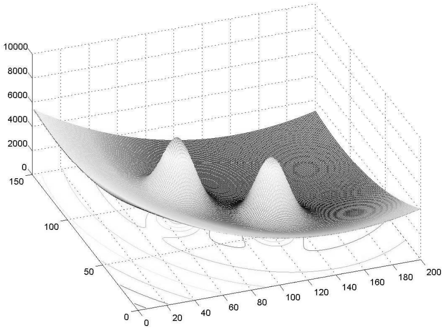


$$E(x) = \frac{1}{2}(x - x_{goal})^T (x - x_{goal}) + \beta e^{\frac{-\gamma}{2}\|x - x_{obs}\|^2}$$

$$\nabla E(x) = x - x_{goal} - \gamma(x - x_{obs})\beta e^{\frac{-\gamma}{2}\|x - x_{obs}\|^2}$$

Parabolic Well Goal & Two Exponential Source Obstacles

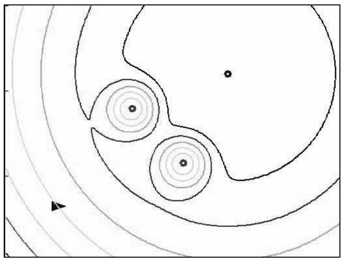


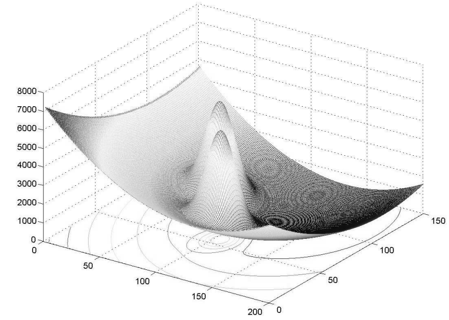


$$E(x) = \frac{1}{2}(x - x_{goal})^T (x - x_{goal}) + \sum_{i=1}^{n} \beta_i e^{\frac{-\gamma_i}{2}\|x - x_{obs(i)}\|^2}$$

$$\nabla E(x) = x - x_{goal} - \sum_{i=1}^{n} \gamma_i (x - x_{obs(i)})\beta_i e^{\frac{-\gamma_i}{2}\|x - x_{obs(i)}\|^2}$$

Parabolic Well Goal & Two Exponential Source Obstacles

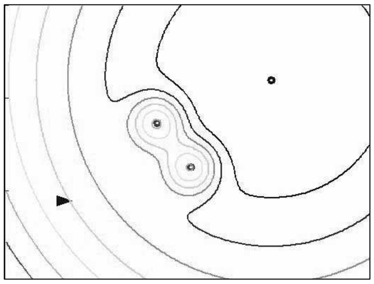


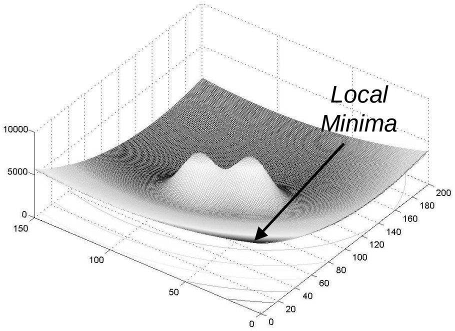


$$E(x) = \frac{1}{2}(x - x_{goal})^T (x - x_{goal}) + \sum_{i=1}^{n} \beta_i e^{\frac{-\gamma_i}{2}\left\| x - x_{obs(i)} \right\|^2}$$

$$\nabla E(x) = x - x_{goal} - \sum_{i=1}^{n} \gamma_i (x - x_{obs(i)}) \beta_i e^{\frac{-\gamma_i}{2}\left\| x - x_{obs(i)} \right\|^2}$$

