Robotics: Science and Systems

Path and Motion Planning I

Zhibin (Alex) Li
School of Informatics
University of Edinburgh
Outline

- Motion planning: goals and challenges
- An intuitive approach: Potential Fields
- Rapidly exploring Random Tree (RRT): principle and code demo
- Variants of RRT algorithms
Motion Planning

Fundamental task: plan collision-free motions for complex systems from a start to a goal position among a set of obstacles. For mobile robots, it is also referred as a path planning problem.

Robot Motion: An Observation

1. Most of us have some prior exposure to equations of motion, e.g., Newton’s law $f = ma$
2. Given a physical setup, and given all parameters, it is clear what it means to “compute forward” the motion of robot
3. The robot’s problem is the opposite one: define (ill-posed) requirements and compute actions to achieve complex goals
What is a Path Planning Problem?

Elements of the problem:
- Description of environment, e.g., a map
- Positions of obstacles, terrain properties, etc.
- Description of the robot and its capabilities, e.g., geometry of body, ability to move, etc.

Problem: Given the above elements and start & goal points (sets), write a program to get from start to end
  - How to define feasible paths?
  - Can we have additional costs?
History

Reactive Paradigm (mid-80’s): no models, relies heavily on good sensing

Hybrids (since 90’s): model-based at higher levels, reactive at lower levels

Probabilistic Robotics (since mid-90’s): seamless integration of models and sensing, inaccurate models, inaccurate sensors

Paradigms

Sampling-Based Planning: RRT, PRM

Alternative approaches to the sampling-based paradigm: potential-field-based techniques.
Concepts: Configuration Space

Configuration is a key concept for motion planning: a complete specification of the position of every point in the robot geometry, or a configuration $q$.

Define by $A$ the complete description of the geometry of a robot, and by $W$ a workspace. The goal of motion planning to find a collision-free path for $A$ to move from an initial configuration to a goal configuration.

The configuration space or C-space: the space of all possible configurations.
The C-space obstacle region, $C_{obs}$, is defined as

$$C_{obs} = \{ q \in C \mid A(q) \cap O \neq \emptyset \}$$

Since $O$ and $A(q)$ are closed sets in $W$.

Thus, the free space that avoids collision is the set of configurations:

$$C_{\text{free}} = C \setminus C_{obs}$$
Geometric Path Planning Problem

1. A workspace $W$, where either $W = \mathbb{R}^2$ or $W = \mathbb{R}^3$.

2. An obstacle region $O \subset W$.

3. A robot defined in $W$ by a rigid body $A$ or a set of $m$ links: $A_1, A_2, \ldots, A_m$.

4. The configuration space $C$, $C_{\text{free}}$, $C_{\text{obs}}$.

5. An initial configuration $q_I \in C_{\text{free}}$.

6. A goal configuration $q_G \in C_{\text{free}}$. The initial and goal configuration are often called a query $(q_I; q_G)$.

Compute a continuous path $\tau : [0, 1]$ in $C_{\text{free}}$ such that $\tau(0) = q_I$, $\tau(1) = q_G$. 
Artificial Potential Fields
Potential Fields

- Imagine a 1-dim ball rolling within a flat bowl
- Where will it eventually end up after long time interval?
- What happens if you push the ball around with your finger?
- If you “create” such a field on your 1-dim flat world, where will the ball go?

Stable fixed point
You could play the same game in higher dimensions.

With contours that shrink down to a point, the ball will move in the direction that decreases a measure of height.

The effect on a 2-dim workspace is that the ball will converge to a fixed point.
Why is this useful?

- Imagine a mobile robot in situations where low-level control isn’t perfect
  - Wheels can slip
  - Support surfaces may change
  - Rover could get pushed around in high winds

- Nice to have the notion of convergence to goal right at the planning level
Potential Fields

- Artificial potential field approach is originally proposed for collision avoidance. It constructs a differentiable real-valued function

\[ U : \mathbb{R}^m \rightarrow \mathbb{R} \]

called a potential function, which guides the motion of the moving object.

- Treat the value as ‘energy’

- Then, gradient is the vector,

\[ \nabla U(q) = DU(q)' = [\frac{\partial U}{\partial q_1}(q), \ldots, \frac{\partial U}{\partial q_m}(q)]' \]

Property: Work done along a closed path is zero

Discuss Why?
Attractive and Repulsive Components

Potential field consists of:

1. an attractive component $U_a$, which pulls the robot towards the goal;
2. and a repulsive component $U_r$, which pushes the robot away from the obstacles.

(Picture source: Springer Handbook of Robotics ISBN 978-3-319-32552-1)
Attractive and Repulsive Components

Attractive Component:

\[ U_a(x) = \frac{1}{2} k_p (x - x_d)^2 \]

Repulsive Component:

\[ U_r(x) = \frac{1}{2} \eta \left( \frac{1}{\varrho} - \frac{1}{\varrho_0} \right)^2, \text{ if } \varrho \leq \varrho_0 \]
\[ U_r(x) = 0, \text{ otherwise} \]

\( \varrho \): shortest distance to the obstacle

\( \varrho_0 \): limit distance of the potential field influence
Potential Field: attractive + repulsive components

Online motion planning with PF

**Input:** Function, $\nabla U(q)$

**Output:** Sequence $[q(0), q(1), ... q(i)]$

1. $q(0) = q_{\text{start}}$
2. $i = 0$
3. while $\nabla U(q(i)) \neq 0$
4. $q(i + 1) = q(i) + \alpha(i)\nabla U(q(i))$
5. $i = i + 1$
6. end while

(Picture source: Springer Handbook of Robotics ISBN 978-3-319-32552-1)
Potential Field: attractive + repulsive components

What are the possible paths given different initial configurations?

