Maxwell Vale, Amy Pham
ME133b Project Ideas

Focus: Algorithm Development and Analysis

We really enjoyed working with search trees that we've been visualizing in the past homeworks and would like to explore the capabilities of the trees and test variations of the algorithms that we have already implemented. Specifically, we are interested in improving upon the RRT algorithms and/or considering multi-tree variations of RRT.

Multi-tree Analysis

We want to observe the effects of adding multiple trees to our RRT search.

- Add a second tree at the goal and note its effects on the path and speed to which the path is found
    - Better or worse?
    - Closer to the optimal path or not?
- Can we add more trees somewhere in the space and test how that affects as well? For example, what if a third tree was rooted as close to the midpoint between start and goal as possible.

Potential Improvements to RRT/EST

- As described in class, having a dynamic step size may affect the behavior and convergence of the RRT to the goal state in interesting ways that we can explore.
- Consider RRT*, in which we take an algorithm utilizing a cost heuristic to minimize the cost of our path and get towards the optimal path between the start and the goal.
    - After finding some path, consider only points of improvement that are within some ellipse → sample and find a new path → get closer to optimal
        - https://ieeexplore.ieee.org/document/6942976
    - Sure there are some other methods for finding an optimal path for RRT

Plan for the coming week:

We are going to focus on the problem of optimizing the RRT search algorithm (RRT*)

1. Figure out a heuristic for calculating our path costs from the start to the goal
2. Find out how we can apply such a heuristic to create an "improvement ellipse," such that sampling points only in the ellipse will generate a path that is guaranteed to improve our path cost.
3. Using an empty 2D space or one with minimal obstacles, run the RRT algorithm to find a path. Demonstrate the creation of the improvement ellipse and sample the new tree to show improvement upon our path (i.e. does it have lower cost?). Iterate a few times until we get close to the optimal path.