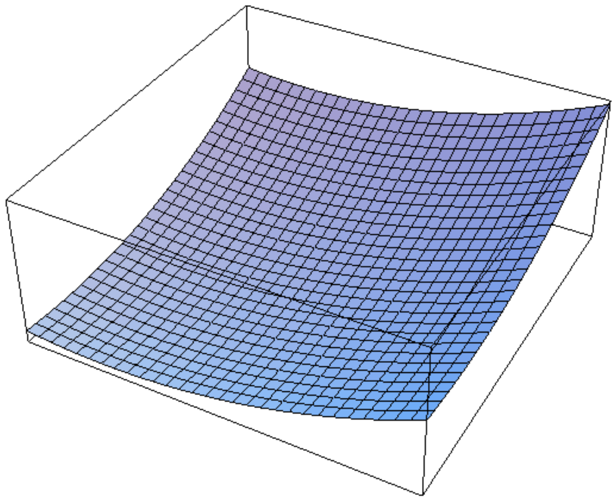
COMS W4733: Computational Aspects of Robotics

Lecture 18: Potential Fields

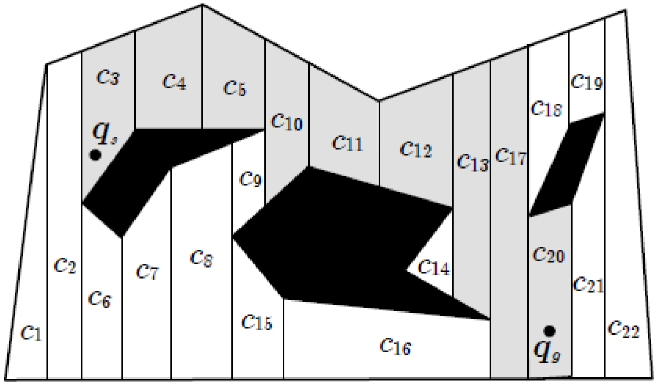


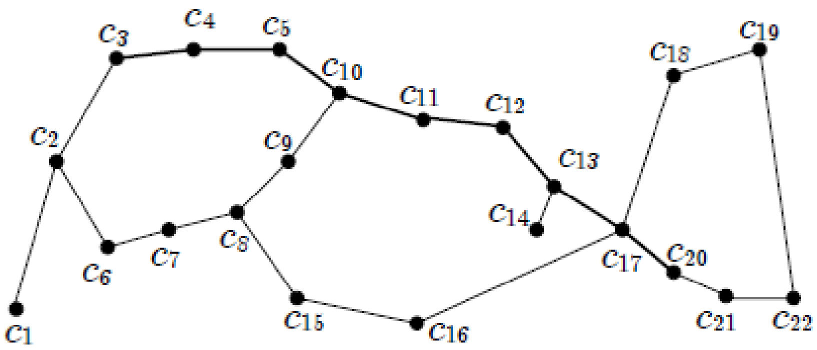
Instructor: Tony Dear

Graph-Based Path Planning

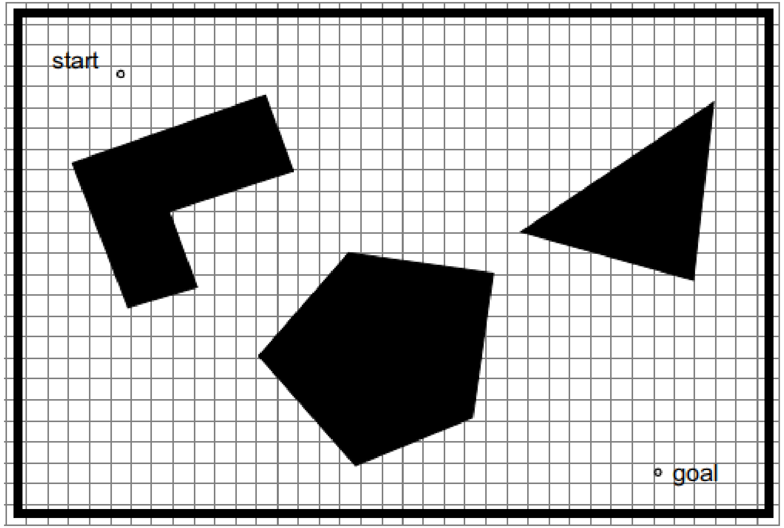
- So far, motion planning in two steps: Construct a graph, then perform search
- Roadmap planning: visibility graphs, Voronoi graphs
- **Cell decomposition** planning: connectivity graphs
  - Assumptions: “Easy” to find paths within cells and between adjacent cells
- *Exact*: Divide environment into union of convex cells (sweep line algorithm)
- *Approximate*: Choose a fixed shape (e.g. grid) or adaptively using a variable resolution as necessary

