Robotics 1: Tutorial #4 Submission

Completed by: Jenna Kelly Luchak
CID: 01429938

December 8, 2017

Question 1

a)

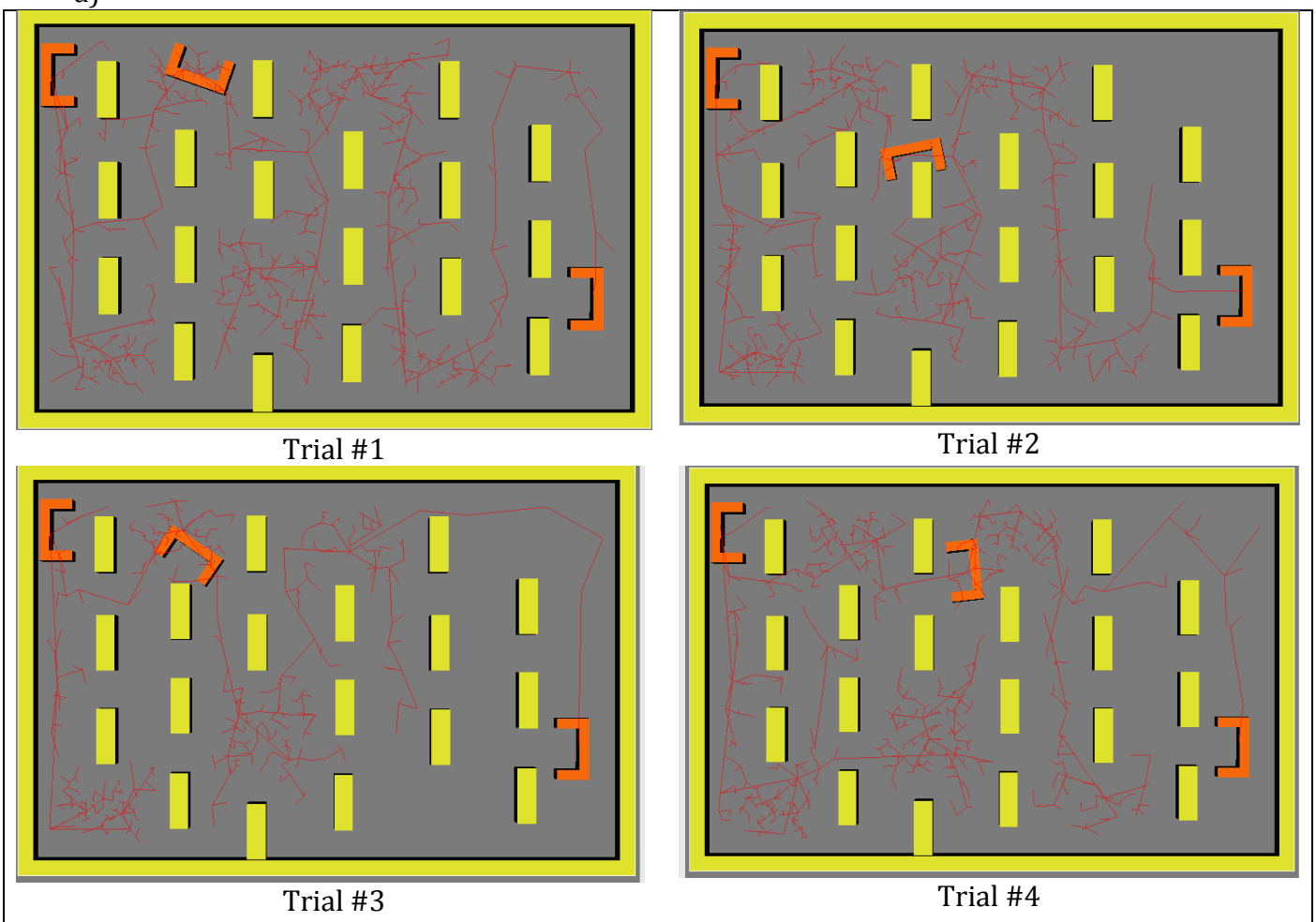

Figure 1: The resulting exploration trees from four trails of a rigid body motion planning simulation set up with Rapidly Exploring Random Trees (RRT) algorithm and the same parameters for all trials.

