RBE550: Motion Planning

Project 4: Out of Control Planning

Student 1: Dhruv, Agrawal
Student 2: Shreyas, Khobragade

1 Theoretical Questions

1. **What is the difference between asymptotic optimality and asymptotic near-optimality?**
Asymptotic Optimality is when given long enough time with probability one, an optimal solution is found if one exists.
Asymptotic-Near Optimality is when given long enough time with probability one, a solution that is within a fixed ϵ-distance from the optimal will be found.

2. **If we have a system with non-holonomic constraints can we use RRT* to find an optimal path? Explain your answer.**
Yes RRT* can be adapted to find optimal paths in systems with non-holonomic constraints, with some modifications.
Instead of choosing a random sample from the C-space, now choose a random sample X_{rand} from the State Space.
Choose a node in a tree X_{near} that is nearest to the random sample based on the non-holonomic constraints.
Generate a local trajectory from the X_{near} to X_{rand} based on the Boundary Value Problem or according any other min path steering function.

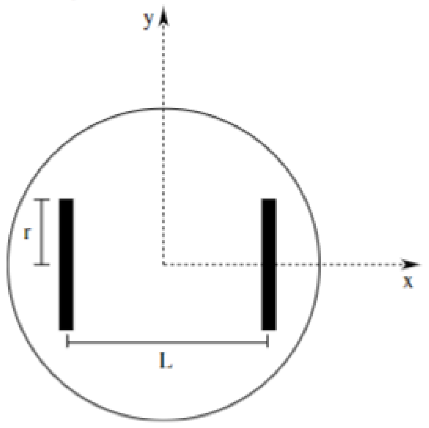

Figure 1: Indoor Vacuum Robot

3. **Figure 1 shows an indoor vacuum robot has a differential drive chassis with two wheels, each with a radius of r, that are separated by a distance L.**
Each wheel is controlled independently with its own motor. Presume that you can specify the torque for each motor, and the motors can directly change the velocities (u_l and u_r) of your Roomba. Then the dynamics of your Roomba can be specified with the following differential equations:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \frac{r}{2}(u_l + u_r)cos\theta \\ \frac{r}{2}(u_l + u_r)sin\theta \\ \frac{r}{L}(u_r - u_L) \end{pmatrix}$$

   - **What is the configuration space, the control space, and the state-space of the robot?** The C-space of the robot is SE2.
   The control space of the robot is u = (u_l,u_r) where u_l, u_r $\in$ R.

The state space of the robot is:

$$X = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

- **Is this a holonomic or non-holonomic robot? explain your answer**
  The robot is non-holonomic because:
  - The constraints on its motion cannot be integrated to reduce the system to a set of independent coordinates. The robot cannot move sideways without changing its orientation, which imposes additional velocity constraint.
  - The non-holonomic nature is characterized by the inability to perform certain motions without specific inputs (in this case moving laterally without turning).
- **Given the above equations of motion and using the Euler approximation, compute the next state of the robot if it starts from configuration (x, y, θ) = (0, 0, 0) and controls (u_l , u_r) = (1, 0.5) are applied for 1 second. Show your work, don't just write a number.**
  Euler approximation formula is given as:

$$X(\Delta t) = X(0) + \Delta t f(\Delta X(t), u)$$

$$X(\Delta t) \approx \begin{pmatrix} x_0 \\ y_0 \\ \theta_0 \end{pmatrix} + \Delta t \begin{pmatrix} \frac{r}{2}(u_l + u_r)cos\theta \\ \frac{r}{2}(u_l + u_r)sin\theta \\ \frac{r}{L}(u_r - u_L) \end{pmatrix}$$

Substituting $(x_0, y_0, z_0) = (0,0,0)$, $\Delta t = 1$, $u_l = 1$, $u_r = 0.5$, we get,

$$X(\Delta t) \approx \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{r}{2}(1.5)cos(0) \\ \frac{r}{2}(1.5)sin(0) \\ \frac{r}{L}(0.5-1) \end{pmatrix}$$

$$X(\Delta t) \approx \begin{pmatrix} \frac{3r}{4} \\ 0 \\ -\frac{r}{2L} \end{pmatrix}$$

2 Programming Exercises

Fill all the necessary functions in car.cpp, pendulum.cpp, CollisionChecking.cpp, CollisionChecking.h RG-RRT.cpp RG-RRT.h to:

1. **Implement the state validity checker and differential equations for the pendulum and the vehicle systems described above by Solve the motion planning problems described for these systems using the RRT planner.**

   - **Visualize the solution paths for car and pendulum (by creating your own script) and attach them in your report to make sure they are correct. Also include a description of the obstacles you used (for the car) and start and goal queries.**
     The solution paths for the car and pendulum are shown below:
     Pendulum Case:
     - RRT Planner for Torque = 3, 5, 10

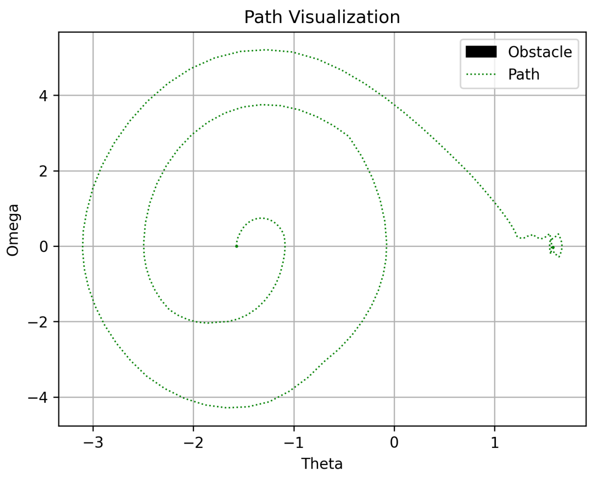

Figure 2: Pendulum path for RRT Planner using Torque=3

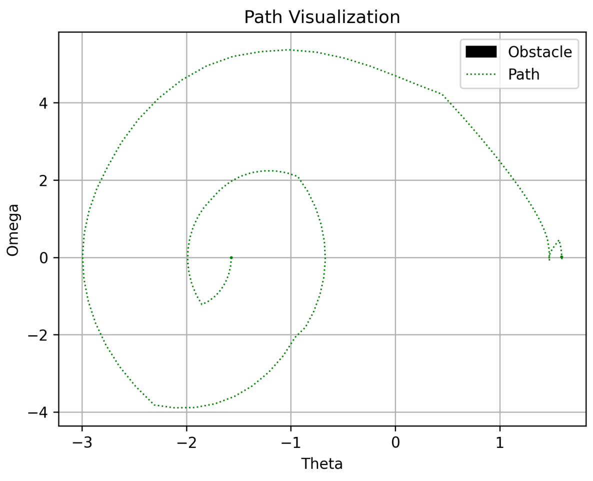

Figure 3: Pendulum path for RRT Planner using Torque=5

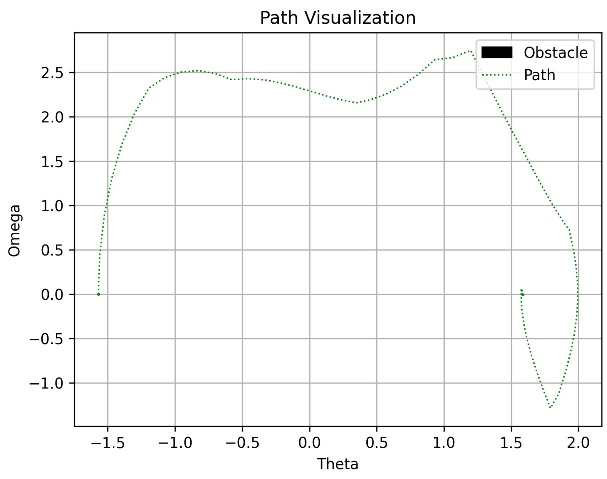

Figure 4: Pendulum path for RRT Planner using Torque=10

Car Case:

– RRT Planner

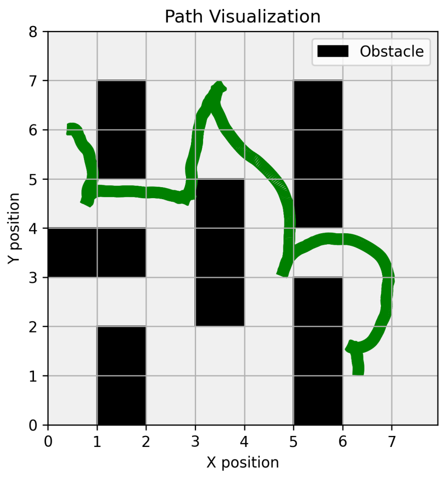

Figure 5: Pendulum path for RRT Planner

Below is the description of the obstacles for the car example:
The environment is bounded within (0,7) in the x-direction and (0,7) in the y-direction.
There are 7 obstacles in the environment having base coordinates and sizes as -

– Obstacle 1: Base coordinate - (0,3), width - 2, height - 1
– Obstacle 2: Base coordinate - (1,5), width - 1, height - 2
– Obstacle 3: Base coordinate - (1,0), width - 1, height - 2
– Obstacle 4: Base coordinate - (3,2), width - 1, height - 3
– Obstacle 5: Base coordinate - (5,4), width - 1, height - 3
– Obstacle 6: Base coordinate - (5,0), width - 1, height - 3

The Start and Goal conditions are -
Pendulum: Start = (-$\frac{\pi}{2}$,0), Goal = ($\frac{\pi}{2}$,0)
Car: Start = (0.5, 6, 0, 0), Goal = (6.5, 1, 0, 0)

- **Compare the solution paths found using torque values of 3, 5, and 10 for the pendulum problem. Explain the differences in solution paths when torque is limited to 3, 5, and 10 for the pendulum problem.**
The path using input as Max Torque limits of abs(3) is the longest and has more loops as it moves back and forth a lot before terminating in the goal region.
The path using input as Max Torque limits of abs(10) is the shortest and has least/no loops as it does not

oscillate back and forth before terminating to the goal region.
Path(Torque = 3) ≥ Path(Torque = 5) ≥ Path(Torque = 10) Higher torques lead to shorter paths and lesser convergence time, however as torque increases, the system's sensitivity to control inputs also increases, making it more challenging to stabilize the pendulum.