Parabolic Well Goal & Two Exponential Source Obstacles

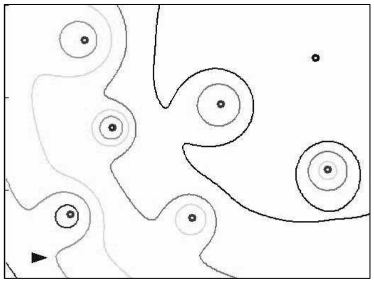


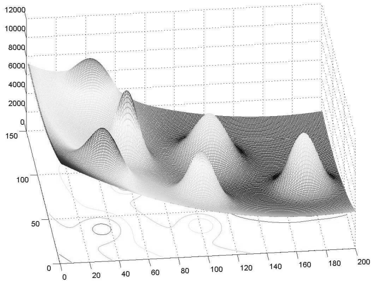


$$E(x) = \frac{1}{2}(x - x_{goal})^T (x - x_{goal}) + \sum_{i=1}^{n} \beta_i e^{\frac{-\gamma_i}{2}\left\| x - x_{obs(i)} \right\|^2}$$

$$\nabla E(x) = x - x_{goal} - \sum_{i=1}^{n} \gamma_i (x - x_{obs(i)}) \beta_i e^{\frac{-\gamma_i}{2}\left\| x - x_{obs(i)} \right\|^2}$$

Parabolic Well Goal & Multiple Exponential Source Obstacles

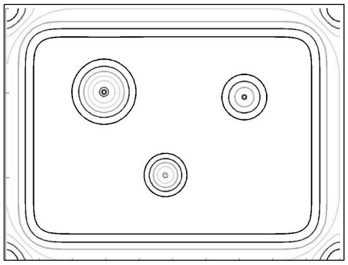


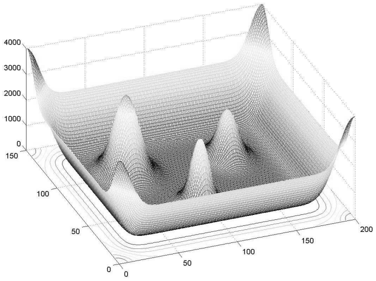


$$E(x)=\frac{1}{2}(x-x_{goal})^T(x-x_{goal})+\sum_{i=1}^{n}\beta_i e^{\frac{-\gamma_i}{2}\left\|x-x_{obs(i)}\right\|^2}$$

$$\nabla E(x)=x-x_{goal}-\sum_{i=1}^{n}\gamma_i(x-x_{obs(i)})\beta_i e^{\frac{-\gamma_i}{2}\left\|x-x_{obs(i)}\right\|^2}$$

Modeling Walls in a Closed Workspace



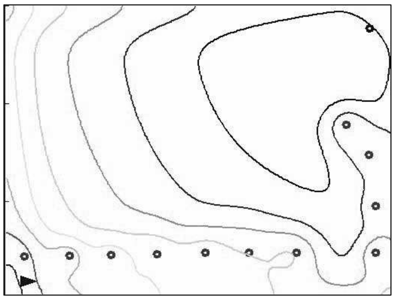


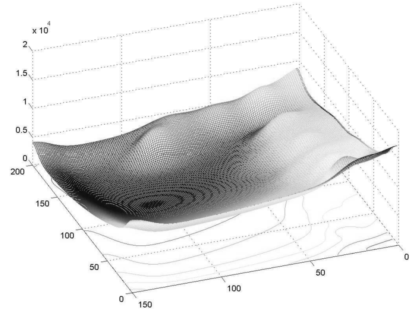
WARNING:
NOTATION ABUSE
(x & y are scalars on RHS)

$$E_{wall}(\vec{x})=\alpha e^{\frac{-\gamma}{2}(y-YMAX)^2}+\alpha e^{\frac{-\gamma}{2}y^2}+\alpha e^{\frac{-\gamma}{2}(x-XMAX)^2}+\alpha e^{\frac{-\gamma}{2}x^2}$$

$$\nabla E(\vec{x})=-\gamma(y-YMAX)\alpha e^{\frac{-\gamma}{2}(y-YMAX)^2}-\gamma y\alpha e^{\frac{-\gamma}{2}y^2}+\gamma(x-XMAX)\alpha e^{\frac{-\gamma}{2}(x-XMAX)^2}+\gamma x\alpha e^{\frac{-\gamma}{2}x^2}$$