Based on the information shown in Figure 1, it can be seen that as the task was re-solved several times, different branched patterns of successful routes from start to goal pose were created. This is an expected result because the RRT planning algorithm is not a repeatable method. It creates motion plans by using an algorithm method that operates by randomly selecting a point and then expanding the nearest vertex of the pathway toward it. Since the RRT's planning algorithm operates on a randomized system, every time the simulation is re-solved it is expected that a new pathway be observed.

When the collision checking resolution parameter value of the planner was adjusted, the complexity of the branches in the motion plan changed. For example, when collision checking resolution was increased from 0.01 to value such as 0.05, the extensions from each vertex were longer and included fewer branches in it's plan. Additionally, the "successful" pathway that the program planned was more likely to collide through obstacles when the collision checking resolution was increased. The opposite occurred when the collision checking resolution was decreased from 0.01. The resulting motion plan had a variety of more branched pathways and the extensions from each vertex were shorter.

b)

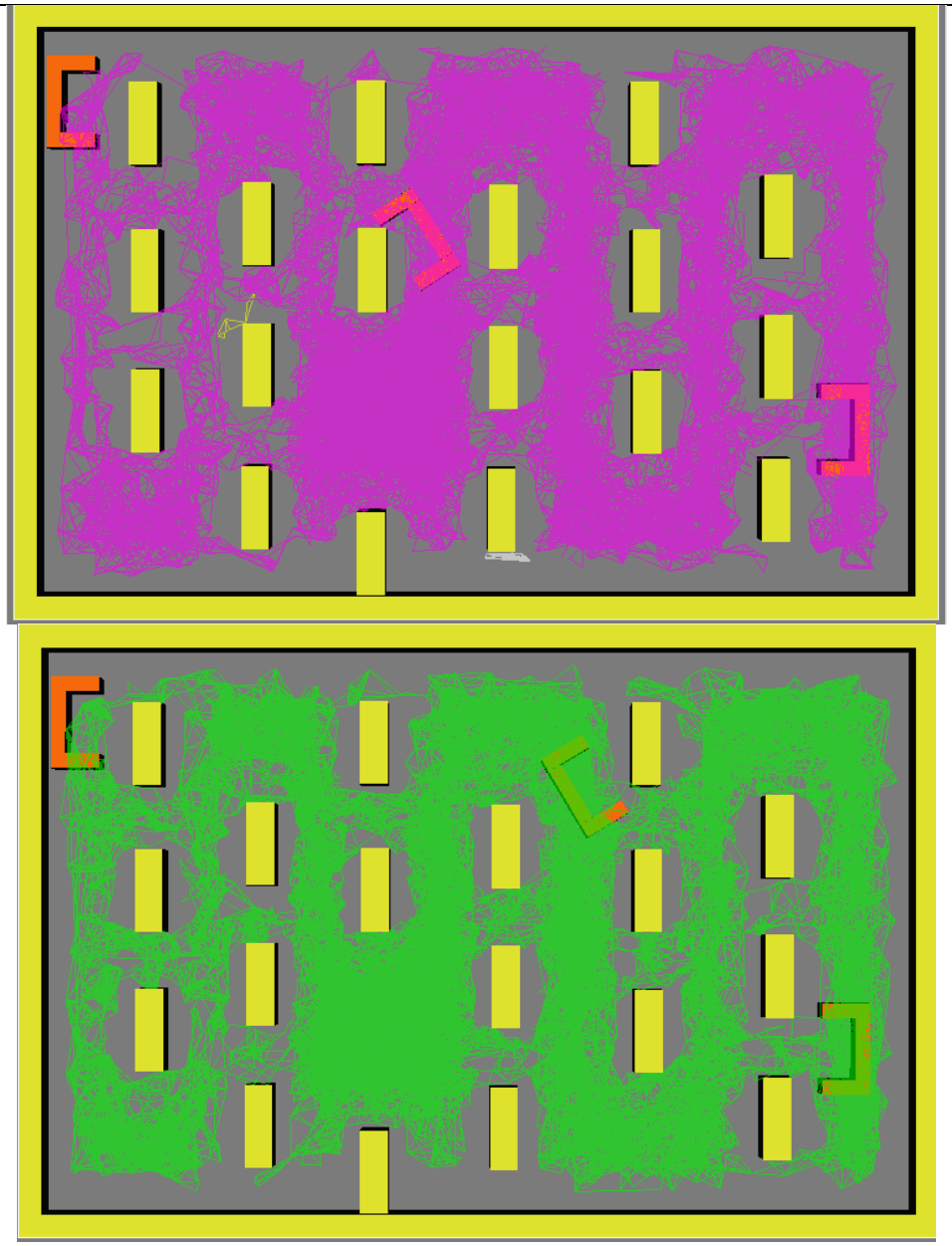

Figure 2: The resulting exploration trees of two rigid body motion planning simulations set up with Probabilistic Roadmaps (PRM) algorithm. [Top] Max nearest neighbors parameter set to 8. [Bottom] Max nearest neighbors parameter set to 30.

Although it may be difficult to observe easily, in Figure 2, it can be seen that the PRM simulation with 30 maximum nearest neighbors (bottom image) resulted in a denser road map than the simulation run with 8 nearest neighbors (top image). The PRM algorithm works by first generating a random configuration to create an initial pathway and then that configuration is connected to some specified number of nearest neighboring configurations, until it ultimately results in a roadmap of multiple pathways. Therefore, the result from Figure 2 is expected because the initial configuration will be connected to its 30 nearest neighbors, resulting in a more crowded and denser roadmap than if it were connected to its nearest 8.

