Maxwell Vale & Amy Pham
ME 133b
Project Update #2

Accomplishments in Week 2:

- Our goal for this week was to tackle more random scenarios and see how the algorithm would perform in finding optimal paths
- This led to us having to further generalize our code to deal with these scenarios
- Elliptical Sampling
    - Originally, the algorithm assumed that the start and goal had the same y-value, resulting in a normal ellipse aligned with the coordinate axes
    - Of course, as soon as we misalign the start and goal, the true ellipse should align with the straight line path between the two points
    - So, we need to now redefine how we sample from the informed elliptical subspace when improving on our solution
        - Introduced more general ellipse equation that takes into account the angle with the x axis
        - Rejection sampling uniformly samples from a rectangular area bounding the ellipse. The values of the ellipse are simple to calculate for an ellipse aligned with the coordinate axes, but more difficult for an angled ellipse
            - Used mathematica to acquire analytical solutions for the max and min x/y values of a given ellipse to draw the bounding box
- Now once we got this more generalized form of the algorithm working, we could test on random obstacles and observe how the algorithm performs in different scenarios
    - Decided to randomly draw triangles of a set size to create a random obstacle map
    - Defaulted the start state at the center of the area and randomly choosing a goal state
        - Noticed that initial sampling takes very long and sometimes cannot find the goal state
        - Since we do not care how optimal this first path is (and more about actually finding one) we let this first search be more greedy and set the sampled state to the goal state some percent of the time (currently 15% of the time)
        - This helps limit the search space, which helps further iterations find the solution faster
            - We still allow the sampled state to be the goalstate around 5% of the time

Next Tasks:

- Post Processing
  - Can perform normal post processing like we have done in the past
  - If the ellipse still allows for taking incorrect paths, can we somehow limit the search to be only around the path we know is shortest?
- Other options
  - Still potentially consider implementing Multi trees ?
  - There is potential for varying step sizes here
    - As the sampling subset shrinks, we will inevitably get closer and closer to the edge of some obstacle assuming one exists between the start and the goal nodes. It gets increasingly more unlikely that you sample a state that allows you to go around the corner since the area around the corner is shrinking
    - Shrinking the step size along with the sampling space may increase the chances that a point is sampling that allows the algorithm to find a path around the corner