2. **Extend the program from above to solve the pendulum and the vehicle problems using the KPIECE planner. You will need to define a projection for the state spaces you create for the pendulum and the car. See projections for details on how to define a projection and associate it with a state space.**
Pendulum Case:

- KPIECE Planner for Torque = 3, 5, 10

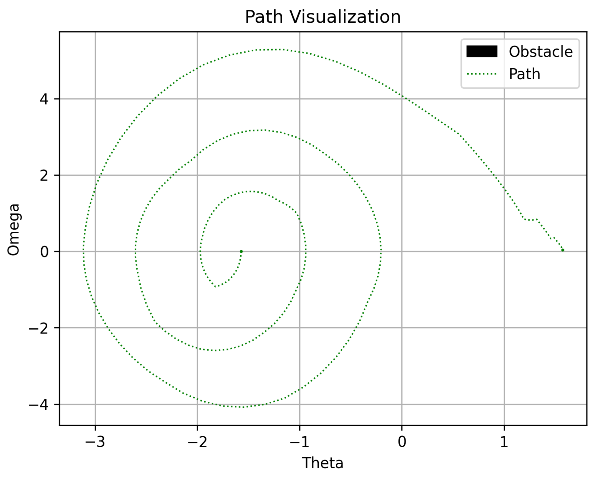

Figure 6: Pendulum path for KPIECE Planner using Torque=3

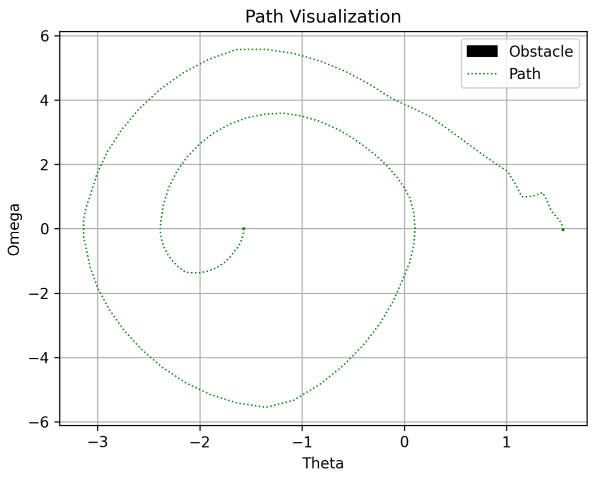

Figure 7: Pendulum path for KPIECE Planner using Torque=5

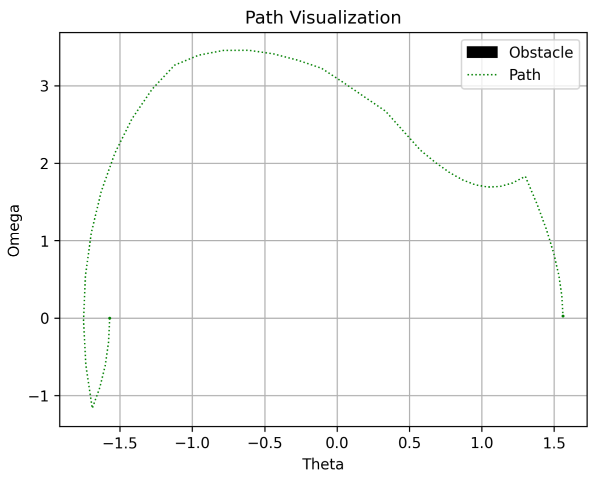

Figure 8: Pendulum path for KPIECE Planner using Torque=10

Car Case:

- KPIECE Planner

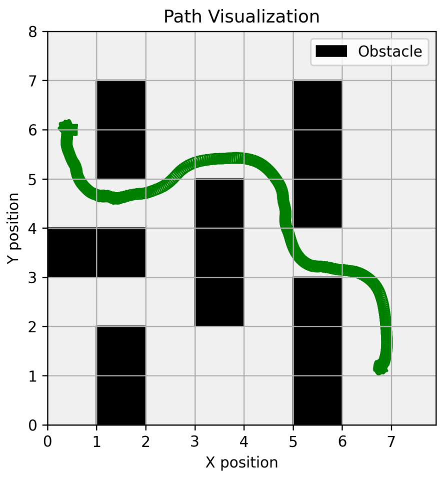

Figure 9: Pendulum path for KPIECE Planner

3. **Implement RG-RRT (see Scholnik et al., 2009) and solve the pendulum and vehicle problems as in. Make sure to visualize the solution paths, and attache them in your report. To implement RG-RRT, we recommended you start with the control RRT implementation from the ompl control rrt.**
Pendulum Case:
    - RGRRT Planner for Torque = 3, 5, 10

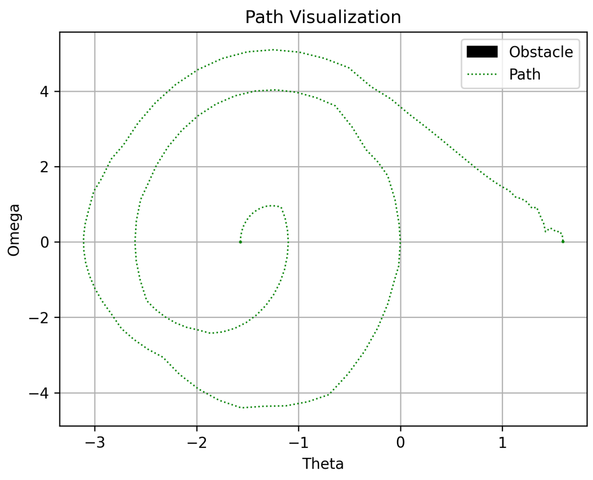

Figure 10: Pendulum path for RGRRT Planner using Torque=3

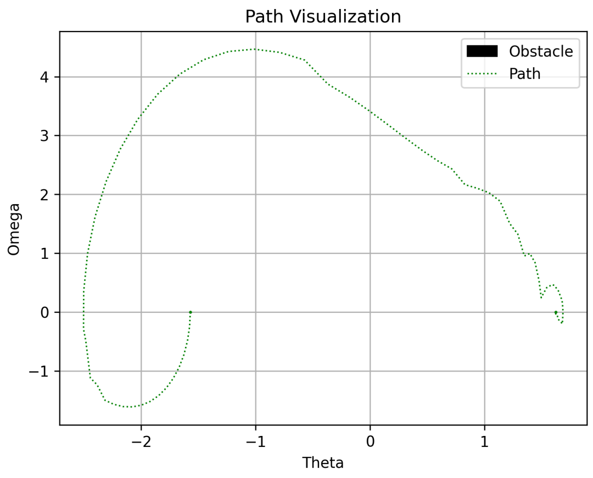

Figure 11: Pendulum path for RGRRT Planner using Torque=5

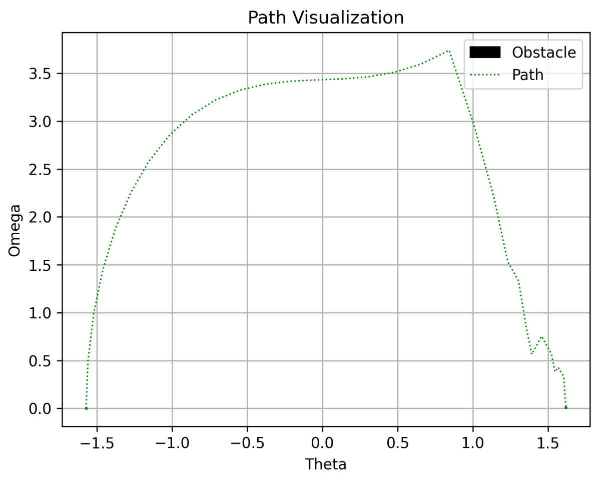

Figure 12: Pendulum path for RGRRT Planner using Torque=10

Car Case:RGRRT Planner

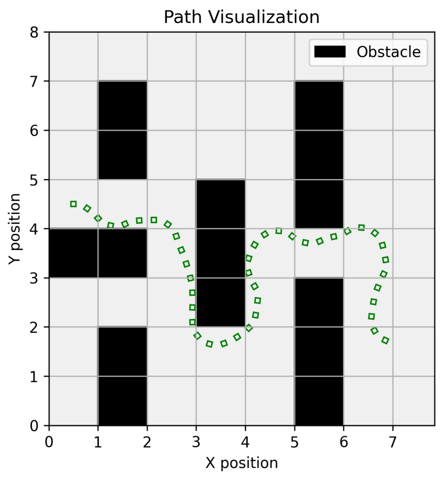

Figure 13: Pendulum path for RGRRT Planner

–

4. **Compare your RG-RRT against RRT and KPIECE using the OMPL Benchmark class for both the car and pendulum environments. Use a torque value of 3 for the pendulum. Any conclusions must come from at least 20 independent runs of each planner. Consider the following metrics: computation time, path length, number of tree nodes, and success rate. When referencing benchmarking results you must include the referenced data as figures. Discuss the performance trade-offs of RG-RRT for computing reachable states in terms of the length of the time period and the number of controls. Support your claims with images and/or benchmark data.**
RGRRT is evaluated against RRT and KPIECE on the following metrics:
   - Computation time
   - Path length
   - Number of Tree nodes
   - Success rate

Pendulum Case:

