Maxwell Vale & Amy Pham
ME 133b
Project Update #1

Accomplishments in Week 1:

- Modified RRT to give us the shortest path from the root to any new sampled point
    - Previously, after we uniformly sampled a point we would choose the closest node to that point to draw a new edge from. The only problem with this is that the path length to reach this node is not necessarily optimized (not the shortest path to reach that node from the root)
    - Now, upon sampling a node from whatever our sampling space is, we check the k-nearest neighbors of our vertex to be added to the tree and find the one that would minimize the cost to reach that node from the root. So, we have essentially implemented K-Nearest RRT*.
    - To determine K, we use an equation that gives an optimal number of neighbors to check depending on the log of the size of the tree.
- Updated how we sample based on previous solutions
    - The very first pass of RRT is the same as before: sample the space uniformly until a path from the start to the goal is found. After a solution is found, put that path into a list of the solutions thus far and choose the best one that we have found. Now, using this best path, we limit the area that we can sample from for our next try at finding a path. This significantly reduced the area of points that we can sample from, with an almost guarantee that picking in that area will provide us a path with similar or better cost to the one we are using.
- Results
    - Implementing these two gave us Informed K-Nearest RRT*. Informed means that after the initial solution, the algorithm samples from a space that it knows will give it an equally good or better path compared to its last run rather than picking points uniformly. K-Nearest means that we search the K-Nearest neighbors of each point to determine where to draw an edge.
    - When testing a space with no obstacles, the algorithm converges to a straight line between the start and the goal, which we know to be the optimal path in that case.
    - Testing with the triangle obstacles from previous problem sets showed us the effect of obstacles on the informed sampling of our algorithm. The nature of the obstacles is that we can no longer draw a straight path; traveling in a direction not pointed at the goal is necessary. Therefore, the sampling subset has a lower bound to which we can no longer reduce the area we sample from.

Tasks for Week 2:

We want to further explore the performance of Informed K-Nearest RRT* on various scenarios and see how we can further optimize the algorithm.

- Use random obstacles and different placements of start and end goal to see how the algorithm handles finding a path (how small can the sampling subset get, how long does it take to converge on a solution)
- Specifically check scenarios where we know where the optimal path is
    - Paths that require traveling back towards the start could be interesting to see
- See if it is possible to further refine (post process) the path once we have reached the lower bound on our sampling space.
    - Maybe the path can be split into several parts and we can create subsampling spaces for each part of the path
- Other ideas:
    - Try a Informed K-Nearest RRT* with multiple trees?