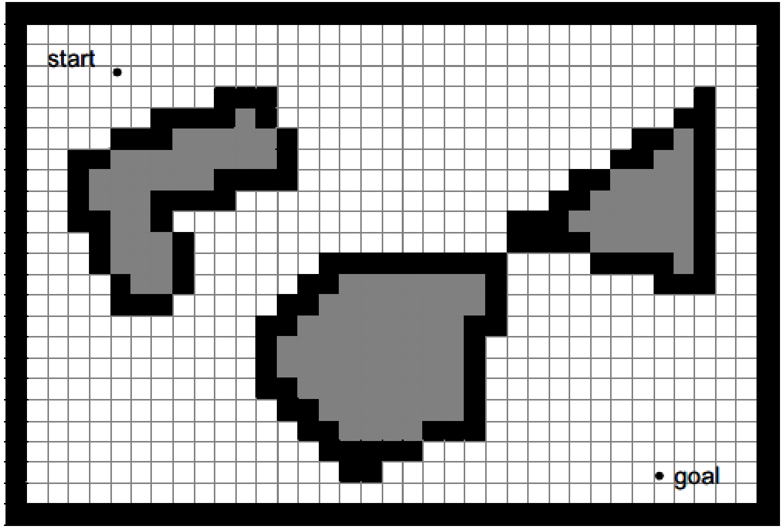
Exact Decomposition



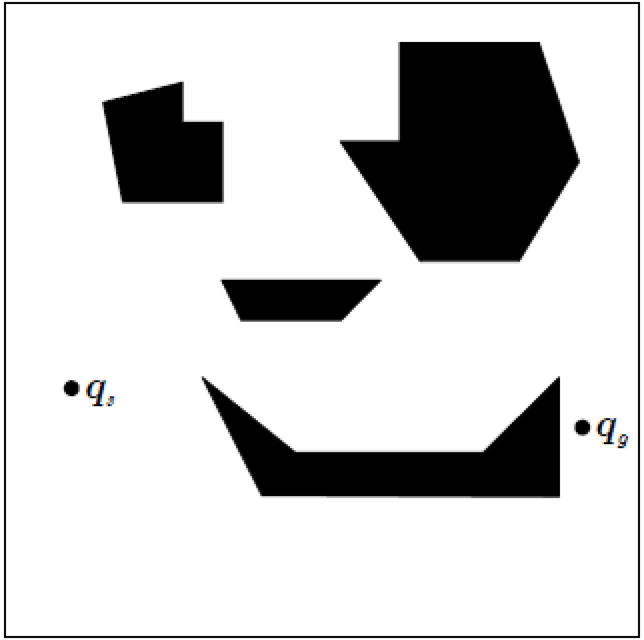


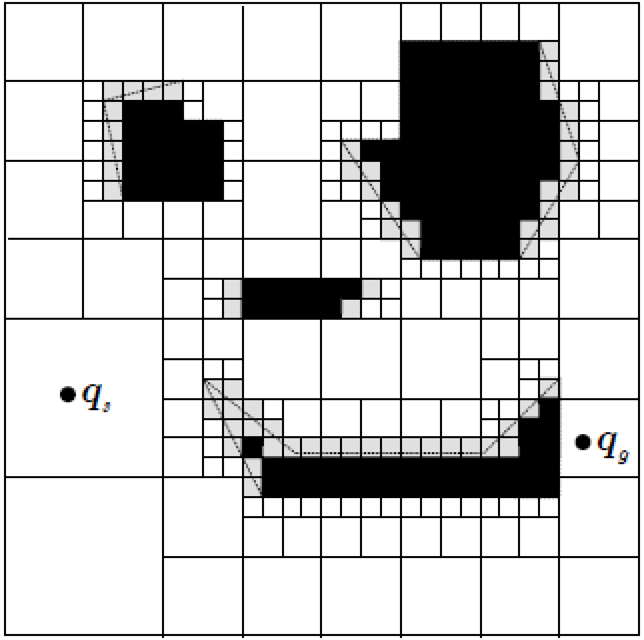
Fixed Decomposition





Adaptive Decomposition





Limitations of Graph-Based Methods

- High-dimensional problems quickly become computationally intensive
- Many environments have non-polygonal / non-convex features