A Summary Example





Issues with
Reactive Approaches Presented

- Obstacles are not points
  - Model as points
  - Bound with ellipses (one or more obstacles)
- Local minima proliferate with multiple obstacles, and failure to achieve goal often results in global motion planning task
  - Combine with global/discrete approaches
    - Best first search
    - Wavefront propagation
    - Voronoi

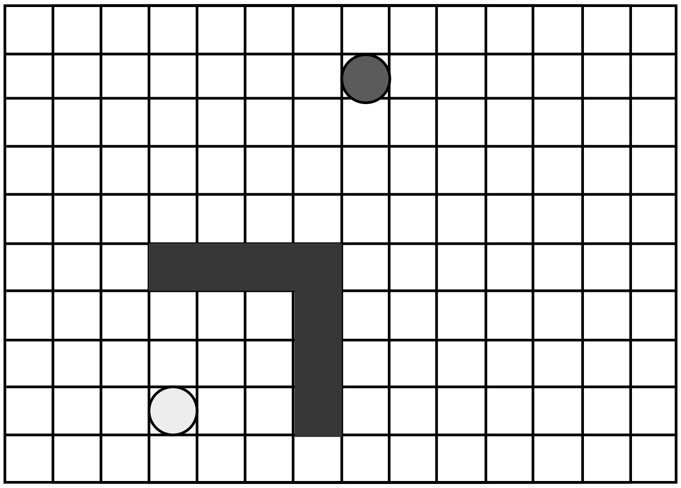
Summary

- We examined 2 different approaches to motion planning
- In our application, *C* is fixed BUT C_{free} changes over time
- Often, only local sensing information is available so it may not be necessary to calculate a complete path to the goal
- Potential fields are computationally inexpensive, as you only need to compute the potential *local* to the robot

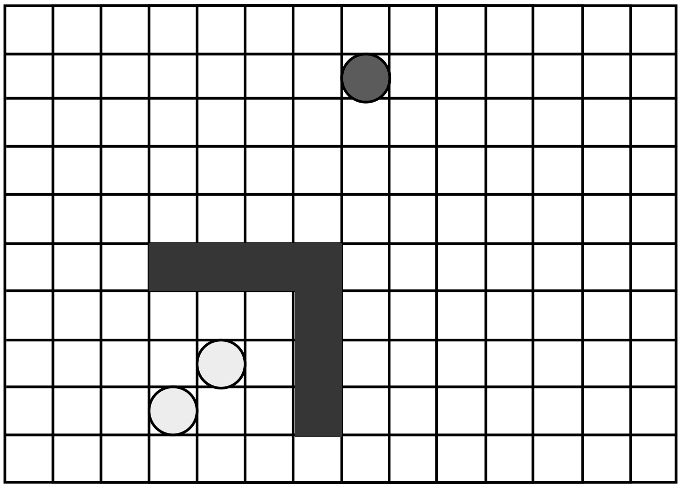
Best First Search

1. Workspace discretized into cells
2. Insert (x_{init},y_{init}) into list OPEN
3. Find all 4-way neighbors to (x_{init},y_{init}) that have *not been previously visited* and insert into OPEN
4. Sort neighbors by minimum potential
5. Form paths from neighbors to (x_{init},y_{init})
6. Delete (x_{init},y_{init}) from OPEN
7. (x_{init},y_{init}) = minPotential(OPEN)
8. GOTO 2 until (x_{init},y_{init})=goal (SUCCESS) or OPEN empty (FAILURE)