c) In general, the information a state needs includes, the geometry of it's surrounding environment, geometry of obstacles, geometry of itself, the x, y, z and theta coordinates of all milestones and nodes so that the state know whether or not to translate or rotate to avoid it, and the proper parameter initial configuration settings i.e. the collision checking resolution should be low enough in the simulation.

The information each generated state should contain in order for the motion plan to result in a feasible collision free trajectory depends on its surrounding environment. For instance, if the state needs to move through a narrow passage to get to it's goal, then the state will need information regarding the size of the passage, the shape of the passage, size and shape of itself and whether or not the state needs to rotate. However, if the state has to move through a maze, then the state will need information for translational movements such as the dimensions of the maze walls.

With respect to the RRT algorithm, I would recommend using it for a dynamic environment for real-time motion planning because although it needs to be re-run every time a parameter or the environment changes in order for the results to be updated, the algorithm simulates fast. The RRT is inherently biased to grow towards large unsearched areas of the problem; therefore, it can easily handle problems with obstacles and differential constraints in a reasonable amount of time.

With respect to the PRM algorithm, I would not recommend using this system for a dynamic environment because of its slow computational speed. The PRM needs to be re-simulated every time it's environment changes, and since the PRM works by connecting configurations to create multiple feasible pathways, the simulation is very successful, but redundant. In a dynamic environment, the PRM algorithm would be too slow and redundant for this application.

