Practical Exercise Probabilistic Roadmap (PRM)

Goals:

In this practical exercise you should implement a sampling based motion planner: the probabilistic roadmap (PRM), which also relies on a roadmap as a subspace of the free space.

Please login into the system with your student account.

1.Introduction

The probabilistic roadmap [1] planner is a motion planning algorithm in robotics, which solves the problem of determining a path between a starting configuration of the robot and a goal configuration while avoiding collisions, without explicitly constructing the configuration space.

The basic idea behind PRM is to take random samples from the configuration space of the robot, testing them whether they are in the free space, and use a local planner to attempt to connect these configurations to other nearby configurations. The starting and goal configurations are added in, and a graph search algorithm is applied to the resulting graph to determine a path between the starting and goal configurations.

The probabilistic roadmap planner consists of two phases: a construction and a query phase. In the construction phase, a roadmap (graph) is built, approximating the motions that can be made in the environment. First, a set of random configurations is created. Then, a configuration is connected – if possible - to some neighbours, typically either the k nearest neighbours or all neighbours less than some predetermined distance. Configurations and connections are added to the graph until the roadmap is dense enough. In the query phase, the start and goal configurations are connected to the graph, and the path is obtained by a Dijkstra's or A* shortest path query.

Given certain relatively weak conditions on the shape of the free space, PRM is provably probabilistically complete, meaning that as the number of sampled points increases without bound, the probability that the algorithm will not find a path if one exists approaches zero. The rate of convergence depends on certain visibility properties of the free space, where visibility is determined by the local planner. Roughly, if each point can "see" a large fraction of the space, and also if a large fraction of each subset of the space can "see" a large fraction of its complement, then the planner will find a path quickly.

There are many variants on the basic PRM method, some quite sophisticated, that vary the sampling strategy and connection strategy to achieve faster performance.

Your implementation is for a 5 DOF robot moving in a 2D workspace. The robot has 2 translational degrees of freedom and 3 rotational degrees of freedom. The EASYROB cell *worm.cel* is used.

2. Simple PRM:

The planner creates a simple probabilistic roadmap.

Initial Sampling: The initial sampling is done with random x and y coordinates and random joint angles. Before a sample is placed, it is checked for collision with obstacles. If it intersects an obstacle, it is disregarded. The initial randomly sampled collision free configurations of the robots can be visualized by EASYROB. The initial sampling must construct an “appropriate” number of collision free configurations.

Nearest Neighbours: To connect the nodes to their neighbours, the nearest neighbours must be found. Each node is tested for connection to an appropriate number of nearest neighbours (for example the top 5).

Connection Testing: To test for connection, a series of temporary nodes are created between the two test nodes. Each temporary node is checked for collision with obstacles by testing each link for collision with each obstacle. The nodes are created and checked sequentially. If two nodes are found to have no intersecting temporary nodes between them, they are labelled as connected (an edge is added to the graph). You might count the number of failed connections. The way temporary nodes for the local planner are constructed is not defined. The simplest way is to connect the node pair by a line segment and check for collision along the line with an appropriate step size. One might also use a potential field based planner as a local planner. The number of connected components of the final graph should be evaluated.

Re-Sampling Since the first sampling is (normally) not sufficient to create a roadmap for most environments, a re-sampling is done with points focused near the edges of obstacles. For each node, new nodes (depending of the number of failed connections) are created with a gaussian distribution of coordinates and joint angles with a mean of this node. Since there are more failed connections near obstacles, most of the new samples appear near obstacles.

Re-connection: The same connection algorithm as before is run on all of the nodes and the connections are updated.

Connection Testing for start and goal configuration: Test for connection of start and goal configuration is done in analogy to step Connection Testing.

Roadmap Search: Using the now possibly still partially connected roadmap, a Dijkstra or A* search is done to find an 'optimal' path from the starting configuration to the goal. Unfortunately, since nodes are only connected to their local neighbours, this path is often quite circuitous.

Path Refinement: An obvious way to improve the path is to try to connect nodes of the calculated shortest path with the “last successor node”, which can be reached

from this node by the local planner without collision. This should result in a smoother and simpler path.

3. Obstacle-based PRM:

The planner creates an obstacle-based probabilistic roadmap.

Initial Sampling: The initial sampling is done with random x and y coordinates and random joint angles. Before a sample is placed, it is checked for collision with obstacles. If it intersects an obstacle, a random direction is picked, and the robot is moved and checked until it no longer intersects an obstacle. The initial randomly sampled configurations of the robots can be visualized by EASYROB. The initial sampling must construct an “appropriate” number of collision free configurations.

Nearest Neighbours: To connect the nodes to their neighbours, the nearest neighbours must be found. A distance metric based on the euclidean distance is used. The distance metric might have 'weights' associated with the translational and rotational distances, and can be tuned for straighter paths, or less wiggling. Each node is tested for connection to an appropriate number of nearest neighbours (for example the top 5).

Connection Testing: see above.

Re-Sampling: see above. In addition, intersecting configurations are again moved into safe areas in a random direction.

Re-connection: see above.

Connection Testing for start and goal configuration: see above.

Roadmap Search: see above.

Path Refinement: Another way to improve the path is to try to connect all of the nodes which make up the Dijkstra or A* solution to each other, and then to throw out the path and run Dijkstra or A* again. This generally results in a much smoother and simpler path, particularly in open areas.

4. Task(s)

You should implement a PRM (simple construction). It is recommended to use the Boost Graph Library (BGL) and you are free to use any graph search algorithm, even though a Dijkstra or A* based search is recommended. If you use the given framework, the BGL is expected to be installed. You need not build a library, you can use the BGL functionality by including the appropriate BGL source files into your code. For that purpose, you need to include the boost directory as an additional include directory into your MSVC project.

The following requirements must be fulfilled:

- The step Nearest Neighbour must be done efficiently, for example based on a a k-d tree approach.
- Generating the metrics: number of nodes, number of edges, number of nearest neighbours, construction time, number of connected components
- The random number generator is seeded with 0
- Running the Re-sampling and Re-connection step, until the number of the connected components of the PRM is “low”.

Voltunteer Task

Improve your algorithm implementing an OBPRM (Obstacle based PRM). The same metrics should be evaluated and be compared to the simple PRM.

or

Implement a potential field based local planner for the step Connection Testing for the simple PRM.

Preparation after

You should prepare (pairwise) a document, which

- Describes the task
- Describes the solution in detail including pictures
- Comprises a well documented code
- Describes the lessons learned
- Is uploaded to Moodle in time.

  You check in your document and your code as a .zip file into moodle. Provide only the .sln solution and source files, so that the overall size keeps small.

5. References

[1] https://en.wikipedia.org/wiki/Probabilistic_roadmap