Potential Field Guided Sampling Based Obstacle Avoidance

Reid Rizvi Rahman
Sakshi Sinha

Indian Institute of Technology, Kanpur

AI Project Presentation
5th March 2014

Introduction to the Problem

Introduction to the Problem

Given a source S, a destination D and an obstacle space Q we need to compute a smooth path from S to D avoiding Q.

Introduction to the Problem

Given a source S, a destination D and an obstacle space Q we need to compute a smooth path from S to D avoiding Q.

In our model, we will represent this path as a sequence of nodes S, X_1, X_2, X_3, ..., X_n, D such that there is an edge connecting every pair of consecutive nodes and none of these edges collide with the obstacle space Q.

Previous Work

Previous Work

- Probabilistic Roadmap (PRM)

Previous Work

- Probabilistic Roadmap (PRM)
- Rapidly Exploring Random Tree (RRT)

Previous Work

- Probabilistic Roadmap (PRM)
- Rapidly Exploring Random Tree (RRT)
- Rapidly Exploring Random Tree Star (RRT*)

Previous Work

- Probabilistic Roadmap (PRM)
- Rapidly Exploring Random Tree (RRT)
- Rapidly Exploring Random Tree Star (RRT*)
- Artificial Potential Field (APF)

Probabilistic Roadmap

Probabilistic Roadmap

- Takes random samples from the configuration space of the robot
- Use a local planner to connect these configurations to other nearby configurations
- Insert the starting and goal configurations
- Use any shortest path graph algorithm to determine a path from source to destination

Rapidly Exploring Random Tree

Rapidly Exploring Random Tree

- Grows a tree rooted at the start configuration using random samples
- Creates an edge between the sample and the nearest tree node
- Adds the randomly generated node to the tree
- RRT growth can be biased by increasing the probability of sampling states from a specific area
- The sampling is stopped whenever a sample is generated in the goal region

Rapidly Exploring Random Tree Star

Rapidly Exploring Random Tree Star

- Optimised version of RRT
- Minimises the distance from the root to the tree nodes at each iteration

Artificial Potential Field

Artificial Potential Field

- Finds an artificial potential in the configuration space
- Negative gradient of this potential is the force
- Robot moves in small incremental steps in the direction given by that of the net force
- Robot always follows minimum potential path from the source to the destination

Drawbacks

Drawbacks

RRT* generates a path which is approximately optimal from the source to the destination but the path returned by this algorithm has sharp corners which are practically difficult to negotiate. The convergence to the optimal solution takes infinite time.

Drawbacks

RRT* generates a path which is approximately optimal from the source to the destination but the path returned by this algorithm has sharp corners which are practically difficult to negotiate. The convergence to the optimal solution takes infinite time.

APF generates a smooth path from the source to the destination but leaves the robot stranded at points of local minima.

Our Approach

- Unify the RRT* algorithm with the APF algorithm into a Potential Guided Directional-RRT* algorithm
- Enhance the rate of convergence towards the optimal solution
- A smooth path from the source to the destination

Our Approach

- Generate a random sample
- Apply the potential field model to generate a new point by moving a small incremental distance in the direction of net force
- Carry out the RRT* algorithm treating this new point as the randomly generated point

References

- A.H. Qureshi, K. F. Iqbal, S. M. Qamar, F. Islam, Y. Ayaz and N. Muhammad: Potential Guided Directional-RRT* for Accelerated Motion Planning in Cluttered Environments
- S. Karaman and E. Frazzoli: Sampling-based Algorithms for Optimal Motion Planning
- J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan and M. S. Muhammad: RRT*-Smart: A Rapid Convergence Implementation of RRT*
- S. Byrne, W. Naeem and R. S. Ferguson: Efficient Local Sampling for Motion Planning of a Robotic Manipulator

RRT Pseudo Code

RRT ()

$V \leftarrow \{x_{init}\}$

$E \leftarrow \phi$

for $i = 1, 2, ..., n$ **do**

$x_{rand} \leftarrow SampleFree_i$

$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$ **then**

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$

return $G = (V, E)$

RRT* Pseudo Code

RRT* ()