Can you get any insight of possible drawbacks of this approach?
Limitations: local minima

- An issue with all gradient descent procedures: trapped in **local minima**.

- This issue is not limited to concave obstacles acting as traps, e.g., see the dual obstacles.

- There are solutions such as adding random walks to get out of local minima, but this problem is generally better resolved by sampling based methods.
Limitations: local minima
Limitations: unstable oscillation

Oscillations caused by disturbances or the discontinuity of the obstacles.

Oscillations in narrow passages: oscillation motion occurs when the robot is traveling in a narrow passage with high speed, because the robot receives repulsive forces from both side of the wall.
An example of using potential field

Extension of potential-field:
A potential-field based method, Indicative Route Method (IRM), for global path planning problem. Corridor Map Method is proposed to avoid local minima.

An example of using potential field

Pros:

➢ computationally fast
➢ if it works, usually it produces quite natural motions

Cons:

➢ Solution is not guaranteed, needs manual tuning
➢ Solution is neither complete nor optimal

* Completeness: the solution must be found if it exists.
Sampling-based Planners

A standard for sampling-based planners is to provide a complete/feasible solution if a solution path indeed exists, then the planner will eventually find it.

Sampling-based approach is not meant to search for the optimal solution, rather it is sub-optimal.

Common categories for the sampling-based planners are:

1. Single-Query Planners: incremental search; a single set of initial-goal is given to the planning algorithm;
2. Multi-Query Planners: mapping the connectivity of $C_{free}$.
Single-Query Planners

This class of methods probe and search the continuous C-space by: 1. extending tree data structures initialized at these known configurations; 2. connecting them eventually.

RRT is a single-query planner.
Basic RRT algorithm

Pseudo code

Algorithm 1 BUILD_RRT(q_{init})
\[
\begin{align*}
\mathcal{T}.\text{init}(q_{init}); \\
\text{for} \ k = 1 \ \text{to} \ K \ \text{do} \\
\quad q_{rand} \leftarrow \text{RANDOM_CONFIG}(); \\
\quad q_{near} \leftarrow \text{NEAREST_NEIGHBOR}(q_{rand}, \mathcal{T}); \\
\quad \text{if} \ \text{edge_valid}(q_{rand}, q_{near}) \ \text{then} \\
\quad\quad \mathcal{T}.\text{add_vertex}(q_{rand}); \\
\quad\quad \mathcal{T}.\text{add_edge}(q_{near}, q_{rand}); \\
\quad\quad \text{if} \ \text{close_to_goal}(q_{rand}) \ \text{then} \\
\quad\quad\quad \text{return} \ \text{SUCCESS} \\
\text{return} \ \text{FAILURE}
\end{align*}
\]
Basic RRT algorithm

- $q_{initial}$
- $q_{near}$
- $q_{rand}$
- $q_{goal}$
Basic RRT algorithm

- $q_{\text{initial}}$
- $q_{\text{near}}$
- $q_{\text{rand}}$
- $q_{\text{goal}}$
Basic RRT algorithm

If there is an obstacle lying between $q_{\text{near}}$ and $q_{\text{rand}}$, the edge travels up to the obstacle boundary, as far as allowed by the collision check algorithm.
Basic RRT algorithm

If there is an obstacle lying between $q_{\text{near}}$ and $q_{\text{rand}}$, the edge travels up to the obstacle boundary, as far as allowed by the collision check algorithm.
Basic RRT algorithm

If there is an obstacle lying between $q_{\text{near}}$ and $q_{\text{rand}}$, the edge travels up to the obstacle boundary, as far as allowed by the collision check algorithm.
Basic RRT algorithm
Basic RRT algorithm

$q_{\text{initial}}$  

Obstacle  

$q_{\text{goal}}$
Basic RRT algorithm

To enable fast convergence, force $q_{\text{rand}} = q_{\text{goal}}$ for every $n^{th}$ iteration.
Basic RRT algorithm

To enable fast convergence, define a collision free region, the search succeeds if a collision free pair \((q_{\text{rand}}, q_{\text{near}})\) falls into this safe region.
In this example, number of iterations is 21.

- : initial
- : goal
Same initial & final positions for 15 runs

Do you see any interesting features?
We create a local minima problem encountered in the potential field approach. In this example, number of iterations is 114.
RRT Example

- We create a local minima problem encountered in the potential field approach.
- In this example, number of iterations is 29.

Number of iterations varies!
A trap, a local minima problem encountered in the potential field.
Number of iterations: 569.

"His path-planning may be sub-optimal, but it's got flair."
Variants of RRT algorithms

1. **RRT-Connect**: grow and connect two RRTs
2. Rapidly-exploring random graph (RRG) and RRT*: a variant of RRT that converges towards an optimal solution
3. **RRT*-Smart**:
4. **A*-RRT and A*-RRT**
5. **RRT*FN**: RRT* with a fixed number of nodes, which randomly removes a leaf node in the tree in every iteration
6. **RRT*-AR**: sampling-based alternate routes planning
RRT-Connect: grow and connect two RRTs

- A simple greedy heuristic that aggressively tries to connect two trees, one from the initial configuration and the other from the goal.

- The idea of constructing search trees from the initial and goal configurations comes from classical AI bi-directional search.
This approach finds a graph that covers the space nicely that is independent of the query, ie RRT-Connect is still single query planning.

RRT-Connect

Bi-directional search
Extensions of RRT algorithms

Bi-directional search

Multi-directional search

Some problems may still be too hard!