CS 689: Robot Motion Planning
Potential Functions, aka *May the Force be with you*

Amarda Shehu

Department of Computer Science
George Mason University
Basic Idea

- Suppose the goal is a point $g \in \mathbb{R}^2$
- Suppose the robot is a point $r \in \mathbb{R}^2$
- Think of a spring drawing the robot toward the goal and away from obstacles
- Can also think of like and opposite charges
Another Idea

- Think of the goal as the bottom of a bowl
- The robot is at the rim of the bowl
- What will happen?
Both the spring and bowl analogies are ways of storing potential energy.
The robot moves to a lower-energy configuration.

A potential function is a function $U : \mathbb{R}^n \rightarrow \mathbb{R}$.

Energy is minimized by following the negated gradient of the potential energy function:

$$\nabla U(q) = \left[ \frac{\partial U}{\partial q_1}(q), \ldots, \frac{\partial U}{\partial q_n}(q) \right]^T$$

We can now think of a vector field over the space of all $q$'s:

- The robot looks at the vector at its current position and goes in that direction.
Attractive + Repulsive Potentials

Desired objectives

- robot moves toward the goal (attractive potential)
- robot stays away from the obstacles (repulsive potential)

\[ U(q) = U_{\text{att}}(q) + U_{\text{rep}}(q) \]
Attractive potential: $U_{att}(q)$

- monotonically increasing with distance from $q_{goal}$
- example: conic potential (scaled distance to goal, $\zeta > 0$ scaling factor)

$$U_{att}(q) = \zeta ||q, q_{goal}||$$

- what’s the gradient?

$$\nabla U_{att}(q) = \frac{\zeta}{||q, q_{goal}||} (q - q_{goal})$$

- what’s the magnitude of the gradient at $q$?

$$||\nabla U_{att}(q)|| = \begin{cases} \zeta, & q \neq q_{goal} \\ undefined, & q = q_{goal} \end{cases}$$

- conic potential has discontinuity at $q_{goal}$
Attractive potential: $U_{\text{att}}(q)$

- monotonically increasing with distance from $q_{\text{goal}}$
- preference:
  - continuously differentiable + magnitude decreases as robot approaches $q_{\text{goal}}$
- example: quadratic potential ($\zeta > 0$ scaling factor)
  \[
  U_{\text{att}}(q) = \frac{1}{2} \zeta ||q - q_{\text{goal}}||^2
  \]

- what’s the gradient?
  \[
  \nabla U_{\text{att}}(q) = \zeta (q - q_{\text{goal}})
  \]

- what’s the magnitude of the gradient at $q$?
  \[
  ||\nabla U_{\text{att}}(q)|| = \zeta ||q, q_{\text{goal}}||
  \]

Figure: (a) Potential Field. (b) Contour Plot. (c) Quadratic Potential.
Attractive Potential: Quadratic Potential

Attractive potential: $U_{\text{att}}(q)$

- monotonically increasing with distance from $q_{\text{goal}}$
- preference:
  - continuously differentiable + magnitude decreases as robot approaches $q_{\text{goal}}$
- example: quadratic potential ($\zeta > 0$ scaling factor)

\[
U_{\text{att}}(q) = \frac{1}{2} \zeta ||q, q_{\text{goal}}||^2
\]

- what’s the gradient?

\[
\nabla U_{\text{att}}(q) = \zeta (q - q_{\text{goal}})
\]

- what’s the magnitude of the gradient at $q$?

\[
||\nabla U_{\text{att}}(q)|| = \zeta ||q, q_{\text{goal}}||
\]

- what happens when robot is far away from the goal?
- robot may move too fast as potential grows without bounds the further away from goal; this may produce a velocity that is too large
Attractive potential: \( U_{\text{att}}(q) \)