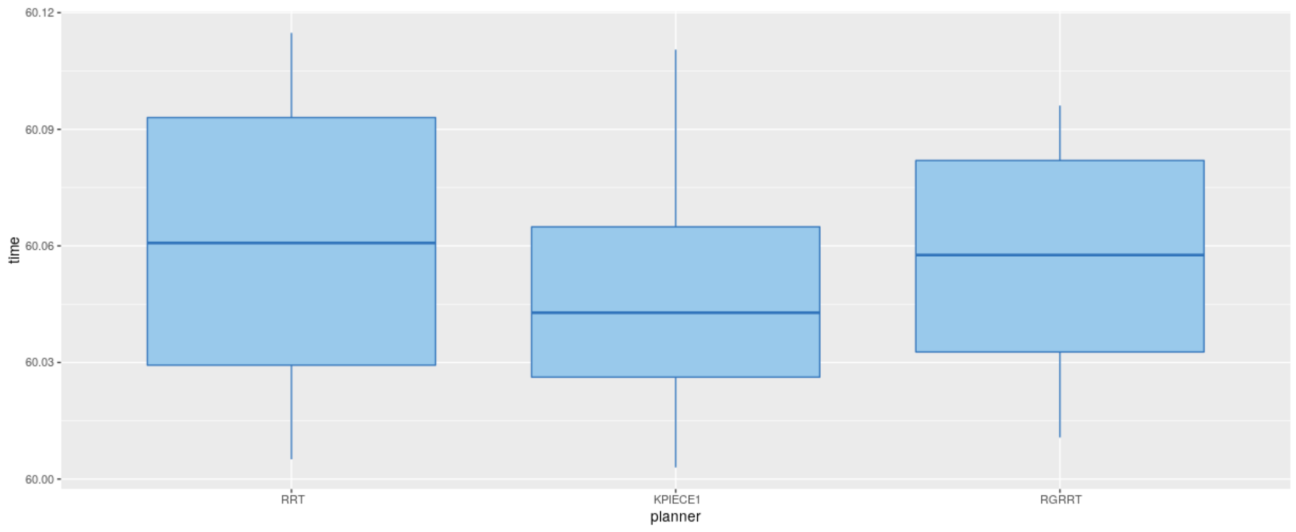

Figure 14: Computation Time

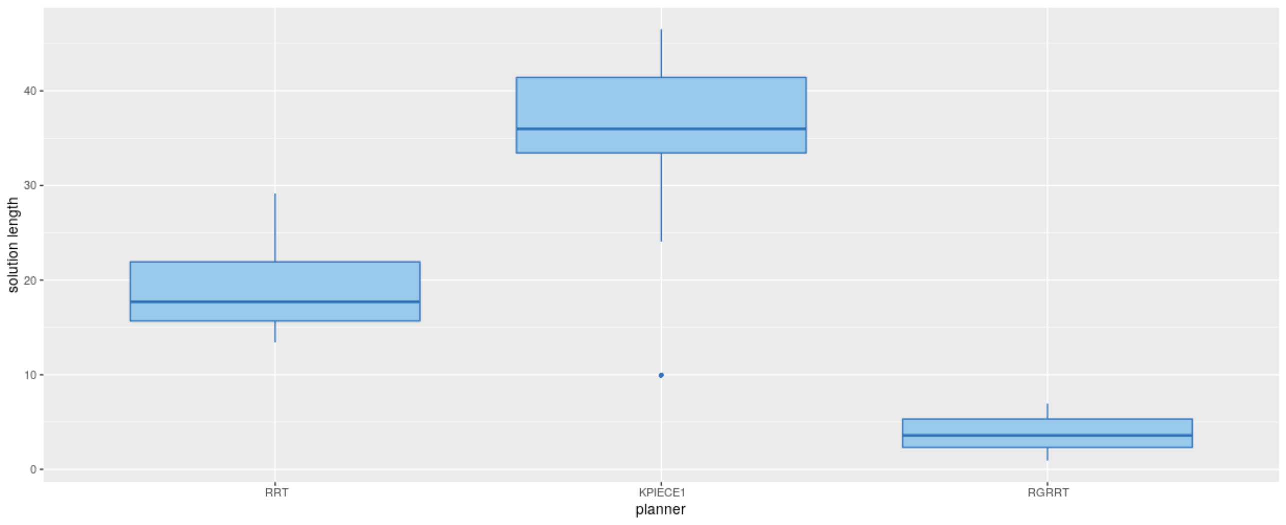

Figure 15: Solution Length

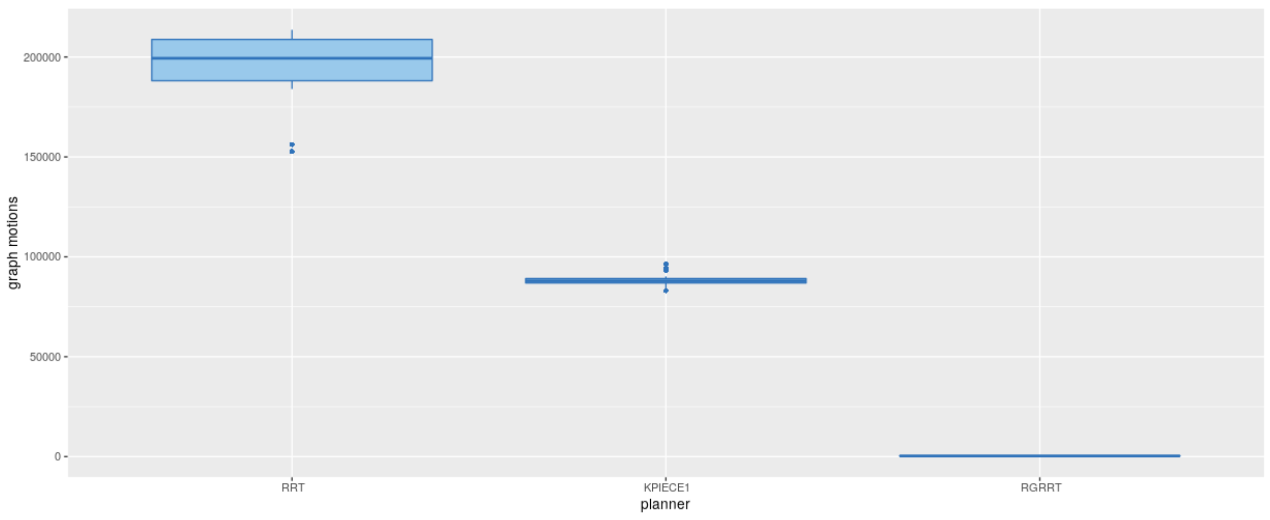

Figure 16: Number of Tree Nodes

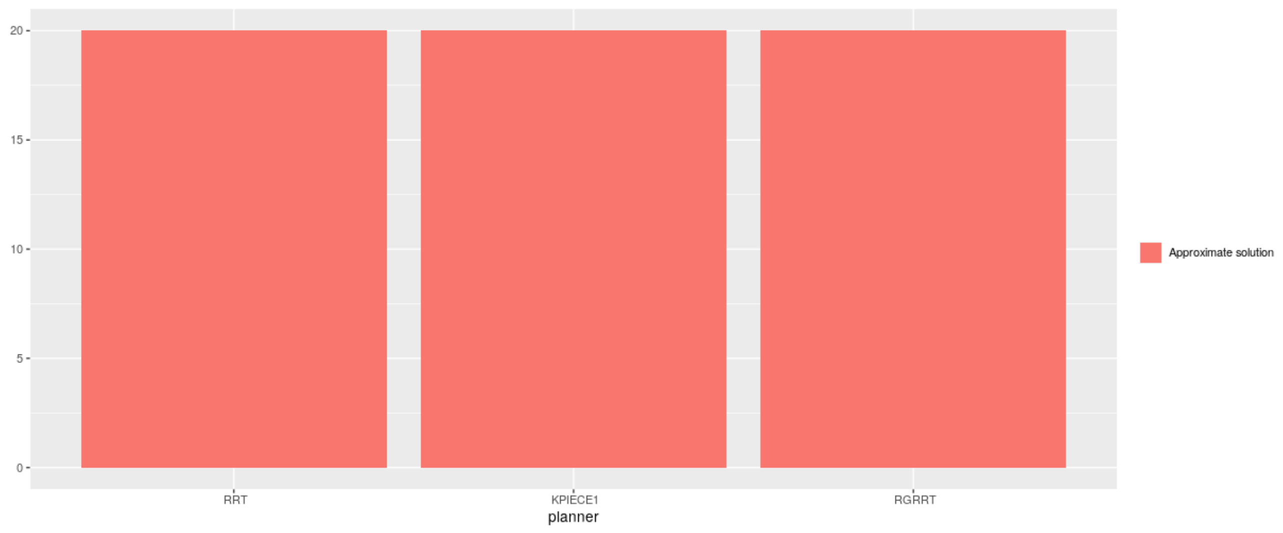

Figure 17: Success Rate

The from above figures the Table 1 is generated which shows comparison between the planners.

Computation time	**RRT**	**KPIECE**	**RGRRT**
Min	Close to KPIECE	Least	Highest
Max	Highest	Close to RRT	Lowest
Median	Highest	Least	Close to RRT
1st Quartile	Close to RGRRT	Least	Highest
3rd Quartile	Highest	Least	Higher than RGRRT
Inter-Quartile Range	Widest	Smallest	Higher than KPIECE
Solution Length	**RRT**	**KPIECE**	**RGRRT**
Min	Significantly high	Highest	Least
Max	Significantly high	Highest	Least
Median	Significantly high	Highest	Least
1st Quartile	Significantly high	Highest	Least
3rd Quartile	Significantly high	Highest	Least
Inter-Quartile Range	Similar to KPIECE	Highest	Least
Tree Nodes	**RRT**	**KPIECE**	**RGRRT**
Min	Highest	Significantly high	Least
Max	Highest	Significantly high	Least
Median	Highest	Significantly high	Least
1st Quartile	Highest	Significantly high	Least
3rd Quartile	Highest	Significantly high	Least
Inter-Quartile Range	Highest	Similar to RGRRT	Least
Success Rate	**RRT**	**KPIECE**	**RGRRT**
Approximate solution	100%	100%	100%

Table 1: Planner Metrics

Summarizing from the above Figures, we conclude that:

- All planners are taking maximum time available to compute the solution and giving approximate solution due to the tolerance being set very low (0.001).
- The median path length for RGRRT is the least.
- The median number of tree nodes is very low for RGRRT.

