CSCE 452/752 Fall 2024

8. Navigation: Potential fields

TEXAS A&M UNIVERSITY

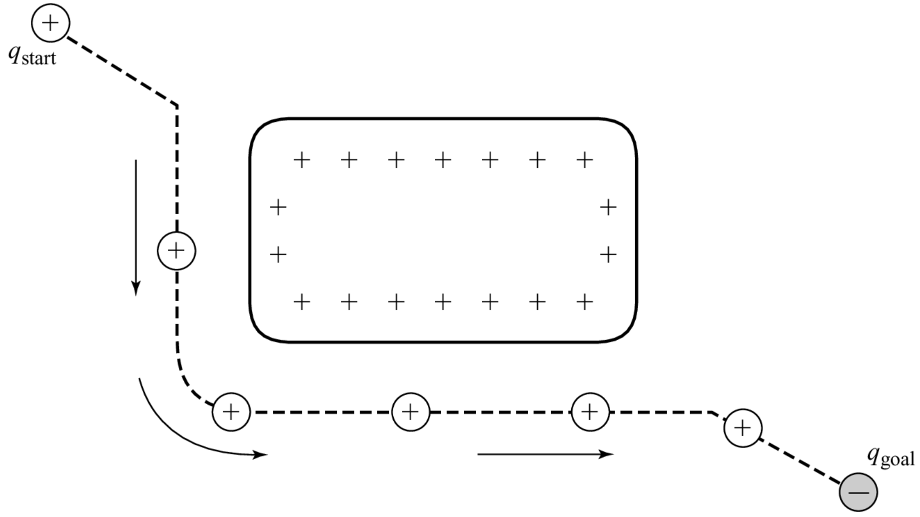
AARRG

Introduction

Definition

Potential fields model the robot as a particle responding to an attractive force from its goal and repulsive forces from obstacles.

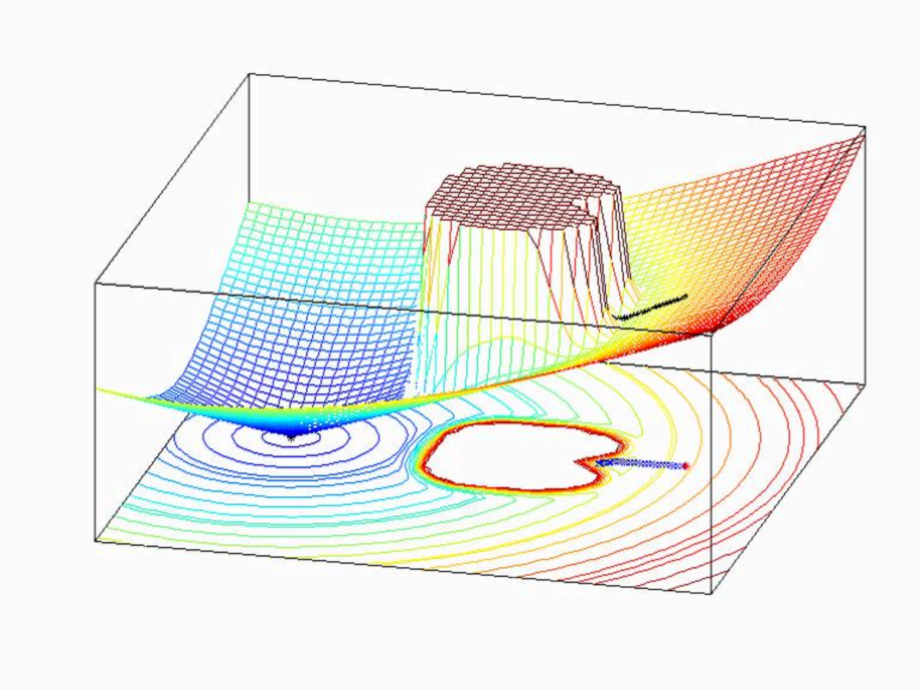
The intuition is that a **potential function** defines a landscape along which the robot moves “downhill”.



Potential functions

A **potential function** is a function $U : X \to \mathbb{R}$.

- Input: A state x.
- Output: The potential $U(x)$ at state x.



Moving in the potential field

The robot's motion follows the negative gradient of the potential function.

$$-\nabla U(x, y) = -\begin{pmatrix} \partial U / \partial x \\ \partial U / \partial y \end{pmatrix}$$

Intuition: Always move in the steepest direction downhill.

This forms a **vector field**, in which each state is associated with a vector showing the direction the robot should move from that state.