- monotonically increasing with distance from \( q_{\text{goal}} \)
- preference:
  - continuously differentiable, magnitude decreases as robot approaches \( q_{\text{goal}} \)
  - does not produce very large velocities
- combine conic and quadratic potentials (\( \zeta > 0 \) scaling factor)

\[
U_{\text{att}}(q) = \begin{cases} 
\frac{1}{2} \zeta \| q, q_{\text{goal}} \|^2, & \text{if } \| q, q_{\text{goal}} \| \leq d^*_\text{goal} \\
 d^*_\text{goal} \zeta \| q, q_{\text{goal}} \| - \frac{1}{2} \zeta (d^*_\text{goal})^2, & \text{if } \| q, q_{\text{goal}} \| > d^*_\text{goal}
\end{cases}
\]

(\( d^*_\text{goal} \): threshold from goal where planner switches between conic and quadratic potentials)

- what’s the gradient? is it well defined at the boundary?

\[
\nabla U_{\text{att}}(q) = \begin{cases} 
\zeta (q - q_{\text{goal}}), & \text{if } \| q, q_{\text{goal}} \| \leq d^*_\text{goal} \\
 d^*_\text{goal} \zeta (q - q_{\text{goal}})/\| q, q_{\text{goal}} \|, & \text{if } \| q, q_{\text{goal}} \| > d^*_\text{goal}
\end{cases}
\]
Repulsive potential: $U_{rep}(q)$

- the closer the robot is to an obstacle, the stronger the repulsive force should be
- robot keeps track of closest obstacle
- there is a threshold so robot can ignore far away obstacles
Repulsive potential: $U_{rep}(q)$

- The closer the robot is to an obstacle, the stronger the repulsive force should be.

$$U_{rep}(q) = \begin{cases} 
\frac{1}{2} \eta \left( \frac{1}{D(q)} - \frac{1}{d_{obst}^*} \right)^2, & \text{if } D(q) \leq d_{obst}^* \\
0, & \text{otherwise}
\end{cases}$$

$$\nabla U_{rep}(q) = \begin{cases} 
\eta \left( \frac{1}{d_{obst}^*} - \frac{1}{D(q)} \right) \frac{1}{(D(q))^2} \nabla D(q), & \text{if } D(q) \leq d_{obst}^* \\
0, & \text{otherwise}
\end{cases}$$

- $D(q)$: Distance to the closest obstacle; $\eta > 0$ scaling factor
- $d_{obst}^*$: Threshold to allow the robot to ignore obstacles far away from it
Repulsive Potential

Repulsive potential: $U_{rep}(q)$

- the closer the robot is to an obstacle, the stronger the repulsive force should be

$$U_{rep}(q) = \begin{cases} \frac{1}{2} \eta \left( \frac{1}{D(q)} - \frac{1}{d^*_\text{obst}} \right)^2, & \text{if } D(q) \leq d^*_\text{obst} \\ 0, & \text{otherwise} \end{cases}$$

$$\nabla U_{rep}(q) = \begin{cases} \eta \left( \frac{1}{d^*_\text{obst}} - \frac{1}{D(q)} \right) \frac{1}{(D(q))^2} \nabla D(q), & \text{if } D(q) \leq d^*_\text{obst} \\ 0, & \text{otherwise} \end{cases}$$

- $D(q)$: distance to the closest obstacle; $\eta > 0$ scaling factor
- $d^*_\text{obst}$: threshold to allow the robot to ignore obstacles far away from it
- what happens around points that are two-way equidistant from obstacles?

$D$ is nonsmooth $\Rightarrow$ path may oscillate
Repulsive Potential

Repulsive potential: $U_{rep}(q)$

- minimum distance to $i$-th obstacle
  
  $$d_i(q) = \min_{c \in \text{Obstacle}_i} d(q, c)$$

- for convex obstacles ($c$ is closest point to $q$)
  
  $$\nabla d_i(q) = \frac{c - q}{\|q, c\|}$$

- repulsive potential for each obstacle
  
  $$U_{rep_i}(q) = \begin{cases} 
  \frac{1}{2} \eta \left( \frac{1}{d_i(q)} - \frac{1}{d^{*}_{\text{obst}_i}} \right)^2, & \text{if } d_i(q) \leq d^{*}_{\text{obst}_i} \\
  0, & \text{otherwise} 
  \end{cases}$$

