Group Homework 2: Motion Planning

Robot Intelligence - Planning CS4649/7649, Fall 2014
Instructor: Sungmoon Joo

Administrative

- **Due Date:** November 11, 11:59PM, 2014 (extended)
- **Deliverables:** (i) A PDF summary: Describes your work, results and answers to all the questions posed below. Thoroughness in analysis, informative plots, great movies and answers in the reports are the primary component of your grade. (ii) A repository(git, dropbox, etc.): Contains the relevant files (summary, source code, movies, README, etc.) Your code needs to be in an organized form with a README on how to run the code to re-create the results. Email the instructor and TA the PDF summary and the link to your repository.
- **Participation:** Include a page in your summary describing what each group member did to participate in the project, in detail. If someone did not contribute, say so. Please be honest. Don't just say ”even contributions.”
- **Printing:** On Nov. 11, bring a printout of your summary to the class.

1 Navigation Planning - Bug Algorithms for Point Robot

a) **Bug 1 algorithm:** Implement the Bug 1 algorithm for domain1 and 2 in Figure 1. Provide videos of the simulations(upload the movies to your repository and include the snapshots in your summary report). For each domain, compute the ratio

$$CR_{Bug1} = \frac{\textbf{actual traveled distance}}{\textbf{Euclidean distance between init. and goal}}$$

b) **Bug 2 algorithm:** Implement the Bug 2 algorithm for domain1 and 2 in Figure 1. Provide videos of the simulations(upload the movies to your repository and include the snapshots in your summary report). For each domain, compute the ratio

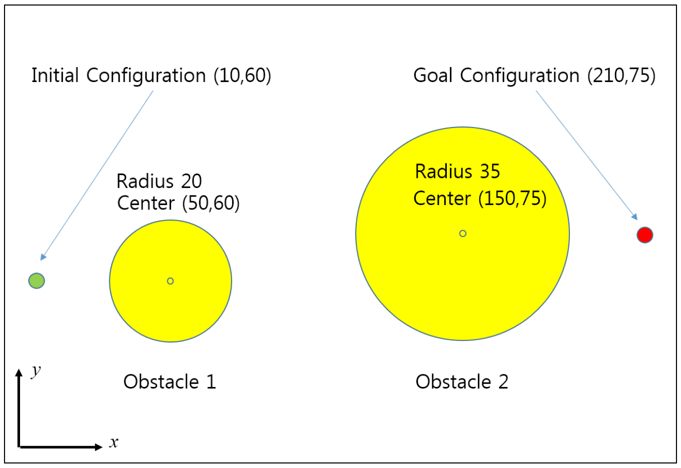
$$CR_{Bug2} = \frac{\textbf{actual traveled distance}}{\textbf{Euclidean distance between init. and goal}}$$

2 Navigation Planning - Potential Field Navigation for Point Robot

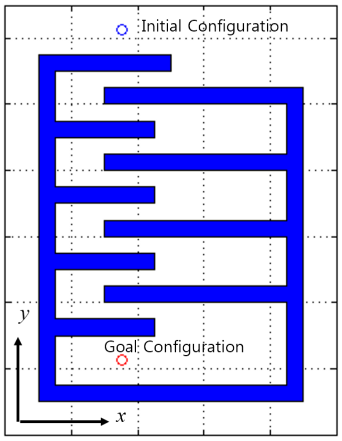
a) **Potential field navigator:** Implement a simple potential field navigator with attractive and repulsive fields for domain1 in Figure 1. Note that unlike the problem 1, your navigator has access to a global map and it will use the map to construct its potential field. Provide a video of the simulation(upload the movie to your repository and include the snapshots in your summary report). Compute the ratio

$$CR_{P.F.} = \frac{\textbf{actual traveled distance}}{\textbf{Euclidean distance between init. and goal}}$$

b) **Local minimum:** Construct an example domain by either adding obstacles to domain1 or changing the positions of obstacles in domain1 such that there is(are) local minimum(minima) and apply your potential field navigator to the modified domain. Show that your P.F. navigator fail to find a solution even when one exists. Provide a video of the simulation(upload the movie to your repository and include the snapshots in your summary report).