- Graph search gives us a plan, but what if we deviate from it?
- Re-planning is an option, but can become expensive; *policies* are better

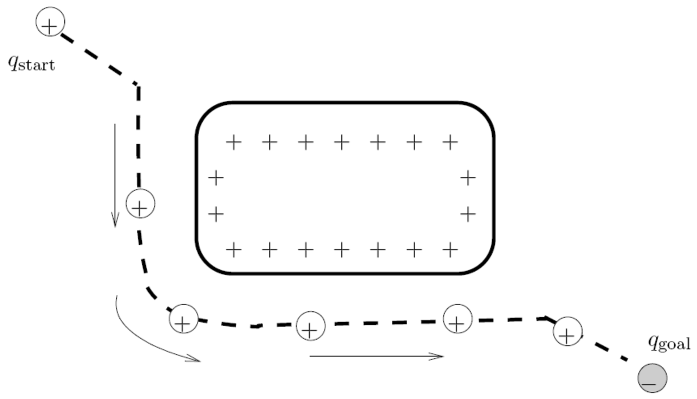
- Reactive or online approaches may be required when environments and obstacles are dynamic

Artificial Potential Fields

- Idea: Generate a *policy* everywhere in the free configuration space
- Introduce an artificial *vector field* moving the robot where we want it to go
  - Physical analogies: Electric charge, gravity wells, spring forces

- Two goals: *Attract* robot toward goal
- *Repel* robot away from obstacles

- Our job: Come up with right **potential function**



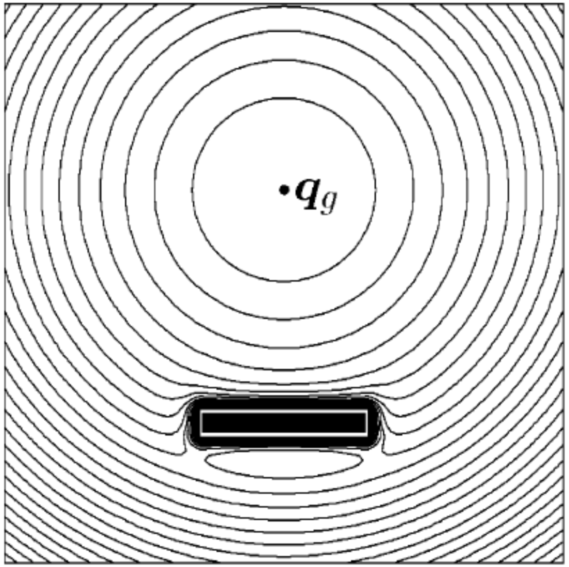
Potential Function

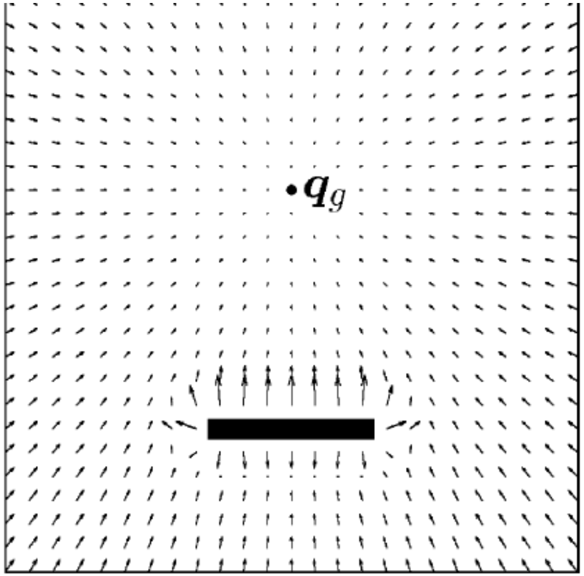
- Potential function U assigns an *energy* value $U(\boldsymbol{q})$ to each point in C-space
- Robot should constantly try to move toward direction with lower energy

- Generally constructed with *attractive* component toward goal and *repulsive* component away from obstacles: $U(\boldsymbol{q}) = U_a(\boldsymbol{q}) + U_r(\boldsymbol{q})$

- Robot tries to find global minimum of U using gradient descent, following a *force field* on the C-space: $\boldsymbol{f}(\boldsymbol{q}) = -\nabla U(\boldsymbol{q}) = -\nabla U_a(\boldsymbol{q}) - \nabla U_r(\boldsymbol{q})$