- overall repulsive potential as sum of obstacle potentials
  
  $$U_{rep}(q) = \sum_i U_{rep_i}(q)$$
Gradient Descent: Moving Opposite to the Gradient

repeat until gradient is zero (or its magnitude very small)
  ■ take small step in the direction opposite the gradient

Pseudocode

1: \( q \leftarrow q_{init} \)
2: while \( ||\nabla U(q)|| > \epsilon \) do
3: \( q \leftarrow q - \alpha \nabla U(q) \)

■ \( \epsilon > 0 \): small constant to ensure termination criteria
■ \( \alpha > 0 \): step size (doesn’t have to be constant)

Figure: (a): Configuration space with gray obstacles. (b) Potential function energy surface. (c) Contour plot for energy surface. (d) Gradient vectors for potential function.
Gradient Descent: Moving Opposite to the Gradient

repeat until gradient is zero (or its magnitude very small)
- take small step in the direction opposite the gradient

Pseudocode
1: $q \leftarrow q_{init}$
2: while $||\nabla U(q)|| > \epsilon$ do
3: $q \leftarrow q - \alpha \nabla U(q)$

- $\epsilon > 0$: small constant to ensure termination criteria
- $\alpha > 0$: step size (doesn’t have to be constant)

Weaknesses of Gradient Descent
- it is relatively slow close to the minimum
- it might ‘zigzag’ down valleys

Better Methods
- Broyden-Fletcher-Goldfarb-Shanno (BFGS) method
  - … but more complex to implement
Mobile Robot Implementation

- Robot knows goal position
- Robot does not know where obstacles are located
- Robot has range sensor and can determine its own position

$U_{att}(q)$ can be easily computed since goal position is known

$U_{rep}(q)$ approximate it via data from range sensor
  - $D(q)$: approximated as the global minimum of the raw distance function $\rho$
  - $d_i(q)$: approximated as local minima with respect to $\theta$ in $\rho(q, \theta)$
**Brushfire Algorithm – Compute Distances on a Grid**

$U_{rep}(q)$:

- Discretize space by imposing a grid (define cell neighbors 4- or 8-connectivity)
- Label with 1 cells that are partially or fully occupied by obstacles
- Label with 2 all unlabeled cells neighboring 1-labeled cells
- ...
- Label with $n$ all unlabeled cells neighboring $(n - 1)$-labeled cells
- Stop when all cells have been labeled

- Gradient from each cell points to a neighbor with lowest label
Brushfire Algorithm – Compute Distances on a Grid

\[ U_{\text{rep}}(q): \]
- discretize space by imposing a grid (define cell neighbors 4- or 8-connectivity)
- label with 1 cells that are partially or fully occupied by obstacles
- label with 2 all unlabeled cells neighboring 1-labeled cells
- \ldots
- label with \( n \) all unlabeled cells neighboring \((n - 1)\)-labeled cells
- stop when all cells have been labeled
- can planner get stuck?
Gradient descent algorithms may get stuck in local minima

Two approaches to avoid local-minima problem

- do something different than gradient descent to overcome/avoid local minima
- define potential function so that there is only one global minimum
Wave-Front Planner: Complete Planner in Grid Spaces

- similar to Brushfire algorithm discretize space by imposing a grid
- label with 1 cells that are partially or fully occupied by obstacles
- label with 2 cell where goal is located
- label with 3 all unlabeled cells neighboring 2-labeled cells
- ... 
- label with $n$ all unlabeled cells neighboring $(n-1)$-labeled cells
- stop when init cell (green circle) has been labeled

![Wave-Front Planner Grid Diagram](image)

- each time move to neighboring non-obstacle cell with lowest label
How can we deal with rigid bodies and manipulators?

