5) When we sample with _get_random_sample_near_goal with probability 0.2, the time it takes to find a complete path drastically decreases, as shown in the video. This makes sense since taking random samples in the entire search space would make the RRT algorithm take longer to find a valid path. Comparing the videos, by decreasing the goal precision to 0.2, the results are much more accurate as the robot arm takes a more accurate path to the goal.

6) The original RRT algorithm guarantees convergence to the goal configuration as the number of iterations approaches infinity. Sampling near the goal may hinder this convergence because the exploration might get stuck in a local region around the goal, neglecting other parts of the configuration space that would have been explored using the _get_random_sample function. Thus, the need to call the _get_random_sample_near_goal function with probability 0.2.