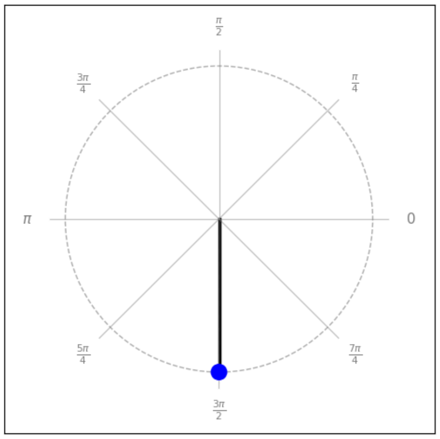
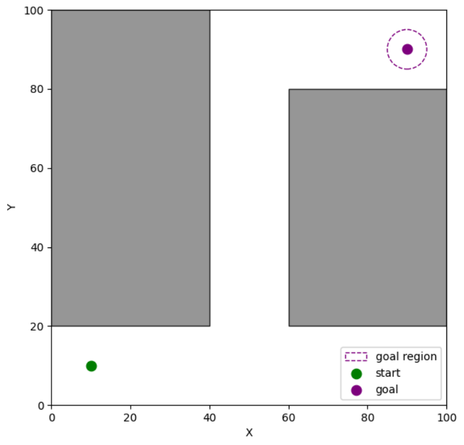
# vc68 cl307 project 4 report

1. In this project, we aim to solve path planning problems for a pendulum and a car (square robot) with dynamical constraints using RRT, KPIECE, and RG-RRT route planning algorithm. In the first part of the experiment, we compare the solution path given under different torque constraints for pendulums. In the second part, we focus on comparing the solution path generated by RG-RRT with RRT and KPIECE with the OMPL benchmarking package.

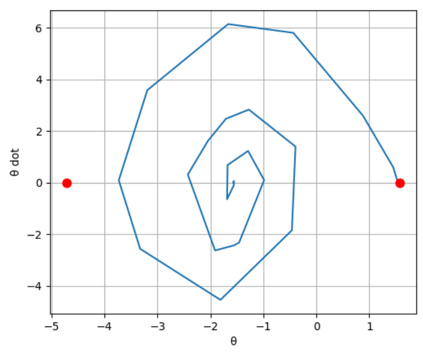
2. **Pendulum**: start (−π/2, 0) (hanging down), goal (π/2, 0) (upright), goal region = 0.05
   **Car**: start (10, 10), goal (90, 90), goal region = 5

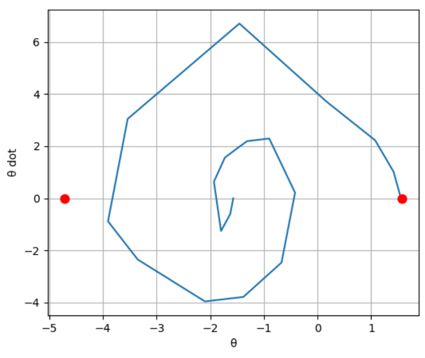


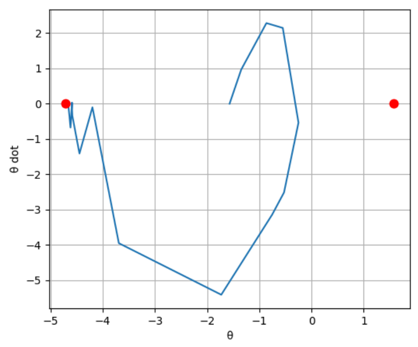


3. The pendulum is a single rigid rod rotating around a fixed point. Its configuration space is S^1 x R (θ,ω), where θ is the rotation angle and ω is the angular velocity.
   The car is a square-shaped robot with a side length of 3 moving on a 2D plane. Its configuration space is SE(2) x R (x,y,θ,v), where x,y represent position, θ is the heading angle, and v is the forward velocity.
   We assume that the control space is a RealVectorControlSpace with bounds.

4. I compare the solution paths of different torque limits using the RRT planner. When the torque limit is 3, the pendulum has less strength and requires multiple swings to reach the upright goal, resulting in more states and a longer path. When the torque limit increases to 5, the pendulum becomes stronger and can reach the target with fewer swings and fewer states. Finally, when the torque limit is 10, the pendulum has enough torque to reach the goal directly without additional swings, so the solution path contains far fewer states and reaches the target much more efficiently.