$V \leftarrow \{x_{init}\}$; $E \leftarrow \phi$

for $i = 1, 2, \ldots, n$ **do**

$x_{rand} \leftarrow SampleFree_i$; $x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$ **then**

$X_{near} \leftarrow Near(G, x_{new}, r, \eta)$

$V \leftarrow V \cup \{x_{new}\}$; $x_{rand} \leftarrow x_{nearest}$

$c_{min} \leftarrow Cost(x_{nearest}) + c(Line(x_{nearest}, x_{new}))$

foreach $x_{near} \epsilon X_{near}$ **do**

if $CollisionFree(x_{near}, x_{new}) \wedge Cost(x_{near}) + c(Line(x_{near}, x_{new})) < c_{min}$ **then**

$x_{min} \leftarrow x_{near}$; $c_{min} \leftarrow Cost(x_{near}) + c(Line(x_{near}, x_{new}))$

$E \leftarrow E \cup \{(x_{min}, x_{new})\}$

foreach $x_{near} \epsilon X_{near}$ **do**

if $CollisionFree(x_{new}, x_{near}) \wedge Cost(x_{new}) + c(Line(x_{new}, x_{near})) < Cost(x_{near})$ **then**

$x_{parent} \leftarrow Parent(x_{near})$

$E \leftarrow (E \setminus \{(x_{parent}, x_{near})\}) \cup \{(x_{new}, x_{near})\}$

return $G = (V, E)$

Potentialised-RRT* Pseudo Code

Potentialised-RRT* ()

$V \leftarrow \{x_{init}\}$; $E \leftarrow \phi$

for $i = 1, 2, \ldots, n$ **do**

$z_{rand} \leftarrow SampleFree_i$; $x_{rand} \leftarrow RandomisedGradientDescent(z_{rand})$;

$x_{nearest} \leftarrow Nearest(G, x_{rand})$; $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$ **then**

$X_{near} \leftarrow Near(G, x_{new}, r, \eta)$

$V \leftarrow V \cup \{x_{new}\}$; $x_{rand} \leftarrow x_{nearest}$

$c_{min} \leftarrow Cost(x_{nearest}) + c(Line(x_{nearest}, x_{new}))$

foreach $x_{near} \epsilon X_{near}$ **do**

if $CollisionFree(x_{near}, x_{new}) \wedge Cost(x_{near}) + c(Line(x_{near}, x_{new})) < c_{min}$ **then**

$x_{min} \leftarrow x_{near}$; $c_{min} \leftarrow Cost(x_{near}) + c(Line(x_{near}, x_{new}))$

$E \leftarrow E \cup \{(x_{min}, x_{new})\}$

foreach $x_{near} \epsilon X_{near}$ **do**

if $CollisionFree(x_{new}, x_{near}) \wedge Cost(x_{new}) + c(Line(x_{new}, x_{near})) < Cost(x_{near})$ **then**

$x_{parent} \leftarrow Parent(x_{near})$

$E \leftarrow (E \setminus \{(x_{parent}, x_{near})\}) \cup \{(x_{new}, x_{near})\}$

return $G = (V, E)$

RandomisedGradientDescent Pseudo Code

RandomisedGradientDescent (z_{rand})

$(\bigtriangledown U_{att}, \bigtriangledown U_{rep}) \leftarrow PotentialGradient(z_{rand})$

$\alpha \leftarrow ComputeStepSize(\bigtriangledown U_{att}, \bigtriangledown U_{rep}, z_{rand})$

$\bigtriangledown U \leftarrow -(\bigtriangledown U_{att} + \bigtriangledown U_{rep})$

$\overrightarrow{D} \leftarrow \frac{(\bigtriangledown U)}{|\bigtriangledown U|}$

$x_{rand} \leftarrow z_{rand} + \alpha * \overrightarrow{D}$

return x_{rand}