Car Case:

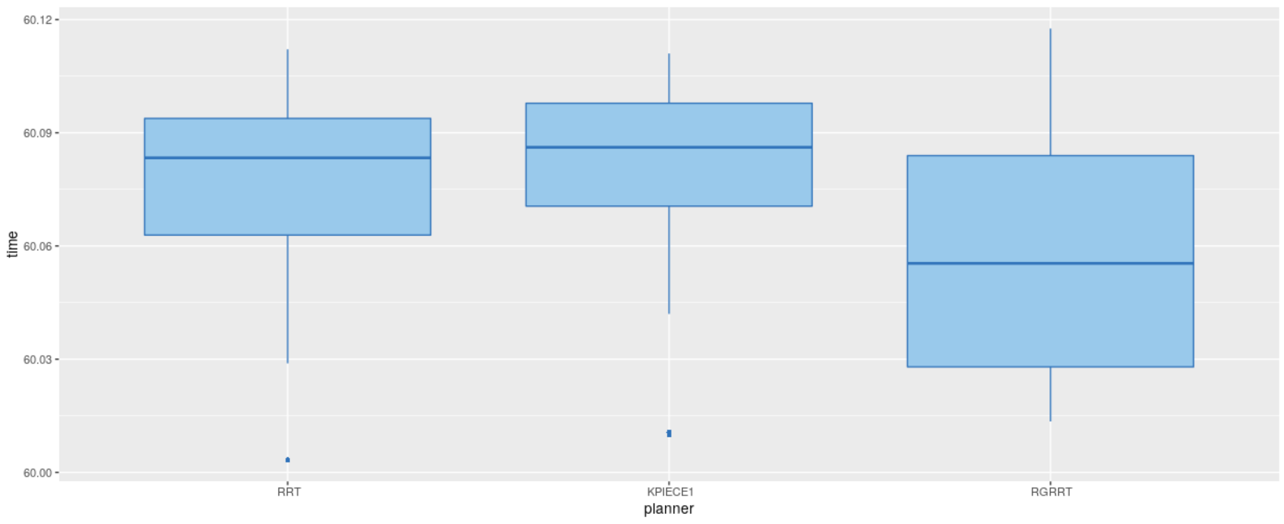

Figure 18: Computation Time

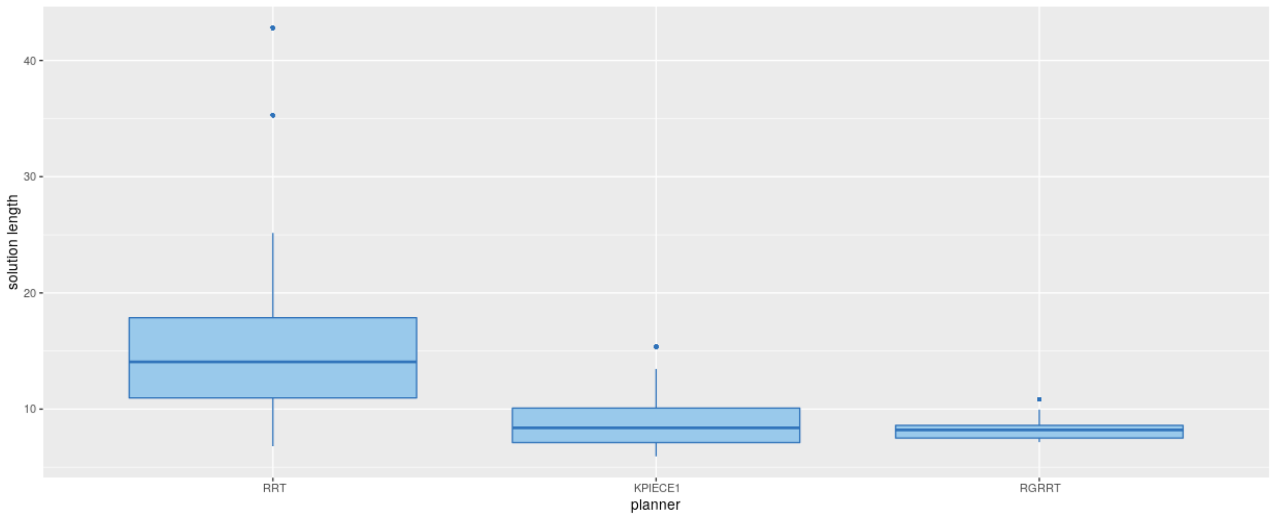

Figure 19: Solution Length

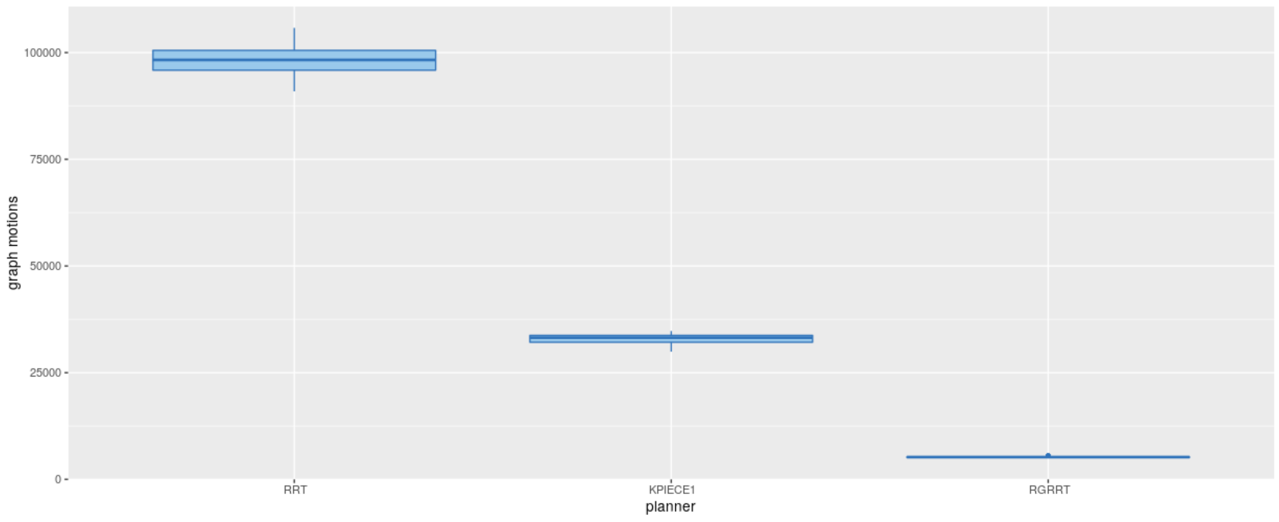

Figure 20: Number of Tree Nodes

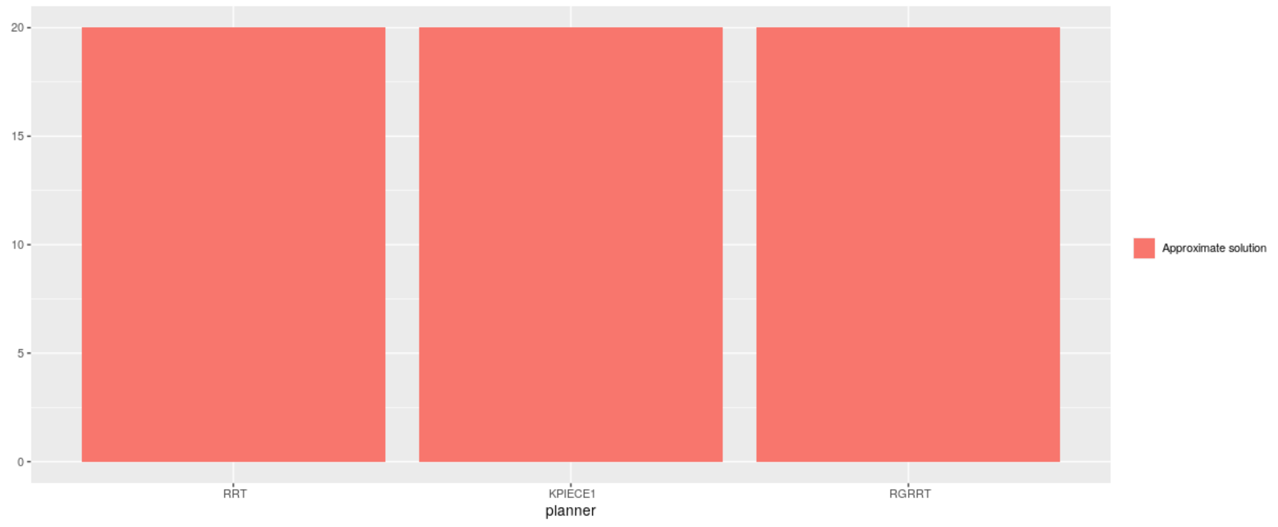

Figure 21: Success Rate

The from above figures the Table 2 is generated which shows comparison between the planners.

Computation time	**RRT**	**KPIECE**	**RGRRT**
Min	Higher than RGRRT	Highest	Least
Max	Close to KPIECE	Least	Highest
Median	Close to KPIECE	Highest	Least
1^{st} Quartile	Close to KPIECE	Highest	Least
3^{rd} Quartile	Close to KPIECE	Highest	Least
Inter-Quartile Range	Close to KPIECE	Smallest	Widest
Solution Length	**RRT**	**KPIECE**	**RGRRT**
Min	Highest	Least	Close to KPIECE
Max	Highest	Higher than RGRRT	Least
Median	Highest	Least	Very close to KPIECE
1^{st} Quartile	Highest	Least	Very close to KPIECE
3^{rd} Quartile	Highest	Close to RGRRT	Least
Inter-Quartile Range	Highest	Higher than RGRRT	Least
Tree Nodes	**RRT**	**KPIECE**	**RGRRT**
Min	Highest	Significantly high	Least
Max	Highest	Significantly high	Least
Median	Highest	Significantly high	Least
1^{st} Quartile	Highest	Significantly high	Least
3^{rd} Quartile	Highest	Significantly high	Least
Inter-Quartile Range	Highest	Higher than RGRRT	Least
Success Rate	**RRT**	**KPIECE**	**RGRRT**
Approximate solution	100%	100%	100%

Table 2: Planner Metrics

Summarizing from the above Figures, we conclude that:

- All planners are taking maximum time available to compute the solution and giving approximate solution due to the tolerance being set very low (0.05).
- The median path length for RGRRT and KPIECE is least and similar to each other. However path length varies more for KPIECE from one iteration to the other.
- The median number of tree nodes is much much less for RGRRT.