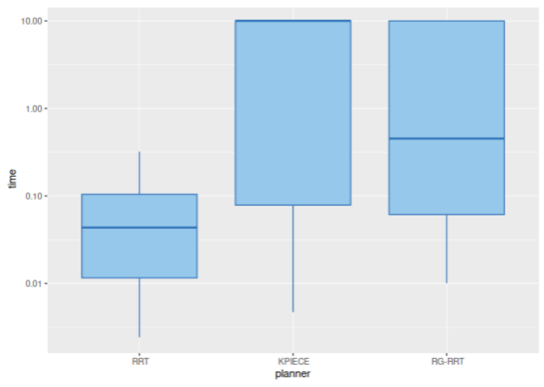


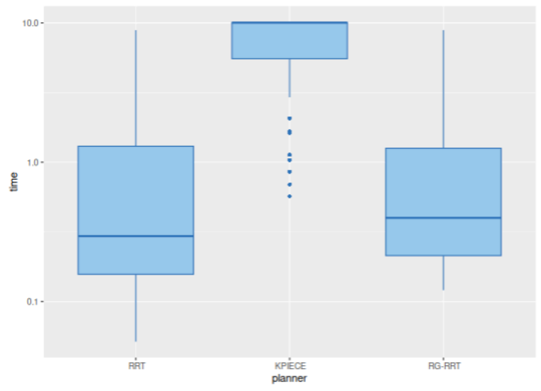




5. Note: The number of trials in each experiment is 50.
   Some of the y-axis is plotted on a logarithmic scale to better visualize differences.

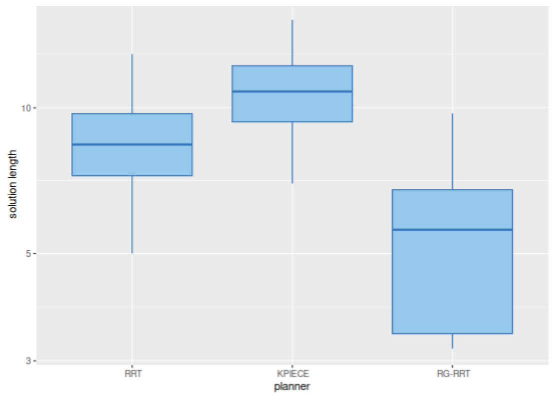
   Computation time (Pendulum, Car):

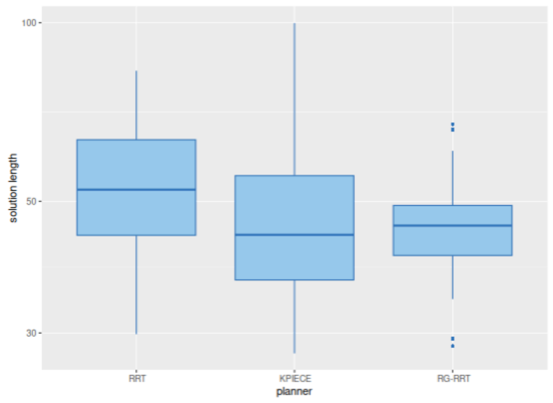




   Among the three planners, RRT achieved the shortest computation time. RG-RRT required slightly more time as we suspect it’s due to the computation of reachable sets and its goal-guided expansion, while KPIECE took the longest time as we guess that it spent more time projecting and exploring the state space.

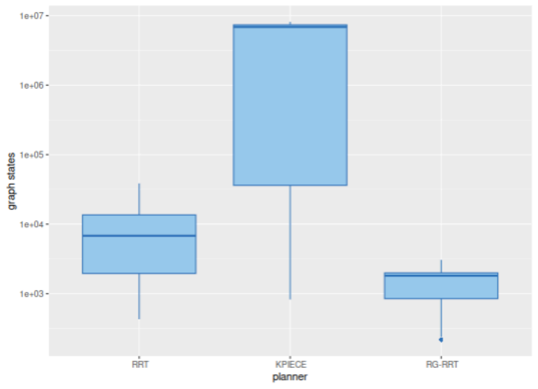
Path length (Pendulum, Car):

