Motion and path planning in a nutshell

Prof. Hadas Kress-Gazit
MAE
Guest lecture: CS 4758/6758
March 15, 2012
easy

not as easy
“How do I get to point B?”

• Motion planning
  – Bug algorithms
  – Roadmaps, cell decomposition
  – Potential functions
  – Sampling-based methods
“How do I get to point B?”

• Motion planning
  – Bug algorithms
  – Roadmaps, cell decomposition
  – Potential functions (“vanilla” potential functions)
  – Sampling-based methods (RRT)
Potential functions – basic idea

Energy function over $C_{\text{free}}$ (obstacle-free configuration space)

Ideally global minimum at goal

$\nabla$
Definitions

• Potential function
  \[ U : \mathbb{R}^n \rightarrow \mathbb{R} \]

• Gradient
  \[ \nabla U(q) = \begin{bmatrix} \frac{\partial U}{\partial q_1}(q) \\ \vdots \\ \frac{\partial U}{\partial q_n}(q) \end{bmatrix} \]

• Control
  \[ \dot{q} = -\nabla U(q) \]
Attractive force = go to goal

\[ u(q, q_{\text{goal}}) = \| q - q_{\text{goal}} \| \]

\[ U_{\text{attr}} = C \cdot u(q, q_{\text{goal}}) \]

\[ \nabla U_{\text{attr}} = \frac{C}{u(q, q_{\text{goal}})} \cdot (q - q_{\text{goal}}) \]

\[ U_{\text{attr}} = \frac{1}{2} C \cdot u(q, q_{\text{goal}})^2 \]

\[ \nabla U_{\text{attr}} = C \cdot (q - q_{\text{goal}}) \]
Repulsive force = keep away from obstacles

Distance from obstacle

$$d_i(q) = \min_{q^* \in \text{obs}} d(q, q^*)$$

$$U_{rep} = \begin{cases} \frac{1}{2} < \left( \frac{1}{d_i(q)} - \frac{1}{Q} \right)^2 \\ 0 \end{cases}$$

$$d_i(q) \leq Q$$

$$d_i(q) > Q$$

$$Q > 0$$
Potential function

\[ U(q) = U_{at} + \sum_{obs} U_{rep_i}(q) \]

\( \bar{r}_i \)

cannot complete (every initial point will reach the goal *)
Problem

Local minima!

Solutions:
- Navigation functions
- Potential functions
  over cell decomposition
Problem (2)

Complex environment
Different approach - samples

- Probabilistically\resolution complete
- Good for complex configuration spaces
Single queries

- Find a path from $q_{\text{init}}$ to $q_{\text{goal}}$
- Idea:
  - grow tree(s) spanning “relevant” space
  - Connect tree(s)
Rapidly-Exploring Random Trees (RRT)
RRTs

Algorithm:

Given: \( q_{\text{start}}, q_{\text{end}}, \text{step-size}, n = \# \text{ of attempts to grow the tree} \)

Find: \( G = (V, E) \) \( v \in \mathbb{R}^n \) \( e \in \mathbb{R}^n \times \mathbb{R}^n \)

Init: \( V = \{ q_{\text{start}} \} \) \( E = \emptyset \)

For \( i = 1 \) to \( n \):

- sample \( q_{\text{rand}} \) \& \( C_{\text{free}} \)

- find \( q_{\text{near}} = \text{closest point} \) \( q \in V \) to \( q_{\text{rand}} \)

- generate \( q_{\text{new}} \): point on line \( (q_{\text{near}}, q_{\text{rand}}) \)

- if \( q_{\text{new}} \in C_{\text{free}} \) AND \( (q_{\text{near}}, q_{\text{new}}) \in C_{\text{free}} \)

then \( V = V \cup \{ q_{\text{new}} \} \), \( E = E \cup \{ (q_{\text{near}}, q_{\text{new}}) \} \)
- try to connect & new to send it successful -> done!