Performance Tradeoffs for RGRRT:
Time Period: RGRRT is benchmarked for 3 different Time periods: 0.05, 0.2 and 0.5
The planner is evaluated on three metrics:

- Memory
- Tree nodes
- Solution Length

Car Case:

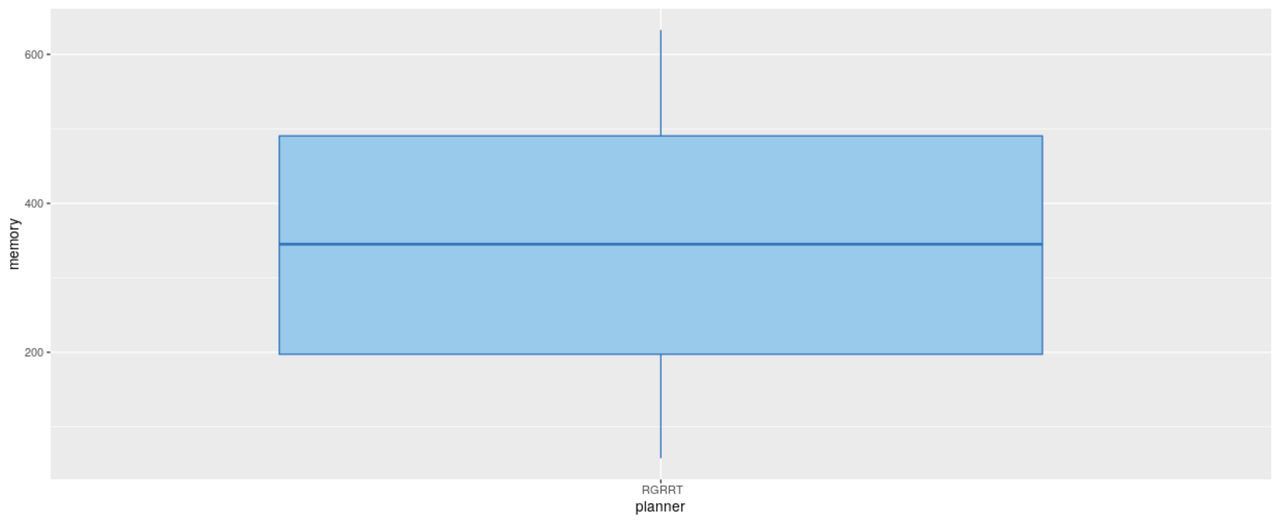

Figure 22: Memory for Time Period = 0.05

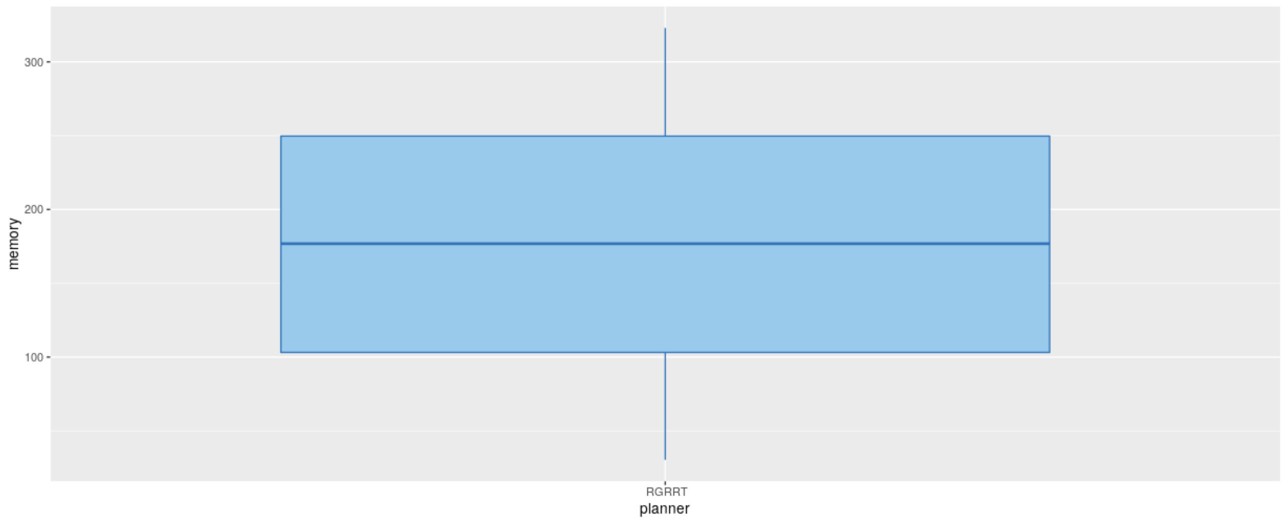

Figure 23: Memory for Time Period = 0.2

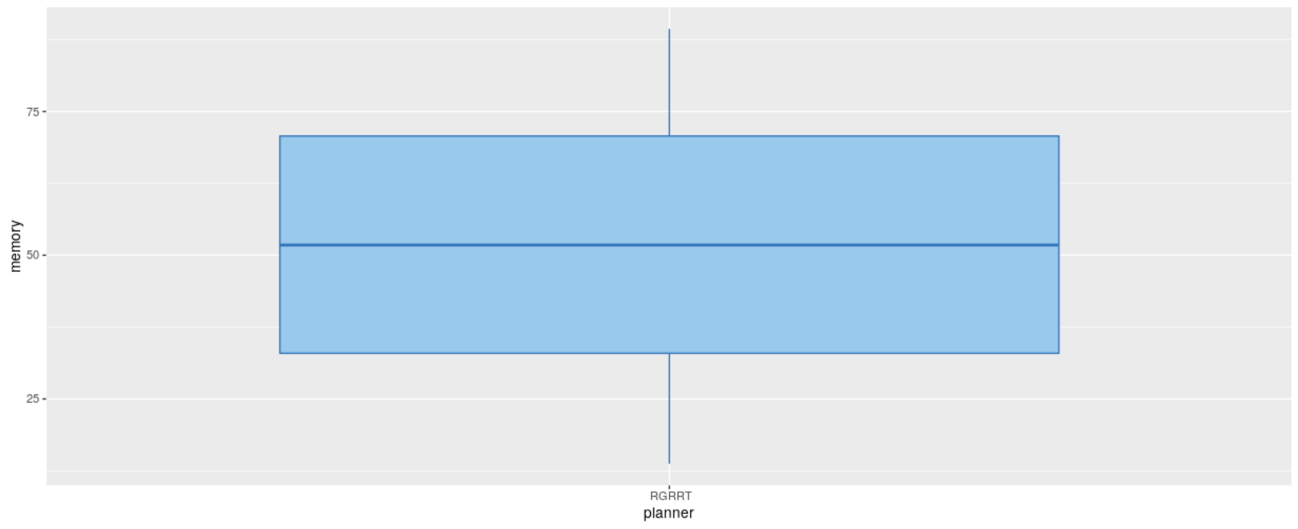

Figure 24: Memory for Time Period = 0.5

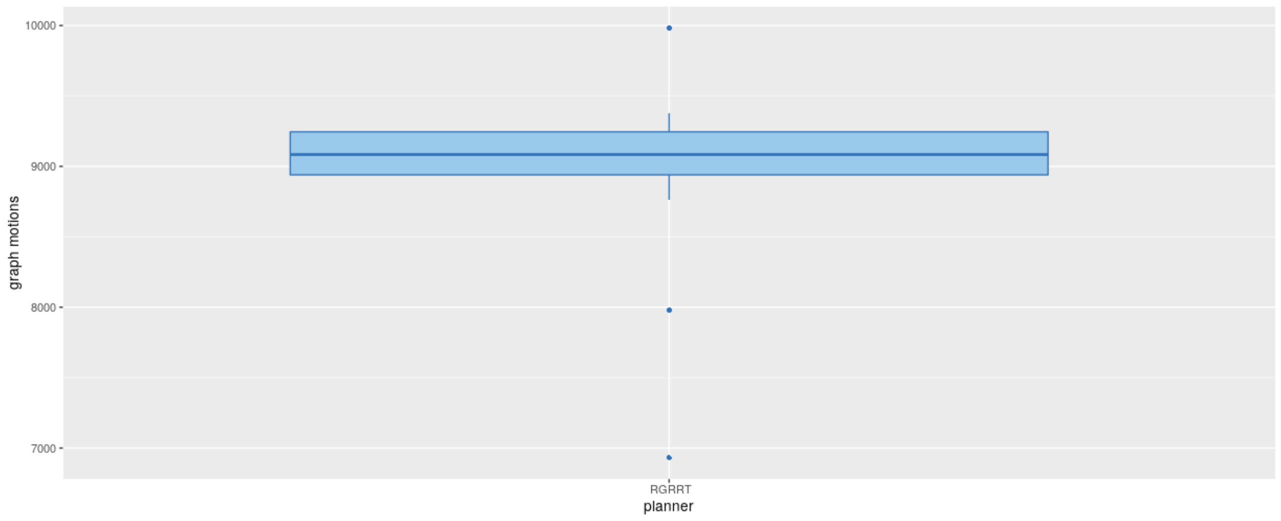

Figure 25: Tree nodes for Time Period = 0.05

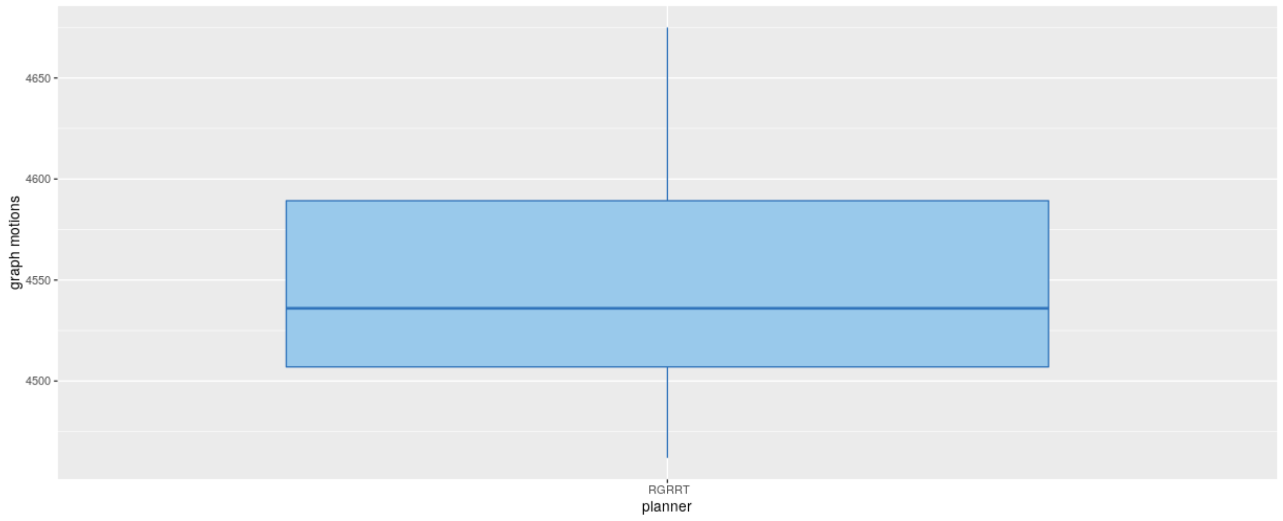

Figure 26: Tree nodes for Time Period = 0.2

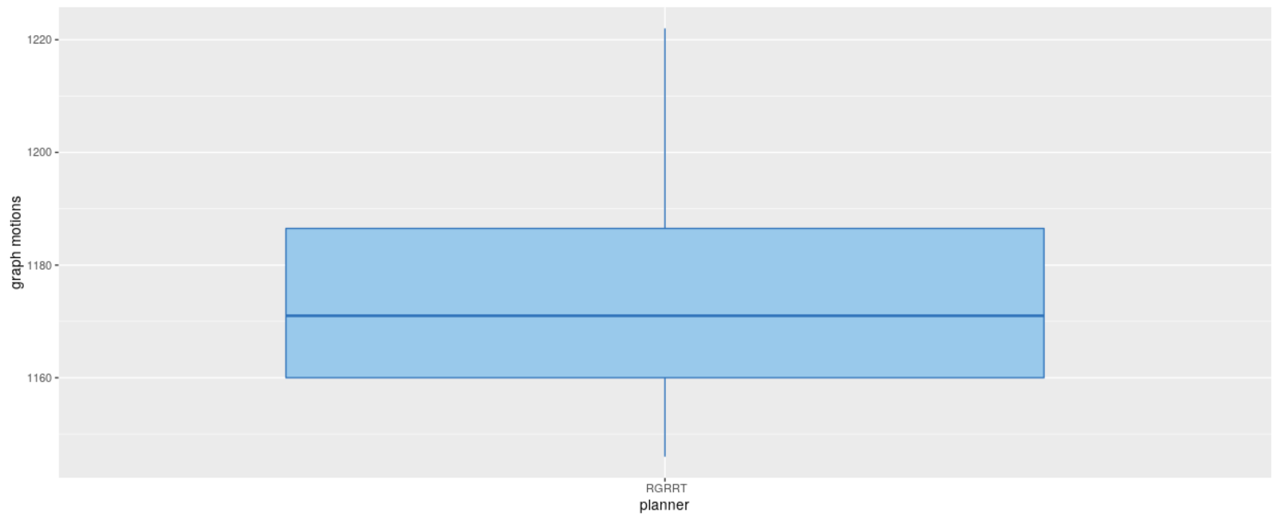

Figure 27: Tree nodes for Time Period = 0.5