- Think of gradient vectors as forces
- Define forces in workspace $W$ (which is $\mathbb{R}^2$ or $\mathbb{R}^3$)
- “Lift up” forces in configuration space $Q$

Relationship between Forces in the Workspace and Configuration Space

- point $x \in W$ in workspace related to configuration $q \in Q$ via forward kinematics

$$x = FK(q)$$

- “virtual work” principle: work (or power) is a coordinate-independent quantity
  - in workspace, power done by a force $f$ is $f^T \dot{x}$
  - in configuration space, power done by a force $u$ is $u^T \dot{q}$
- mapping from workspace forces to configuration space forces done via Jacobian $J = \partial FK/\partial q$ of the forward kinematic function

$$f^T \dot{x} = u^T \dot{q} \quad \text{(by the “virtual work” principle)}$$

$$\Rightarrow f^T J \dot{q} = u^T \dot{q} \quad \text{(by Jacobian property $\dot{x} = J \dot{q}$)}$$

$$\Rightarrow f^T J = u^T$$

$$\Rightarrow J^T f = u$$
Pick control points $r_1, \ldots, r_n$ on the robot in its initial placement, e.g., $r_j$ could be selected as the $j$-th robot vertex.

Let $FK_j(q)$ denote the forward kinematics of point $r_j$

Example: when $q = (x, y, \theta)$ and $r_j = (x_j, y_j)$

$$FK_j(q) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x_j \\ y_j \end{pmatrix} + \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_j \cos \theta - y_j \sin \theta + x \\ x_j \sin \theta + y_j \cos \theta + y \end{pmatrix}$$

Define $\nabla U_{attj}$ in workspace for each control point $r_j$, and scale it appropriately, e.g.,

$$\nabla U_{attj}(q) = \text{SCALE}_{att} \left( FK_j(q) - \begin{pmatrix} g_x \\ g_y \end{pmatrix} \right), \quad \text{where } (g_x, g_y) \text{ is goal center}$$

Define $\nabla U_{rep_{i,j}}$ in workspace for each control point $r_j$ and obstacle $i$, and scale it appropriately,

$$\nabla U_{rep_{i,j}}(q) = \text{SCALE}_{rep} \left( \begin{pmatrix} o_{i,x} \\ o_{i,y} \end{pmatrix} - FK_j(q) \right),$$

where $(o_{i,x}, o_{i,y})$ is closest point to $FK_j(q)$ on obstacle $i$.
Potential Functions for Rigid-Body Robots (cont.)

- Compute Jacobian

\[
J_j(q) = \begin{pmatrix}
\frac{\partial FK_j(q)[1]}{\partial x} & \frac{\partial FK_j(q)[1]}{\partial y} & \frac{\partial FK_j(q)[1]}{\partial \theta} \\
\frac{\partial FK_j(q)[2]}{\partial x} & \frac{\partial FK_j(q)[2]}{\partial y} & \frac{\partial FK_j(q)[2]}{\partial \theta}
\end{pmatrix}
\]

- Compute overall gradient in configuration space (apply Jacobian to scaled versions of the workspace gradients)

\[
\nabla U_{cs}(q) = \sum_j J_j^T(q) \nabla U_{att_j}(q) + \sum_j J_j^T(q) \sum_i \nabla U_{rep_{i,j}}(q)
\]

Apply appropriate scaling to position and orientation components separately, i.e.,

\[
\text{move}_{x,y} \leftarrow \text{SCALE}_{cs}(\nabla U_{cs_{x,y}}(q)), \quad \text{move}_\theta \leftarrow \text{SCALE}_{cs}(\nabla U_{cs_\theta}(q))
\]
End position of the $j$-th link ($1 \leq j \leq n$):

$$FK_j(\theta_1, \theta_2, \ldots, \theta_n) = M(\theta_1)M(\theta_2) \ldots M(\theta_j) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \text{where for } 1 \leq i \leq j$$

$$M(\theta_i) = \begin{pmatrix} \cos \theta_i & -\sin \theta_i & 0 \\ \sin \theta_i & \cos \theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \ell_i \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos \theta_i & -\sin \theta_i & \ell_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i & \ell_i \sin \theta_i \\ 0 & 0 & 1 \end{pmatrix}$$