Potential Function vs Force Field





Attractive Potential

- Simple requirement: U_a monotonically increases with distance from goal

$$\boldsymbol{e}(\boldsymbol{q}) = \boldsymbol{q}_g - \boldsymbol{q}$$

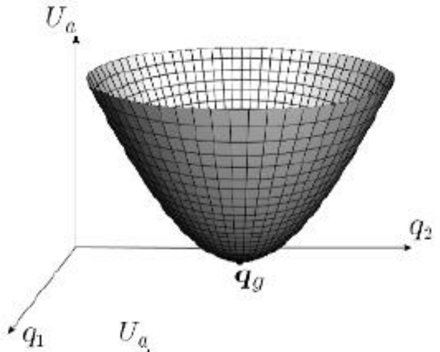
Error function

- *Parabolic* potential

$$U_a(\boldsymbol{q}) = \frac{1}{2} k_a \|\boldsymbol{e}(\boldsymbol{q})\|^2$$

$$k_a > 0$$

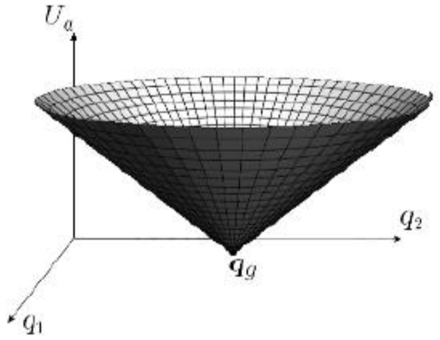
$$\boldsymbol{f}_a(\boldsymbol{q}) = -\nabla U_a(\boldsymbol{q}) = k_a \boldsymbol{e}(\boldsymbol{q})$$



- *Conical* potential

$$U_a(\boldsymbol{q}) = k_a \|\boldsymbol{e}(\boldsymbol{q})\|$$

$$\boldsymbol{f}_a(\boldsymbol{q}) = -\nabla U_a(\boldsymbol{q}) = k_a \frac{\boldsymbol{e}(\boldsymbol{q})}{\|\boldsymbol{e}(\boldsymbol{q})\|}$$



Attractive Potential

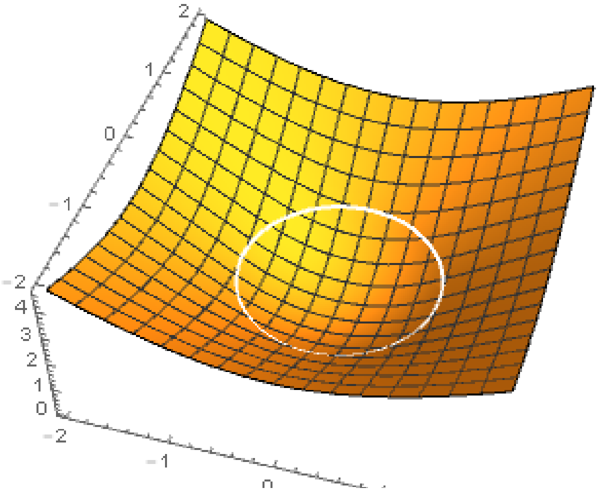
- Both parabolic and conical potentials are positive-definite away from $\boldsymbol{q}_g$
- Issues: Parabolic grows too fast, leading to large initial attractive forces
- Conical potential undefined at $\boldsymbol{q}_g$, causing stability problems nearby

- Solution: Use parabolic near goal, use conical far away, and connect them smoothly

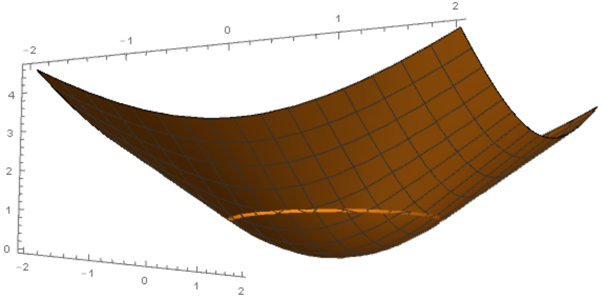
$$U_a(\boldsymbol{q}) = \begin{cases} \frac{1}{2}k_a\|\boldsymbol{e}(\boldsymbol{q})\|^2, & \|\boldsymbol{e}(\boldsymbol{q})\| \leq d \\ dk_a\|\boldsymbol{e}(\boldsymbol{q})\| - \frac{1}{2}k_a d^2, & \|\boldsymbol{e}(\boldsymbol{q})\| \geq d \end{cases} \qquad f_a(\boldsymbol{q}) = \begin{cases} k_a\boldsymbol{e}(\boldsymbol{q}), & \|\boldsymbol{e}(\boldsymbol{q})\| \leq d \\ dk_a\frac{\boldsymbol{e}(\boldsymbol{q})}{\|\boldsymbol{e}(\boldsymbol{q})\|}, & \|\boldsymbol{e}(\boldsymbol{q})\| \geq d \end{cases}$$