Additive potential functions

The most common technique for constructing potential functions is to use an **additive model**:

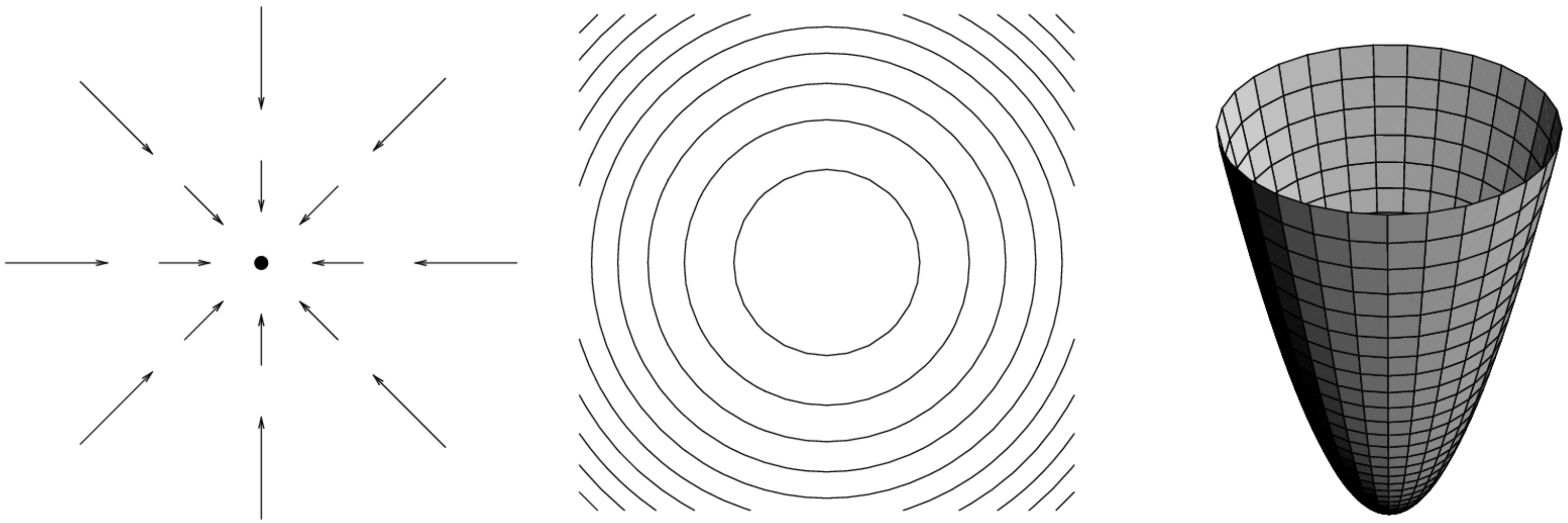
$$U(x) = U_{\text{goal}}(x) + \sum_i U^{(i)}_{\text{obst}}(x)$$

- U_{goal} generates an attractive force toward the goal.
- U_{obst} generates repulsive forces away from each obstacle.

Attractive potential

A simple choice for the attractive potential is based on a scaled squared distance to the goal:

$$U_{\text{goal}}(x) = \frac{1}{2}\zeta\left[d(x, x_{\text{goal}})\right]^2$$



(a) (b) (c)

Repulsive potential

We also need to define a repulsive potential for each obstacle. A common choice is

$$U_{\text{obst}}^{(i)}(x) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d_i(x)} - \frac{1}{Q_i^*}\right)^2 & \text{if } d_i(x) \leq Q_i^* \\ 0 & \text{if } d_i(x) > Q_i^* \end{cases}$$

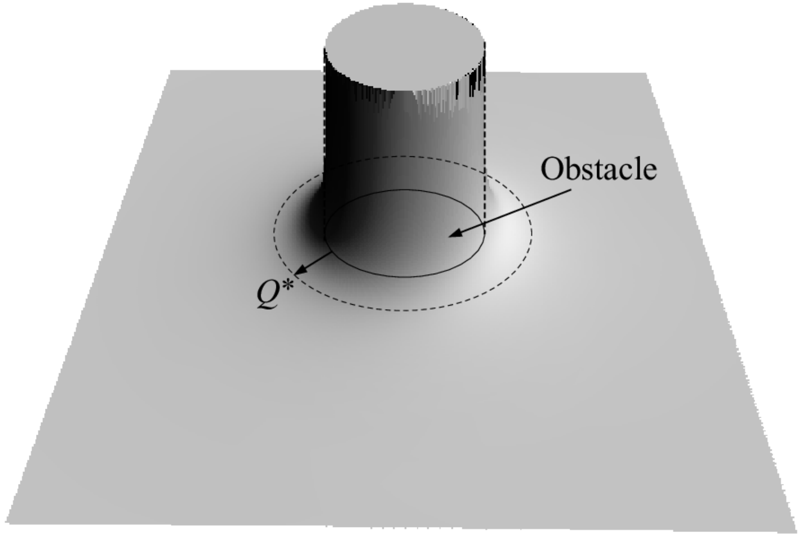
In this setup:

- $d_i(x)$ is the distance from state x to the closest point on obstacle i.
- Q_i^* is the maximum “range of influence” for obstacle i.
- η is a scaling factor.