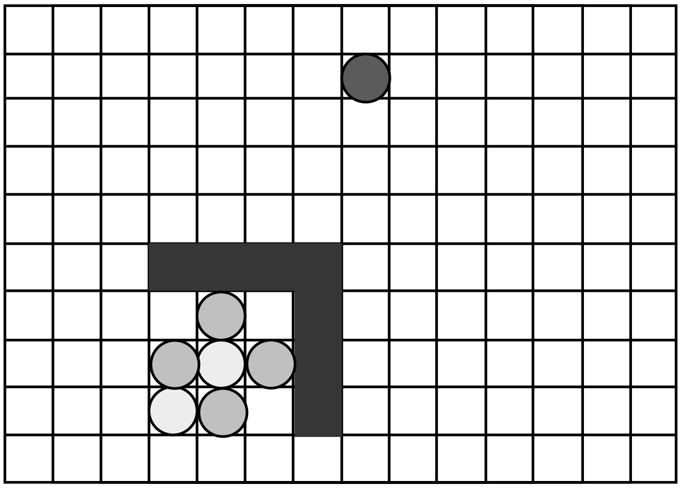
Best First Search Example



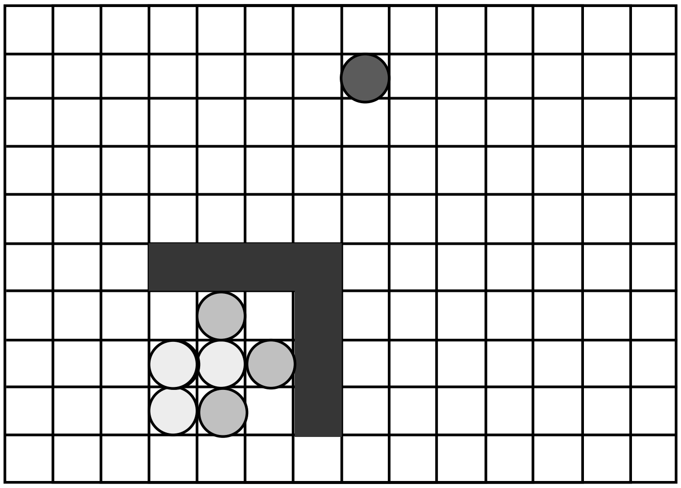
Best First Search Example



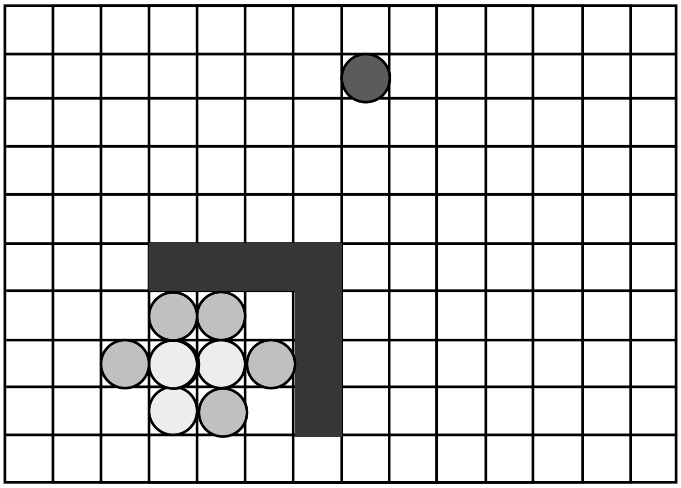
Best First Search Example



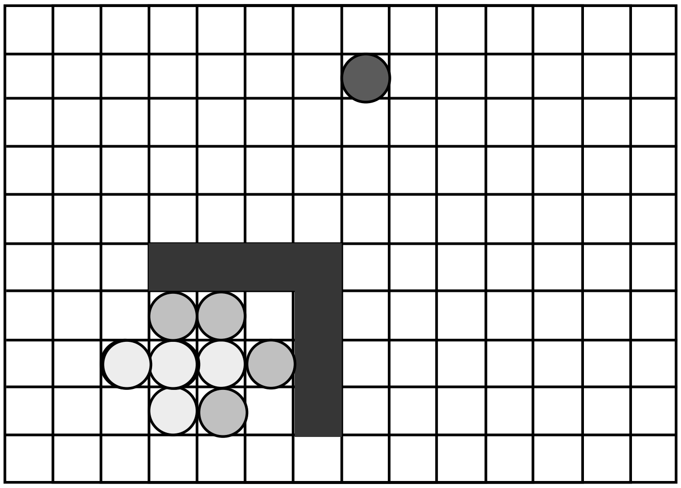
Best First Search Example