- At $\|\boldsymbol{e}(\boldsymbol{q})\| = d$, $U_a(\boldsymbol{q}) = \frac{1}{2}k_a d^2$ and $f_a(\boldsymbol{q}) = k_a d$

Combined Potential

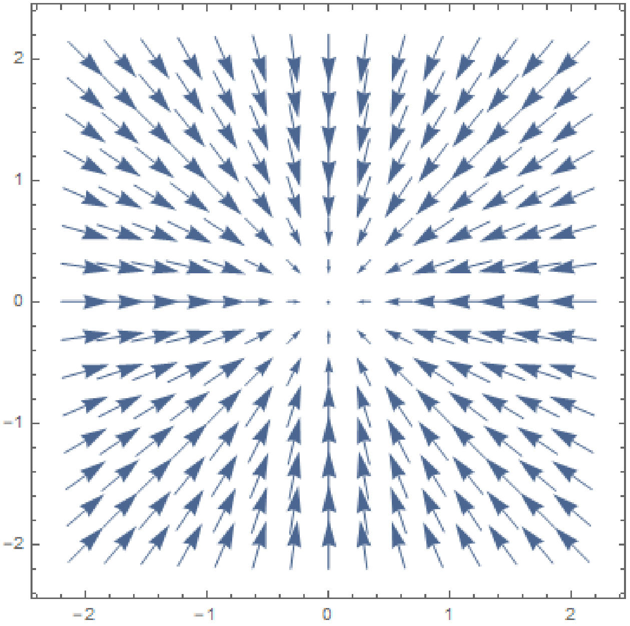


$U_a(\boldsymbol{q})$
(top
view)



$U_a(\boldsymbol{q})$
(bottom
view)

$f_a(\boldsymbol{q})$

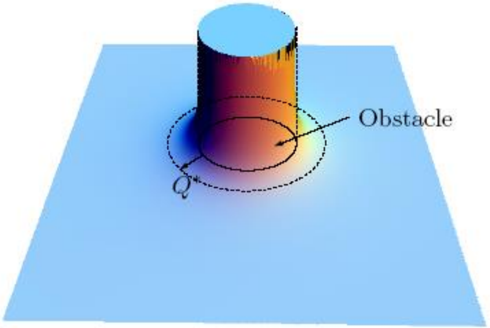


Repulsive Potential

- Requirements: Robot should never collide with obstacle, but obstacle should have little to no influence on robot when they are far away from each other
- Solution: Infinite potential at obstacle boundary, zero some distance away

- Define $\eta_i(\boldsymbol{q})$ to be distance from $\boldsymbol{q}$ to closest point on obstacle i
- Define $\eta_{0,i}$ to be *range of influence* of obstacle i

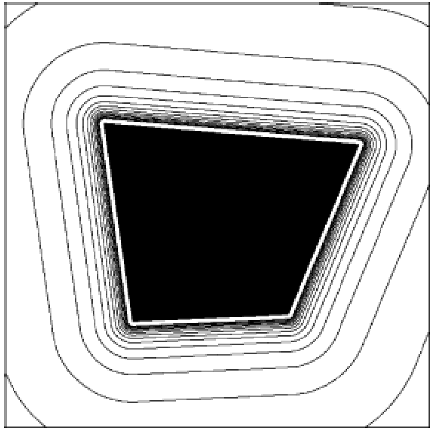
$$U_{r,i}(\boldsymbol{q}) = \begin{cases} \frac{1}{2} k_{r,i} \left(\frac{1}{\eta_i(\boldsymbol{q})} - \frac{1}{\eta_{0,i}} \right)^2, & \eta_i(\boldsymbol{q}) \leq \eta_{0,i} \\ 0, & \eta_i(\boldsymbol{q}) > \eta_{0,i} \end{cases}$$