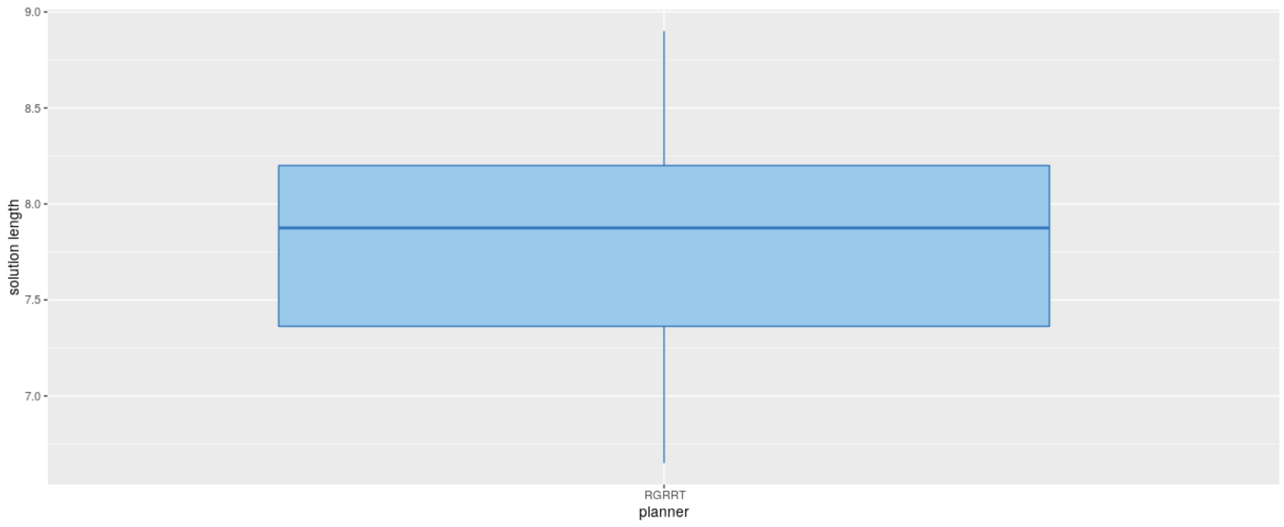

Figure 28: Solution Length for Time Period = 0.05

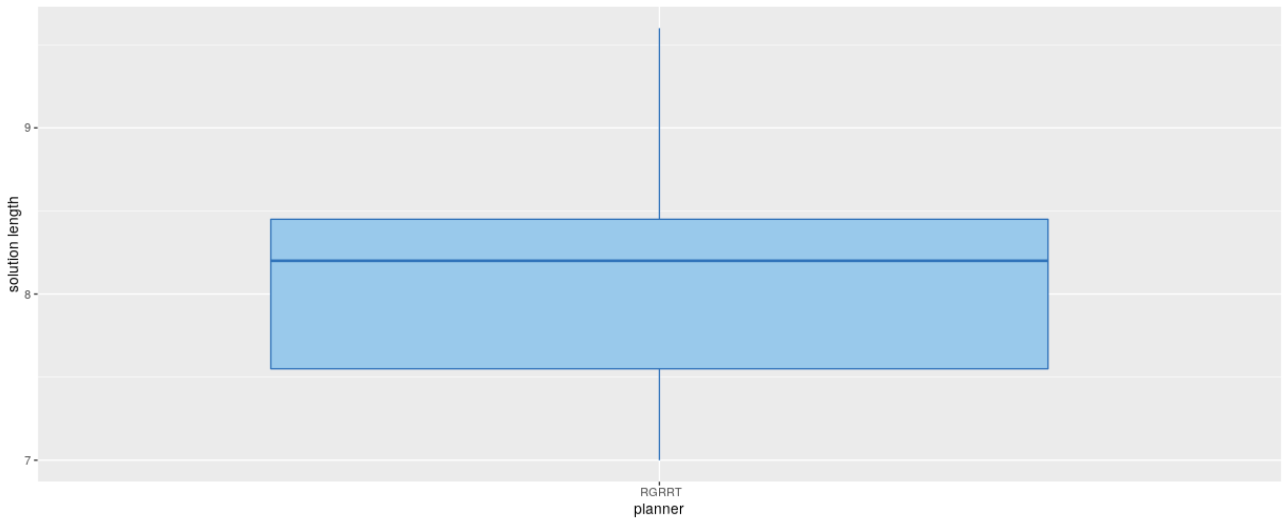

Figure 29: Solution Length for Time Period = 0.2

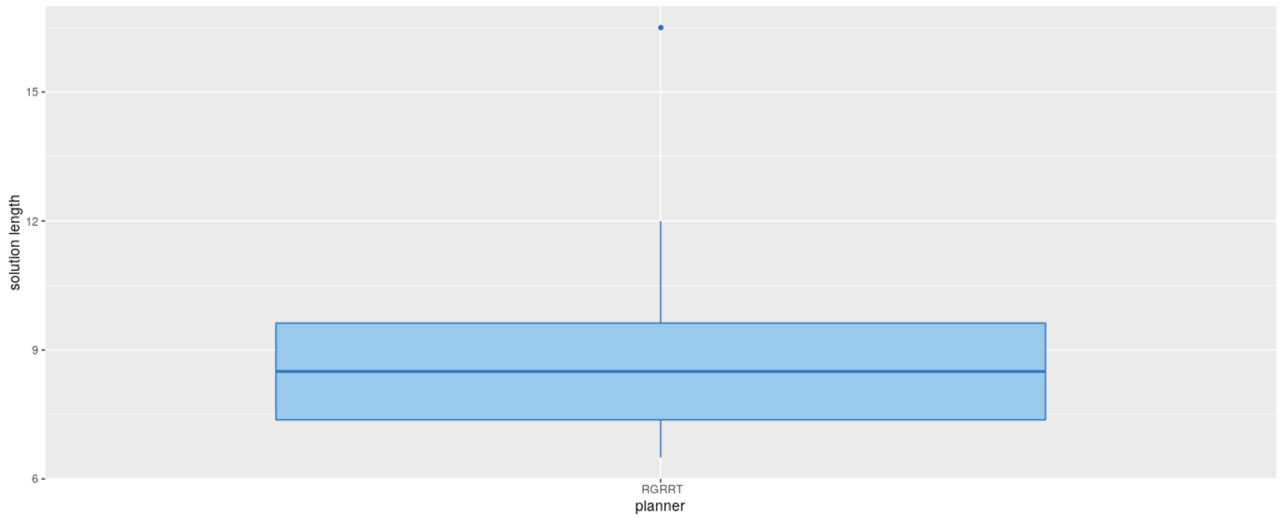

Figure 30: Solution Length for Time Period = 0.5

The from above figures the Table 3 is generated which shows comparison between the planners.

Time Period	Memory	Nodes	Solution Length
0.05	Around 400	Around 9000	Below 8
0.2	Around 200	Around 4550	Above 8
0.5	Around 50	Around 1170	Above 8

Table 3: Planner Metrics

- Increasing the Time period decreases the memory requirement.
- Increasing the Time period decreases the number of tree nodes.
- No effect of Time period on solution length.