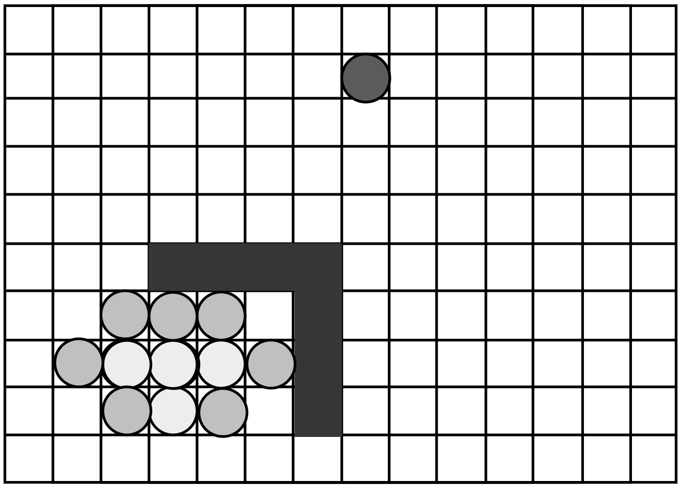
Best First Search Example



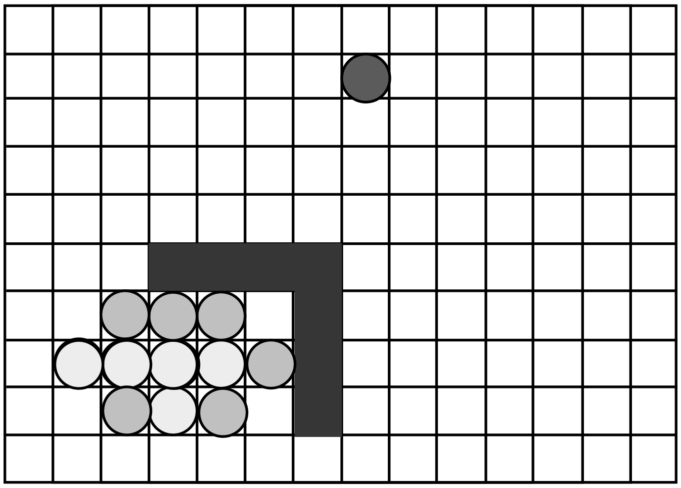
Best First Search Example



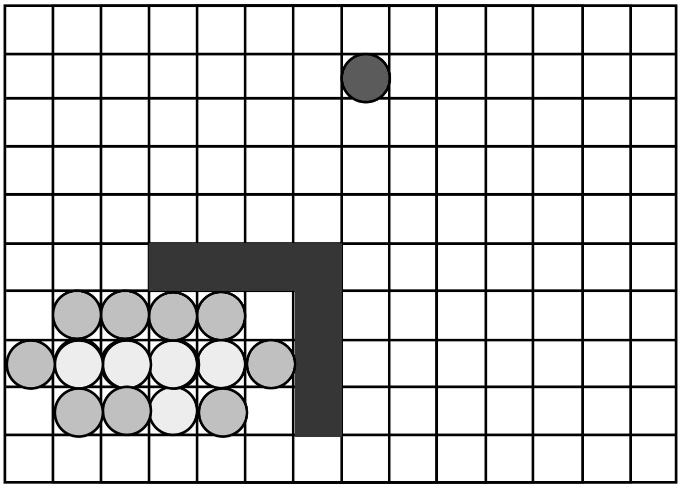
Best First Search Example



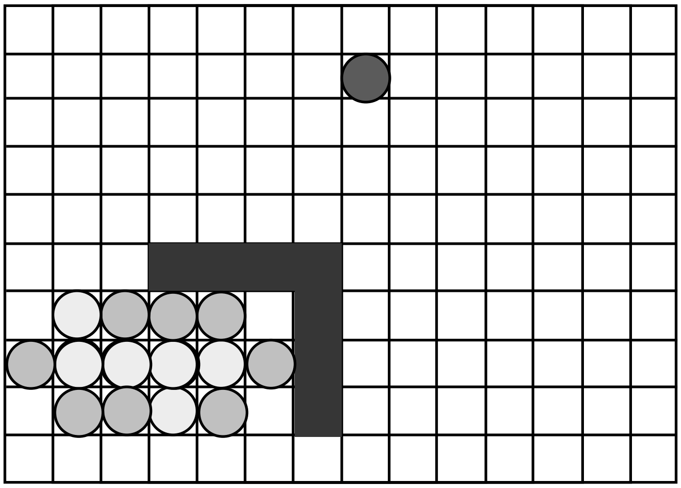
Best First Search Example