Repulsive Potential

- Resulting *equipotential contours* wrap around the obstacles, parallel to edges and curved around corners
- Repulsive force:

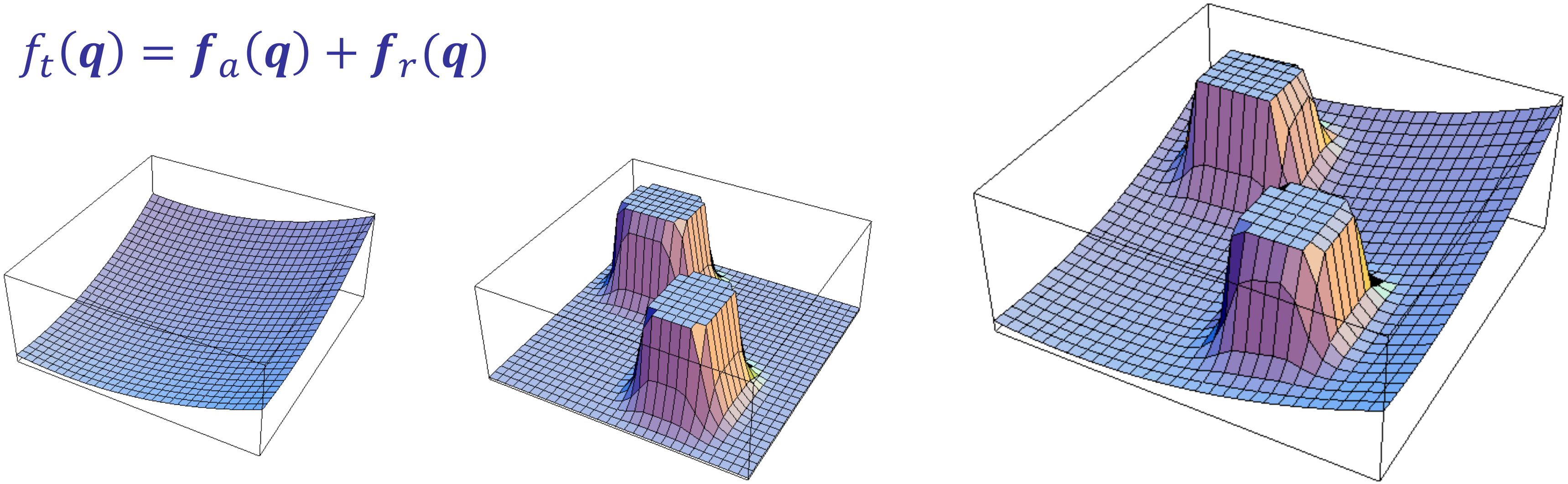
$$f_{r,i}(\boldsymbol{q}) = \begin{cases} \dfrac{k_{r,i}}{\eta_i^2(\boldsymbol{q})}\left(\dfrac{1}{\eta_i(\boldsymbol{q})} - \dfrac{1}{\eta_{0,i}}\right)\nabla\eta_i(\boldsymbol{q}), & \eta_i(\boldsymbol{q}) \leq \eta_{0,i} \\ 0, & \eta_i(\boldsymbol{q}) > \eta_{0,i} \end{cases}$$



- $\nabla\eta_i(\boldsymbol{q})$ is gradient vector between obstacle and $\boldsymbol{q}$
- Total repulsive potential: $U_r(\boldsymbol{q}) = \sum_i U_{r,i}(\boldsymbol{q})$

Total Potential

- $U_t(\boldsymbol{q}) = U_a(\boldsymbol{q}) + U_r(\boldsymbol{q})$
- $f_t(\boldsymbol{q}) = \boldsymbol{f}_a(\boldsymbol{q}) + \boldsymbol{f}_r(\boldsymbol{q})$