(a) Domain1: Two circular obstacles



(b) Domain2: Maze-like obstacle. Domain info. in bugdomain2.txt

Figure 1: Navigation domains

3 Manipulation Planning - Differential Kinematics

Let $\mathbf{x} = f(q)$ be the forward-kinematics function that takes the joint angles q and outputs the pose of the end-effector in the world frame. We know that the end-effector velocity can be written in terms of joint velocities as follows:

$$\dot{\mathbf{x}} = J\dot{q} \tag{1}$$

where J is a Jacobian which has the partial derivatives of $f(q)$ with respect to q. For a 3 degrees of freedom, planar robot arm, the pose of the robot $\mathbf{x}$ would denote the (x, y) location of the end-effector and θ, the orientation(in radian) of the end-effector with respect to the $+x$ axis. In this case, the Jacobian would be a 3×3 matrix with 3 columns for each of the 3 joints. For problem 3 - problem 5, we will be working with a 3 DoF planar robot arm with link lengths $l_1 = 2$, $l_2 = 2$ and $l_3 = 1$ (Figure 2).

a) Move the end-effector from the initial pose of $p_i = (2.6, 1.3, 1.0)$ to the goal pose $p_g = (-1.4, 1.6, -2.0)$ using the inverse Jacobian. The idea is that the end-effector takes small steps towards the goal with a small target velocity $\dot{x}_j$ at each iteration j. Provide the video of the arm as it moves from the initial to goal pose(upload the movie to your repository and include the snapshots in your summary report).

b) You have implemented the Jacobian control approach and probably saw that joint limits and collisions are not taken into consideration. How would you incorporate these two important aspects of planning into this approach? To verify your idea, construct an example with a circular obstacle (radius 1), and implement your idea for collision avoidance(assume there are no joint limits.) Provide a video of the simulation(upload the movie to your repository and include the snapshots in your summary report.)

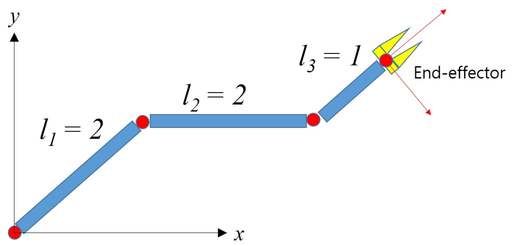


Figure 2: Manipulation domain

4 Manipulation Planning - RRTs

In this problem, you will implement 4 types of RRTs - (a) baseline, (b) goal-directed, (c) connect, (d) bidirectional - and (e)analyse the result. For each implementation, provide (i) a video of the tree as it grows until it finishes expanding and (ii) a video of the arm as it moves from start to goal configuration(upload movies to your repository and include the snapshots in your summary report). Use the same arm as in problem 3.

a) **Baseline:** Create a tree of 300 nodes for both without and with obstacle cases(see scene1.txt for obstacle info.). In this case, no video for arm motion is needed.

b) **Goal-directed:** The idea is to move towards the goal configuration every now and then, and keep expanding randomly other times. The start configuration is $q_s = (0.14, 1.45, 1.0)$ and goal is $q_g = (-2.5, -0.5, -2.1)$. Try for both without and with obstacle cases(see scene2.txt for obstacle info.)

c) **Connect:** The idea is to move towards the goal and take as many steps as possible. Use the same start/goal configurations(as in (b)) and try for both without and with obstacle cases(see scene2.txt for obstacle info.)

d) **Bidirectional:** Instead of growing one tree from the start to goal, we can grow two trees: one from start and the other from goal. Every now and then, select a random configuration and grow both trees with connect-style towards it. If they meet, a path is found. Otherwise, randomly expand. Use the same start/goal configurations(as in (b)) and try for both without and with obstacle cases(see scene2.txt for obstacle info.)

e) **Analysis:** Compare the results of the 3 approaches - goal-directed, connect and bidirectional RRTs- in terms of number of nodes and timings. Note that the RRT algorithms have several parameters such as the number of iterations, step size and etc. You might need to calibrate these values appropriately from problem to problem. Explain which parameters you think played the most important role in the outcomes and why.

5 Manipulation Planning - RRT with Task Constraints

The problem with the classical approach of configuration space RRTs is that it is not possible (or difficult, if possible) to impose workspace constraints on the trajectories in configuration space. For instance, if we want a robot arm to open a drawer, this requires the end-effector to move only horizontally, keeping the same height and vertical location. This constraint can be easily enforced as we build the tree by making sure that each node we add in the tree satisfies the constraint.

How do we do it? The approach is similar to that of a normal RRT sampling. Select a random target, find closest neighbor and expand towards it to find a candidate node q_c. However, there is no guarantee that this candidate node fulfills the workspace criteria. Now, we can compute an error between the pose of the candidate node p_c and the workspace criteria, and move the node q_c to a goal node q_g with Jacobian control such that the error is diminished. See the papers [1] and [2] for more details.

Move the same 3-DoF arm(as in problem 3) such that it would move from the start configuration $q_s = (1.5707, -1.2308, 0)$ to the goal configuration $q_g = (1.5707, 1.2308, 0)$ with the constraint end-effector y location fixed at $y = 3$ line. Provide the videos of the trees growing and the arm moving(upload the movies to your repository and include snapshots in your summary report.)

References

[1] Stilman, Mike. ”Task constrained motion planning in robot joint space.” Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on. IEEE, 2007.

[2] Kunz, Tobias, and Mike Stilman. ”Manipulation planning with soft task constraints.” Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on. IEEE, 2012.