Number of Controls: RGRRT is benchmarked for 3 different input Control sizes: 5, 10 and 15 The planner is evaluated on three metrics:

- Memory
- Tree nodes
- Solution Length

Car Case:

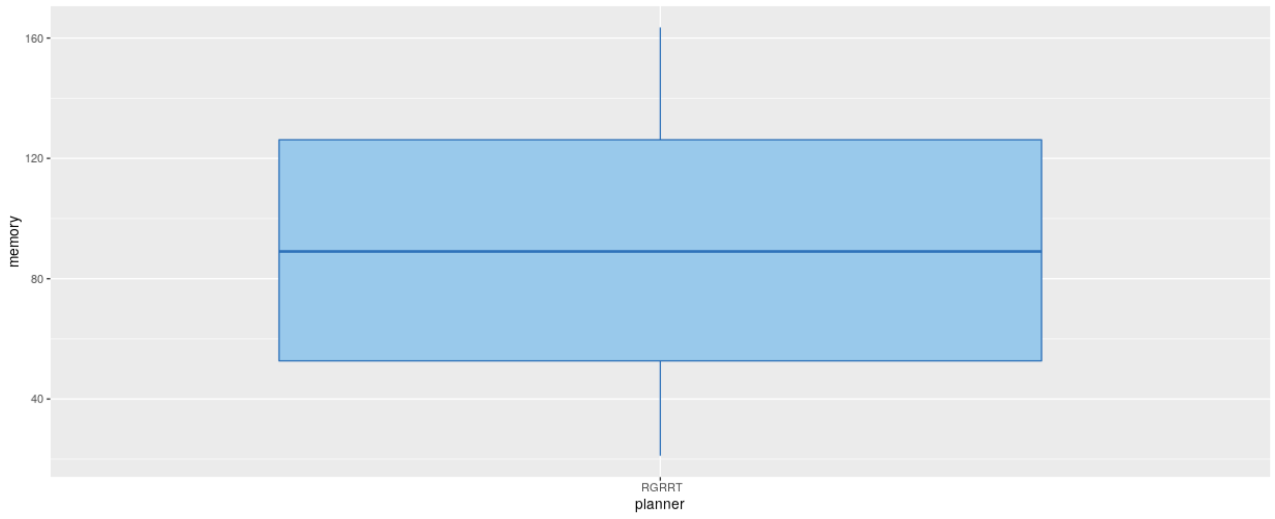

Figure 31: Memory for Control size = 5

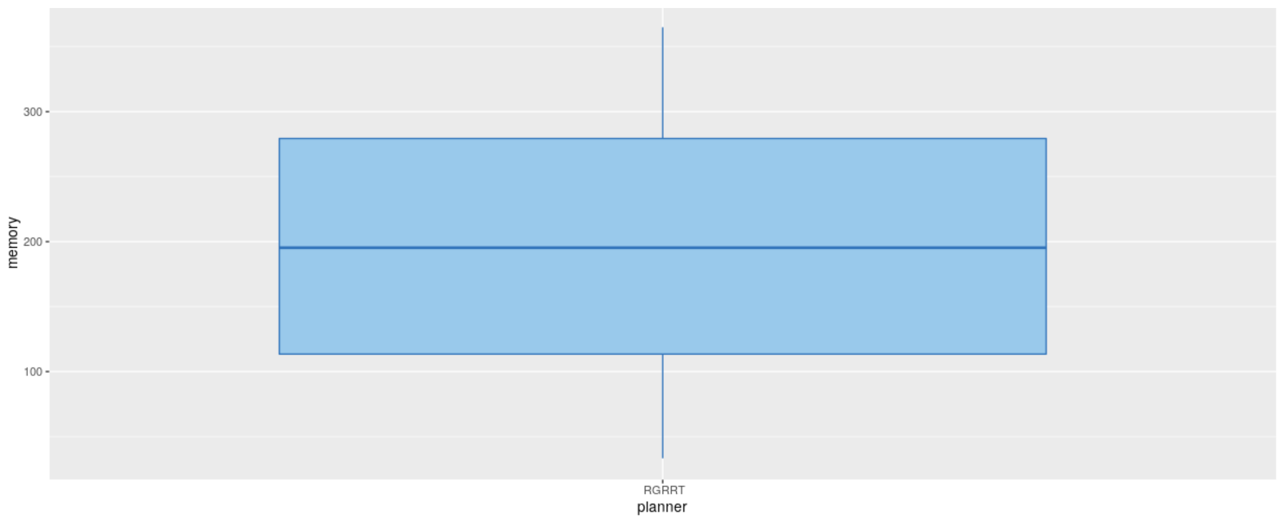

Figure 32: Memory for Control size = 10

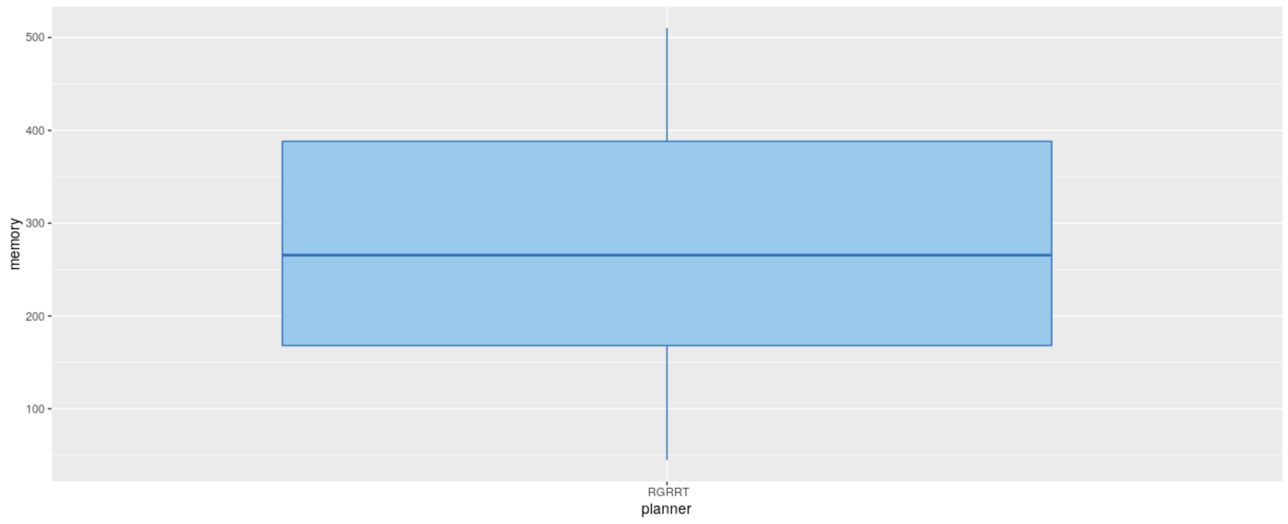

Figure 33: Memory for Control size = 15

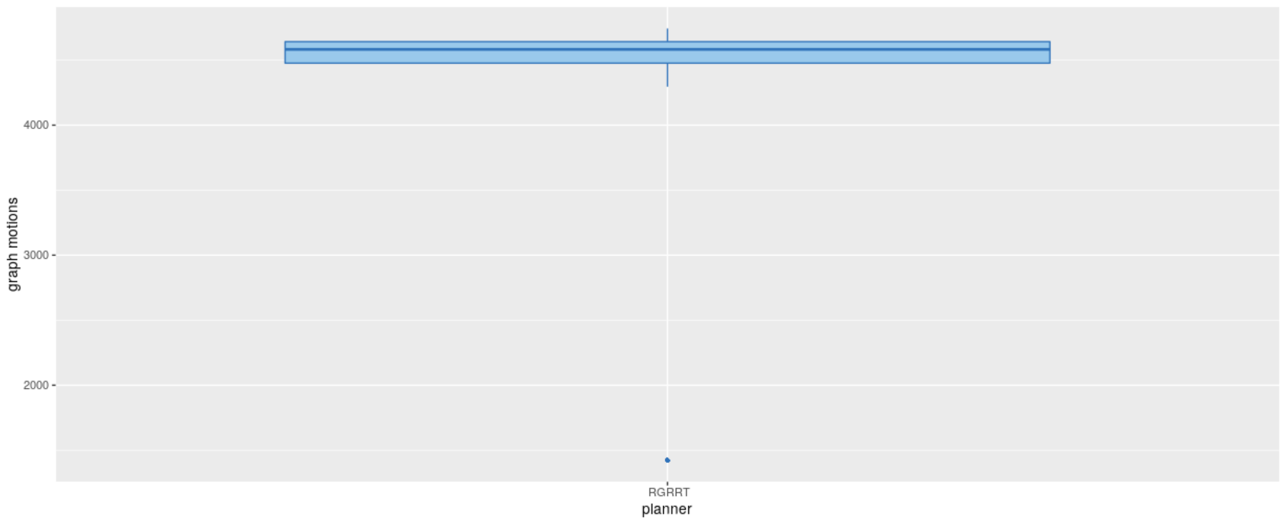

Figure 34: Tree nodes for Control size = 5

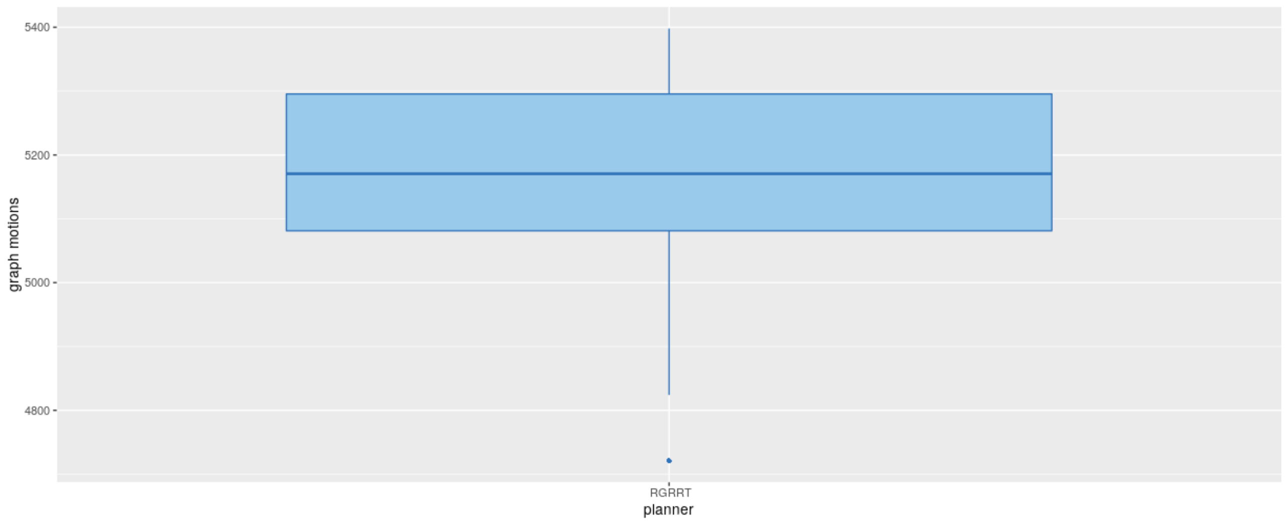

Figure 35: Tree nodes for Control size = 10

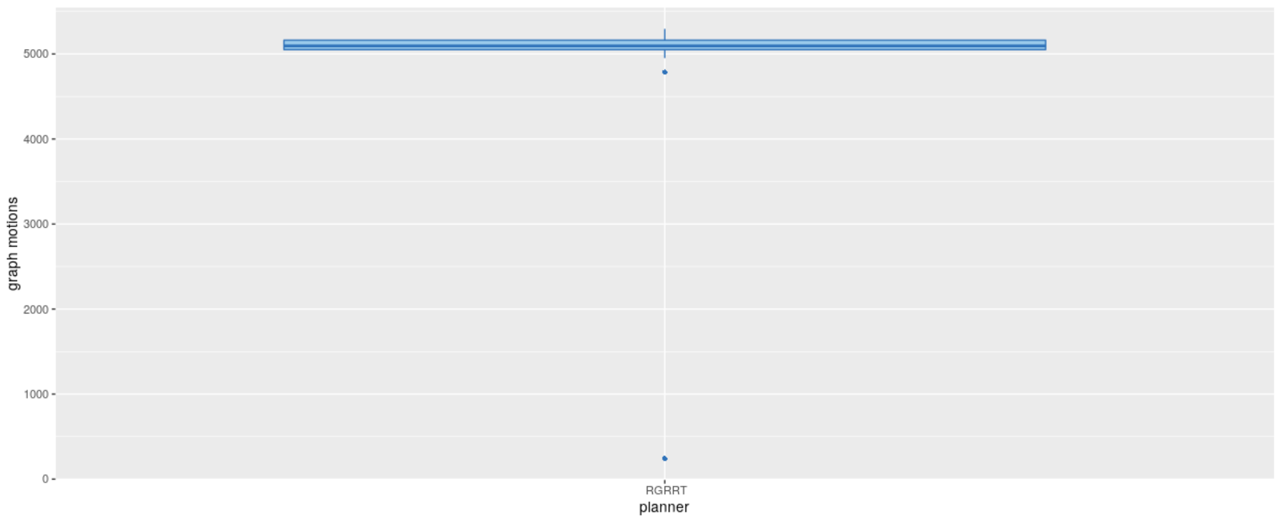

Figure 36: Tree nodes for Control size = 15

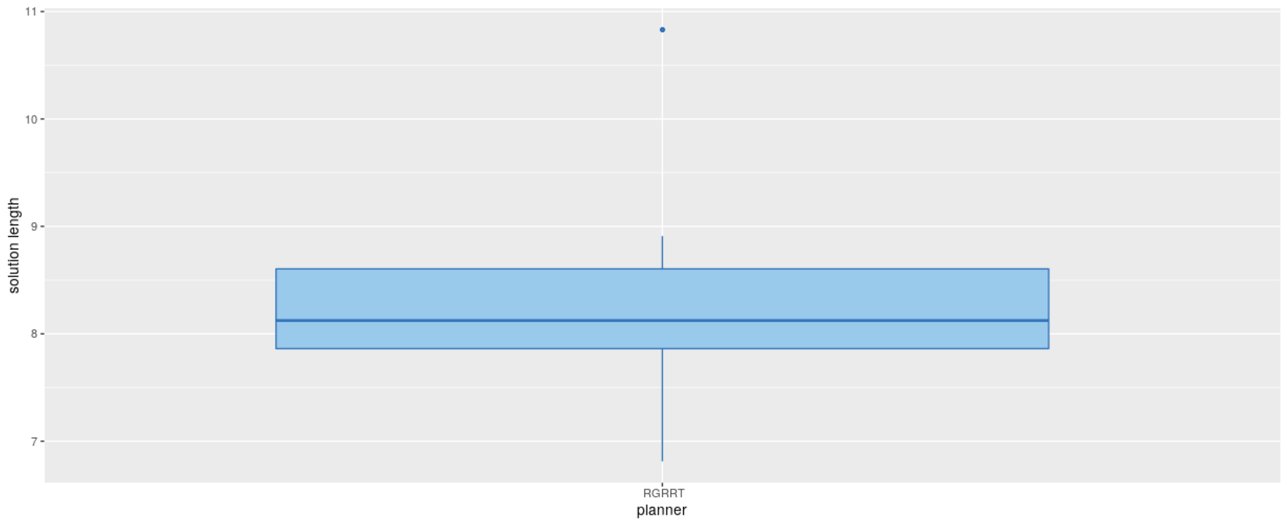