$U_a(\boldsymbol{q})$ $U_r(\boldsymbol{q})$ $U_t(\boldsymbol{q})$

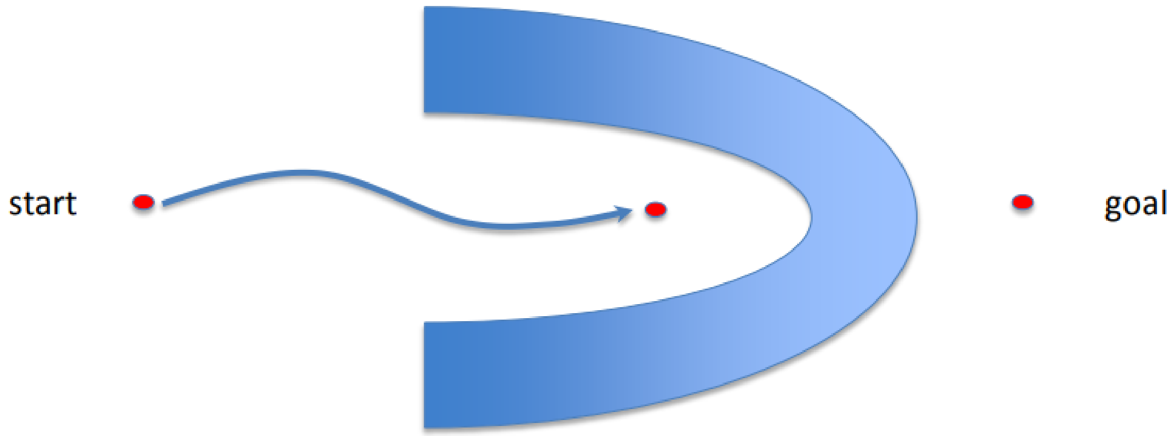
Motion Control

- How do potential and force field move the robot?
- One method: Treat $\boldsymbol{f}_t(\boldsymbol{q})$ as a local gradient in gradient descent algorithm
- Iteratively change configuration $\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \alpha \boldsymbol{f}_t(\boldsymbol{q}_k)$
  - Step size α may also change from iteration to iteration

- Parameters to tweak
  - $k_{r,i}$, relative influence of obstacles—usually want values near the goal to be smaller to avoid pushing robot away
  - $\eta_{0,i}$, range of influence—don't want goal to be in any obstacle's range, may want to tweak so that different obstacles don't overlap

Local Minima

- Local minima occur wherever attractive force cancel out repulsive forces
- More likely with large number of attractive and repulsive fields
- Robot will stop moving! How to get out or avoid in the first place?

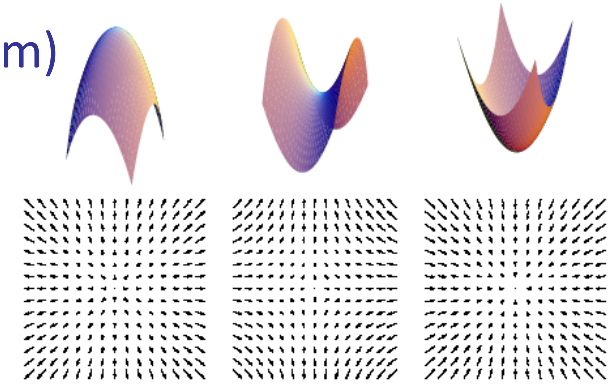


Local Minima

- Stationary points exist wherever $\nabla U = 0$ and robot stops moving

- How to tell the type of stationary point?
- Look at the Hessian $\boldsymbol{H}$—(matrix) derivative of the gradient vector

- $\boldsymbol{H}$ is negative-definite: local maximum (not a problem)
- $\boldsymbol{H}$ is positive-definite: local minimum
- $\boldsymbol{H}$ is indefinite: saddle point (unstable)



(Maximum) (Saddle) (Minimum)

Random Walks

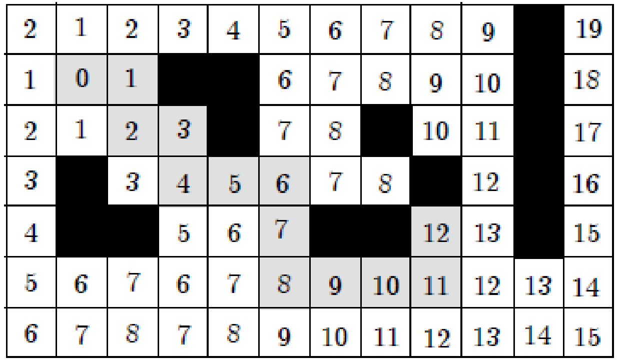
- Idea: If we get stuck at or near a local minimum, perturb the robot to get out
- Detection: May need some thresholding for successive updates to be within a certain range of each other (but still not near the goal)
- Implementation: Simulate Brownian motion by sampling random steps from a zero-mean Gaussian and add to current configuration