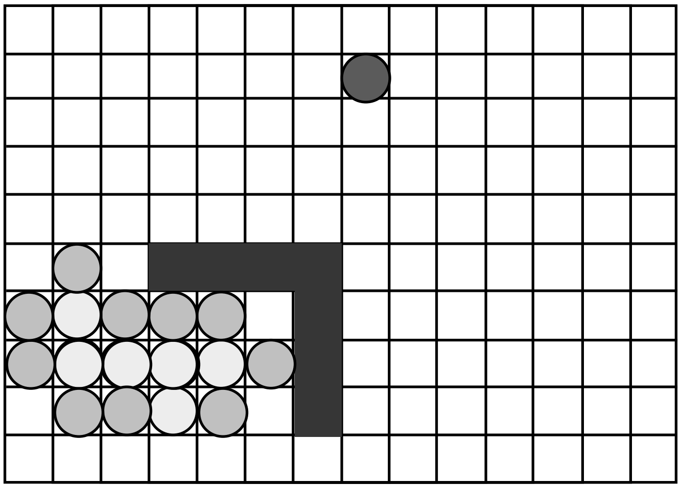
Best First Search Example



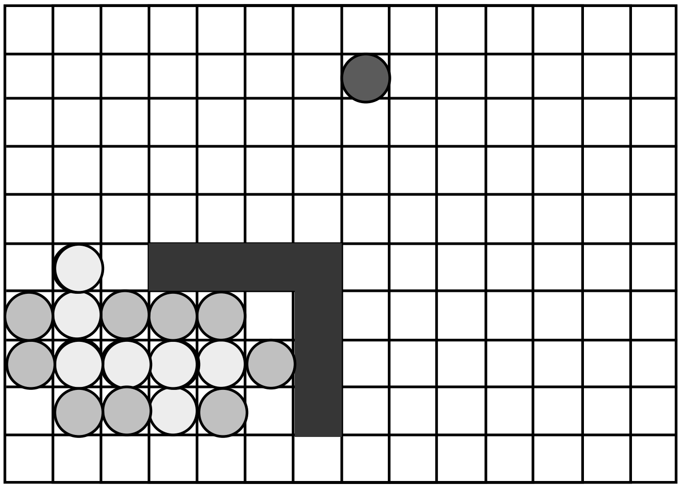
Best First Search Example



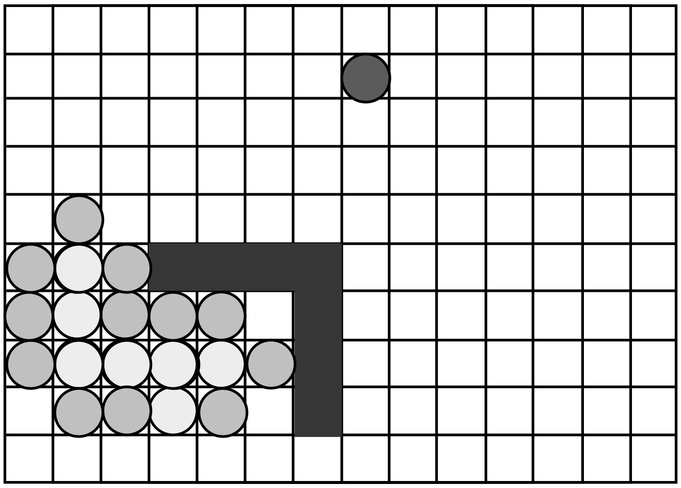
Best First Search Example



Best First Search Example



Best First Search Example



Best First Search Example