Jacobian of $j$-th link ($1 \leq j \leq n$):

$$J_j(\theta_1, \ldots, \theta_n) = \begin{pmatrix} \frac{\partial FK_j(\theta_1, \ldots, \theta_n)[1]}{\partial \theta_1} & \ldots & \frac{\partial FK_j(\theta_1, \ldots, \theta_n)[1]}{\partial \theta_j} \\ \ldots & \ldots & \ldots \\ \frac{\partial FK_j(\theta_1, \ldots, \theta_n)[2]}{\partial \theta_1} & \ldots & \frac{\partial FK_j(\theta_1, \ldots, \theta_n)[2]}{\partial \theta_j} \end{pmatrix} \begin{pmatrix} 0 & \ldots & 0 \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 \end{pmatrix}, \text{where for } 1 \leq i \leq j$$

$$\frac{\partial FK_j(\theta_1, \ldots, \theta_n)[1]}{\partial \theta_i} = -\sin \theta_i (ga + hb + a\ell_i) + \cos \theta_i (gb - ha + b\ell_i)$$

$$\frac{\partial FK_j(\theta_1, \ldots, \theta_n)[2]}{\partial \theta_i} = -\sin \theta_i (gd + he + d\ell_i) + \cos \theta_i (ge - hd + e\ell_i)$$

$$\begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} = M(\theta_1) \ldots M(\theta_{i-1}), \begin{pmatrix} g \\ h \\ 1 \end{pmatrix} = M(\theta_{i+1}) \ldots M(\theta_j) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$
2d chain with n revolute joints where link j has length \( \ell_j \)

- Compute \( J_j(\theta_1, \ldots, \theta_n), \ i \leq j \), using a simplified but equivalent definition

\[
\frac{\partial \text{FK}_j}{\partial \theta_i} = \left( \begin{array}{c} -\text{FK}_j(\theta_1, \ldots, \theta_n)[2] + \text{FK}_{i-1}(\theta_1, \ldots, \theta_n)[2] \\ \text{FK}_j(\theta_1, \ldots, \theta_n)[1] - \text{FK}_{i-1}(\theta_1, \ldots, \theta_n)[1] \end{array} \right)
\]

- Define \( \nabla U_{\text{att}_n} \) for the end-effector and scale it appropriately:

\[
\nabla U_{\text{att}_n}(\theta_1, \ldots, \theta_n) = \text{SCALE}_{\text{att}} \left( \text{FK}_n(\theta_1, \ldots, \theta_n) - \left( \begin{array}{c} g_x \\ g_y \end{array} \right) \right), \quad (g_x, g_y): \text{goal center}
\]

- Define \( \nabla U_{\text{rep}_{i,j}} \) in workspace between the end-position of the j-th link and the i-th obstacle and scale it appropriately, e.g.,

\[
\nabla U_{\text{rep}_{i,j}}(\theta_1, \ldots, \theta_n) = \text{SCALE}_{\text{rep}} \left( \left( \begin{array}{c} o_{i,x} \\ o_{i,y} \end{array} \right) - \text{FK}_j(\theta_1, \ldots, \theta_n) \right),
\]

\((o_{i,x}, o_{i,y}): \text{closest point on the i-th obstacle to the end position of the j-th link}\)

- Compute overall gradient in configuration space

\[
\nabla U_{\text{cs}}(\theta_1, \ldots, \theta_n) = \text{SCALE} (J_j^T(\theta_1, \ldots, \theta_n) \nabla U_{\text{att}_n}(\theta_1, \ldots, \theta_n) + \sum_{i,j} J_j^T(\theta_1, \ldots, \theta_n) \nabla U_{\text{rep}_{i,j}}(\theta_1, \ldots, \theta_n))
\]
Summary

Basic potential fields: attractive/repulsive forces

Path planning by following gradient of potential field
- Gradient descent (incomplete, suffers from local minima)
- Brushfire algorithm (incomplete, suffers from local minima, grid world)
- Wavefront planner (complete, grid world)

Potential Functions in Non-Euclidean Spaces
- Gradients as forces
- Lift up workspace forces to configuration space forces
- Applicable to rigid body robots and manipulators