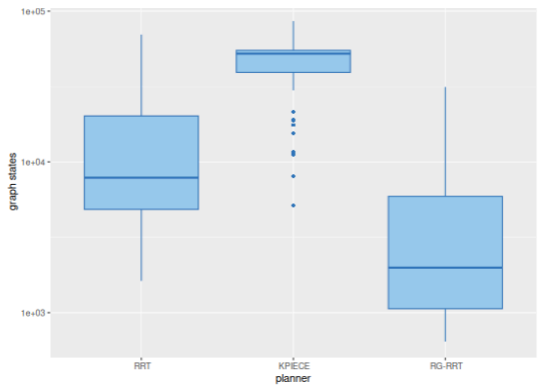




In terms of path length, RG-RRT produced the shortest trajectories as its designed intention is to guide tree expansion toward the goal region using a reachable set. RRT has moderately longer paths since it relies purely on random sampling compared to RG-RRT. KPIECE resulted in the longest trajectories; we suspect it spends more time exploring broader regions of the state space rather than directly targeting the goal. However, in the Car environment, KPIECE actually performed the best; we assume this is because the projection we implemented allowed KPIECE to prioritize useful cells and explore the 2D workspace more efficiently than the other planners.

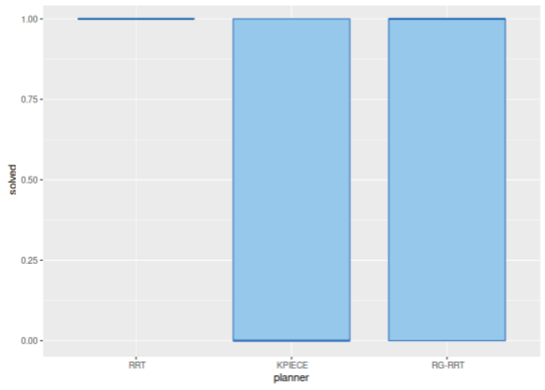
Number of Nodes (Pendulum, Car):

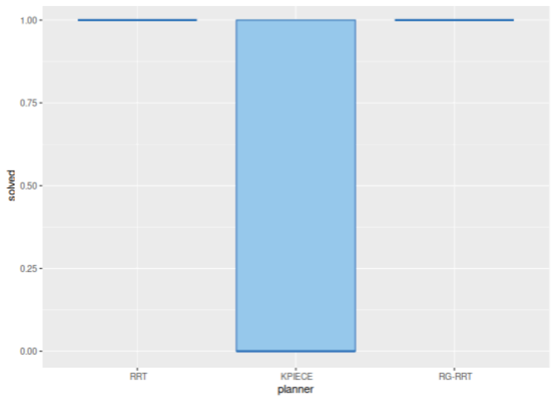




For the number of nodes generated, RG-RRT expanded the fewest nodes because its guided exploration was more focused and efficient. RRT required a slightly larger number of nodes to find valid trajectories due to its randomness. KPIECE produced the largest trees, as it explored more of the state space.

Success Rate (Pendulum, Car):





In terms of success rate, RRT achieved a 100% success rate, consistently finding valid solutions. RG-RRT also performed well with an almost 100% success rate, though some runs resulted in approximate rather than exact solutions. KPIECE, on the other hand, produced mostly approximate solutions, indicating that its performance was less consistent.

6. Comparing the benchmarking graph of RRT and RG-RRT above, we can find the trade off between the planning time and the solution quality (solution length and graph states). We suggest that due to the maintenance of reachable motions, RG-RRT took a slightly longer time to plan and resulted in better solution length and fewer graph states. In comparison, with its straightforward expansion, RRT has lower planning time but longer solution length and more graph states.

7. Exercise 1: diff: 8, time: 6 hrs
   Exercise 2: diff: 7, time: 1 hrs
   Exercise 3: diff: 8, time: 9 hrs
   Exercise 4: diff: 7, time: 3 hrs
   The hardest part of the project is to understand how to add the dynamical constraints and set up the control in the environment. Understanding the core concept in RG-RRT also took us a large amount of time.
   Individual contribution: vc68:
   Car part, RG-RRT except for solve, written report
   cl307: Pendulum part, RG-RRT::solve, written report