Grid Discretization

- If dimension is low, we can use potential function to approximate a grid in the C-space with cell values, followed by search using wave front / brushfire

- Each cell assigned potential value of centroid
- Move toward neighboring cell with lowest value

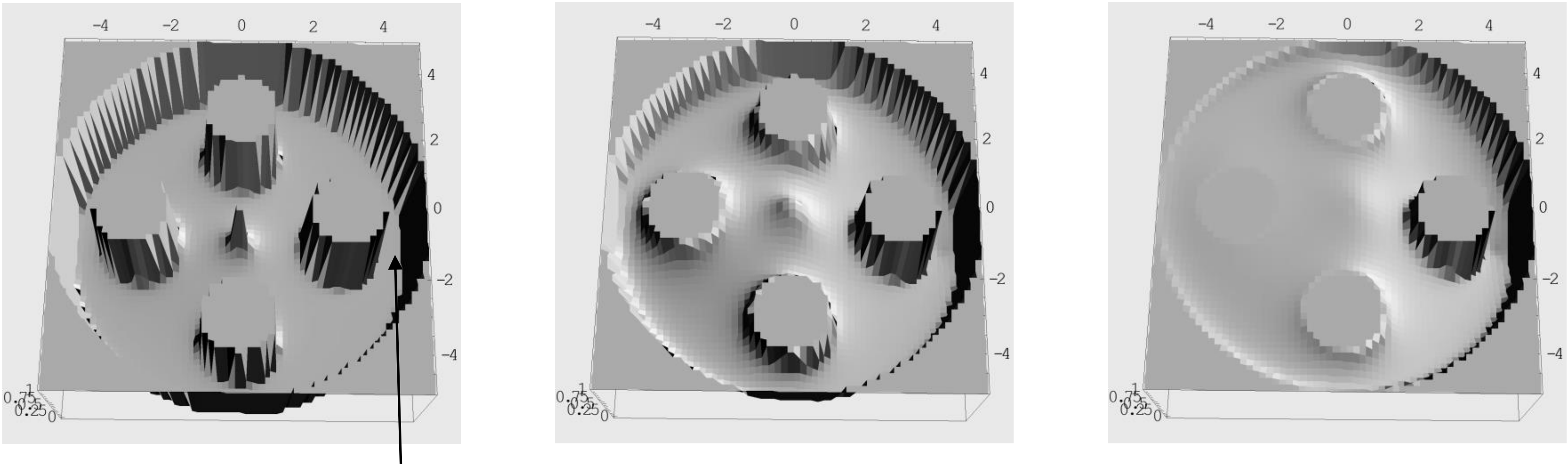
- If local minimum reached, expand uniformly outward
- Continue search when new decreasing path found



- Procedure is resolution-complete; declare failure if all cells explored

Navigation Functions

- *Navigation functions:* special case of artificial potentials with no local minima
- E.g., true if all obstacles are spheres, if k_a and $k_{r,i}$ are large enough



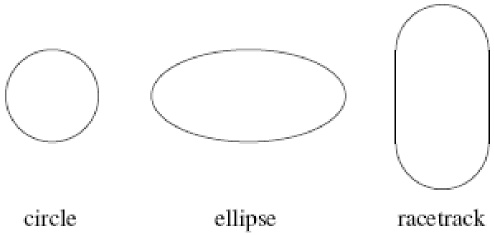
Goal

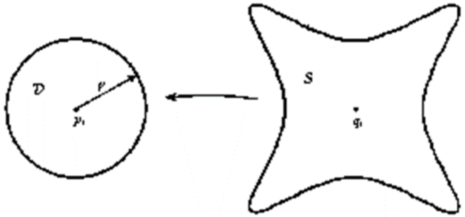
k increases →

Diffeomorphisms to Spheres

- *Diffeomorphism*: A bijective (invertible), smooth (continuous and all partial derivatives exist) mapping whose inverse is also smooth

- If a diffeomorphism exists between a set of C-obstacles and a set of spheres, then we can find a local mimima-free potential in the sphere world and transform back to original environment





- Ellipses and “racetracks” are diffeomorphic to circles
- So are stars—sets in which all boundaries can be seen from any point within the set

Summary

- Potential fields are a gradient-based approach to path planning

- Attractive (parabolic, conic) potentials to points of interest
- Repulsive potentials from obstacles

- Local minima are a problem; can try to randomly perturb out of them
- Navigation functions—avoid having local minima in the first place