Repulsive potential

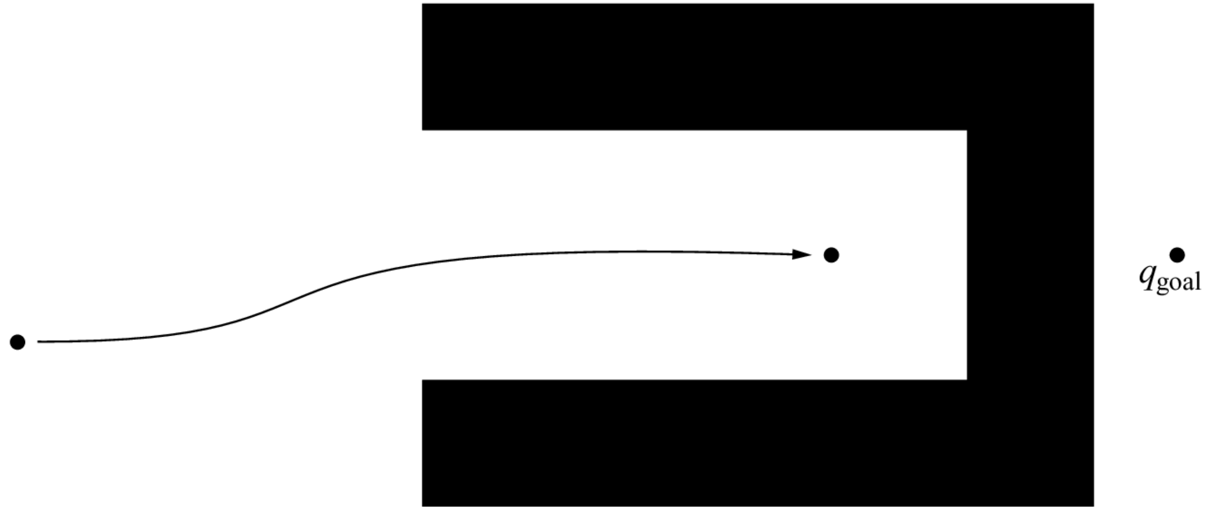
We also need to define a repulsive potential for each obstacle. A common choice is

$$U_{\text{obst}}^{(i)}(x) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d_i(x)} - \frac{1}{Q_i^*}\right)^2 & \text{if } d_i(x) \le Q_i^* \\ 0 & \text{if } d_i(x) > Q_i^* \end{cases}$$



What could go wrong?

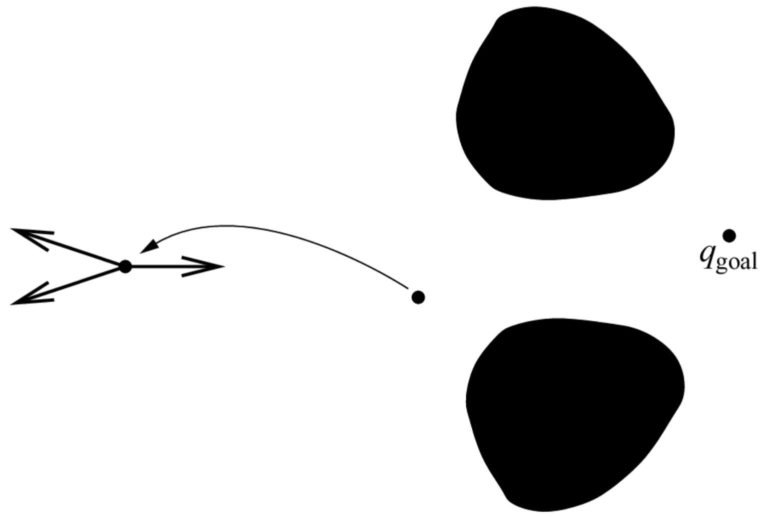
The main limitation of the potential field method is the problem of **local minima**: points other than the goal (the “global minimum”) at which the gradient is zero.



The robot gets “stuck”. There is **no guarantee** that the robot will reach its goal.

What could go wrong?

The main limitation of the potential field method is the problem of **local minima**: points other than the goal (the “global minimum”) at which the gradient is zero.



The robot gets “stuck”. There is **no guarantee** that the robot will reach its goal.

Avoiding local minima

How can a robot avoid becoming stuck in a local minimum of its potential field?

- Find a better potential function that doesn't have local minima. (A **navigation function**.)
- **Notice** that we've reached a local minimum, and try to escape. (Short-term gradient ascent; random motions; etc)

Avoiding local minima

How can a robot avoid becoming stuck in a local minimum of its potential field?

- Find a better potential function that doesn't have local minima. (A **navigation function**.)
- **Notice** that we've reached a local minimum, and try to escape. (Short-term gradient ascent; random motions; etc)

Or... choose a different navigation algorithm altogether.

Summary

We saw three classes of navigation algorithms.

- Visibility graphs
- Bug algorithms
- Potential fields

When is it possible to use each one? When is it a good idea?