Figure 37: Solution Length for Control size = 5

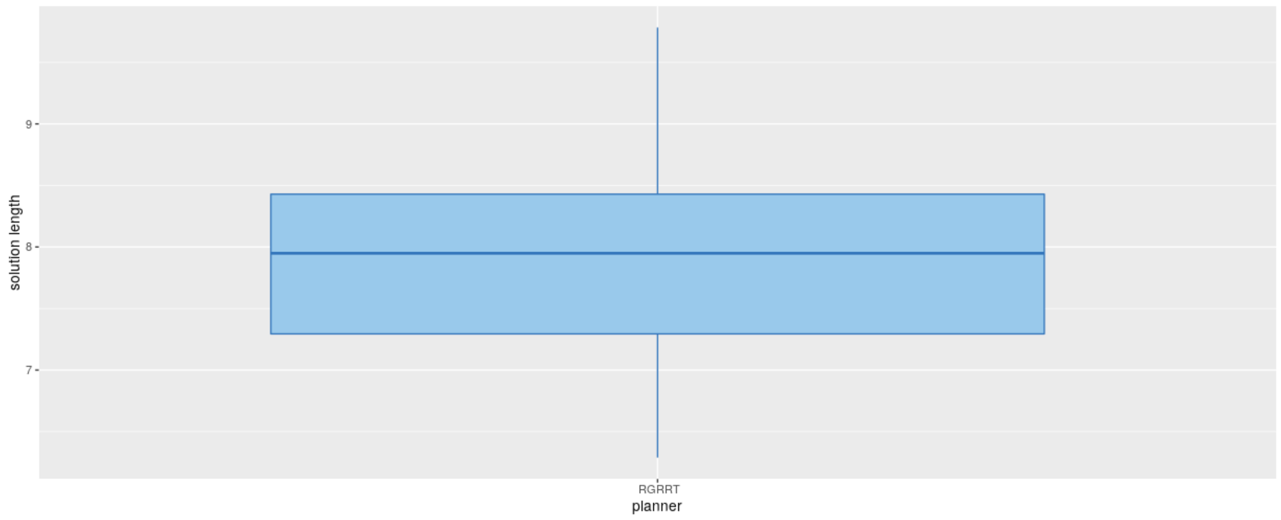

Figure 38: Solution Length for Control size = 10

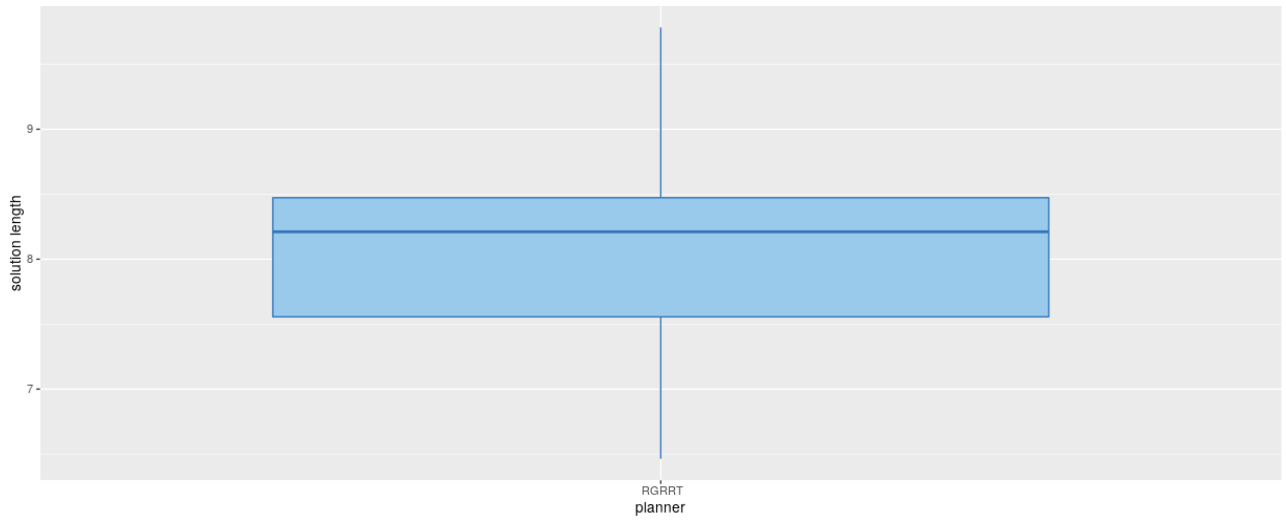

Figure 39: Solution Length for Control size = 15

The from above figures the Table 4 is generated which shows comparison between the planners.

Control Inputs	**Memory**	**Nodes**	**Solution Length**
5	Around 80	Around 4500	Around 8
10	Around 200	Around 5200	Around 8
15	Around 250	Around 5200	Around 8

Table 4: Planner Metrics

- Increasing the Control size increases the memory requirement.
- Increasing the Control size slightly increases the number of tree nodes.
- No effect of Control size on solution length.