Question 2

a)

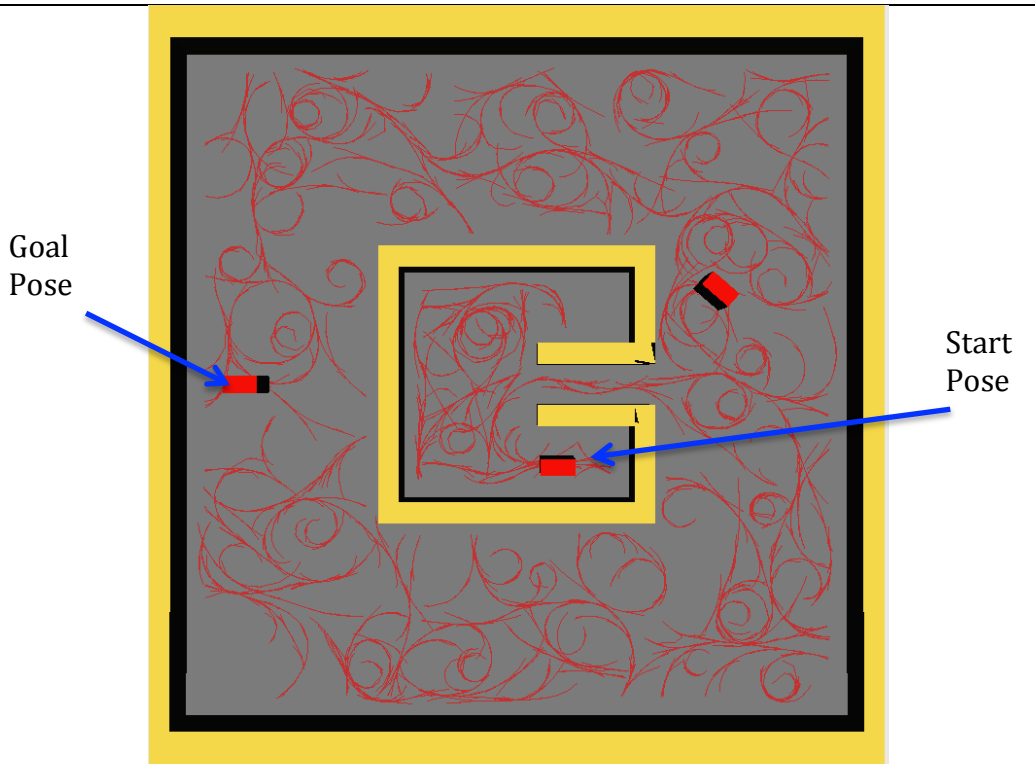

Figure 3: The successful motion plan of the kinodynamic simulation using the Rapidly Exploring Random Trees (RRT) algorithm, annotated with the start and goal pose labels.

Table 1: Parameters associated with the successful motion plan of the kinodynamic simulation using the Rapidly Exploring Random Trees (RRT) algorithm.

Robot Type: Dynamic Car	Time: 10.0
Start pose: X=6, Y=-12, Rotation=0	Propagation Step Size: 0.20
Goal pose: X=-40, Y=0, Rotation=0	Control Duration: 0-20
Optimization Objective: Length	Goal Bias: 0.01

The parameters that resulted in the successful plan can be seen in Table above and were identical to the original problems configuration, however Goal bias was changed from 0.05 to 0.01. Goal bias was the one parameter that had to change because the parameter indicates how likely the trees pathway is to tend toward the goal pose. By observing the environment, specifically the location of the central walls around the starting pose in Figure 3, it can be seen that in order to reach the goal pose, the pathway will need to move around this obstacle, and in order to do that, the motion plan will have to move in a direction away from the goal pose. Therefore the goal bias had to be reduced.

b)

There are observed differences between the motion plans generated from the rigid body problems and the kinodynamic problem. For example, the rigid body problems produced branched pathways, where the extensions from each vertex were linear and jagged, where as the kinodynamic motion plan produced branched pathways where the

extensions were curved and smooth paths. Additionally, the trajectories of the kinodynamic motion plans depend on the current state, where as the rigid body does not.

These differences are due to the fact that kinodynamic motion planning is smoother than rigid body motion planning. Kinodynamic motion planning uses both kinematic and dynamic constraints simultaneously, as it uses pre-defined motion primitives to sample the possible future motion of the robot, where as rigid body planning has no constraints.