Journal of Engineering Science and Technology
Special Issue on INCITEST2019, February (2020) 66 - 76

NEW SAMPLING BASED PLANNING ALGORITHM FOR LOCAL PATH PLANNING FOR AUTONOMOUS VEHICLES

MUHAMMAD ARIA

Electrical Engineering Department, Universitas Komputer Indonesia,
Jl. Dipatiukur 102-116, Bandung 40132, Indonesia
E-mail: muhammad.aria@email.unikom.ac.id

Abstract

The purpose of this paper was to design a new sampling-based planning algorithm based on the integration of Rapidly-exploring Random Trees (RRT) and Probabilistic Roadmap Method (PRM) algorithms that could be used to build local path planning for autonomous vehicles. The RRT algorithm had the advantage of low computational time but provided suboptimal solutions, while the PRM algorithm had the advantage of providing asymptotically optimal solutions, but high computational time. Then the proposed algorithm combined the advantages of the two algorithms so that they had low computational time and provided optimal asymptotic solutions. The process was carried out by running the RRT algorithm several times to obtain several alternative suboptimal paths. Furthermore, the optimal solution was built using these suboptimal pathways, using the PRM-Djikstra path optimization algorithm. After the algorithm produced the final path, smoothing techniques using the Reed Sheep Planner algorithm employed to produce a smooth curved path. This study also compared the effect of using several variations of the RRT algorithm. With, tested algorithm in motion-planning problems of the non-holonomic vehicle. The results showed that our algorithm could produce higher quality output paths because the algorithm generated several sub-optimal paths and then combines them.

Keywords: Autonomous vehicles, Djikstra, Local path planning, Probabilistic roadmap method, Rapidly-exploring random trees.

1. Introduction

The autonomous vehicle is a vehicle system that can travel to a predetermined destination without the need for involvement of a manual driver by humans [1]. The basic workflow of an autonomous vehicle system is as follows [2]. First of all, the system builds a path from the current position to the destination position (known as global path planning). Global paths consist of waypoints that must be passed by vehicles. Global path planning usually uses the Djikstra or A* algorithm [3]. Next, the system access the 3D cloud map of the environment. Then the system uses its sensors to detect objects around it. Next, the system predicts the movement direction of each object. Finally, the system decides the direction of vehicle movement and speed that must be carried to reach the next waypoint (also known as local path planning). This paper focused on the discussion of developing local path planning algorithms.

There were several methods used to find the best path. Search algorithms such as AD*found optimal solutions in dynamic graphs. But, the use of graph search methods involved discretizing workspaces, and their performance degraded in high dimensions [4]. The work in produced state lattices using primitive motion and combines them with graph search algorithms. But still, suffer from unwanted discretization [5]. Previous research illustrates that the strategic technique for creating a service operation that is free from errors is by using Failure Mode and Effect Analysis (FMEA), but there are some things that need to be reviewed in more depth from this system, namely Risk Priority Number (RPN), reprioritization and flexibility when applying it. Previous research illustrates that the strategic technique for creating a service operation that is free from errors is by using Failure Mode and Effect Analysis (FMEA), but there are some things that need to be reviewed in more depth from this system, namely Risk Priority Number (RPN), reprioritization and flexibility when applying it [6]. While other studies say that each approach used must be able to balance between economic needs and existing availability, which will have an impact on increasing the burden when making decisions by related parties, so it is very important to pay attention to the formation of models that will be oriented towards the availability of resources power [7].

The Ant Colony Optimization Methodapplied in robot path planning, but it suffered from falling to the local minimum and performed poorly in narrow areas [8] Sensor-based reactive planning methods proposed but it could be considered as global planners [9, 10]. Control-based methods required accurate model formulation for robots and the environment which could be a rather daunting task [11]. Sampling-based planning (SBP) had advantages in terms of providing fast solutions for difficult problems [12]. The most commonly SBP used algorithms perhaps are Probabilistic Roadmap Method (PRM) and Rapidly-exploring Random Trees (RRT) [13, 14]. The PRM algorithm has the advantage of providing asymptotically optimal solutions but had high computational time ($O(n^2)$), while the RRT algorithm has the advantage of low computational time ($O(n \log n)$) but provides suboptimal solutions [15]. The RRT algorithm was then developed into RRT* [16] which provided an asymptotically optimal solution [17]. However, the RRT* computing time was still high. Currently, there are many development algorithms from this RRT*. Akgun and Stilman introduced the B-RRT* algorithm This method improved path optimization by using a two-way search of the RRT* algorithm. Kumar et al proposed the RRT-biased algorithm [17, 18]. Gammell et al proposed the Informed RRT* algorithm which offered a new way of sampling the RRT algorithm. The sampling process was carried out in the ellipse area

surrounding the initial node with the final node [19]. Nasir et al introduced the RRT* -Smart algorithm to accelerate the rate of convergence. There were two features used in RRT* -smart, namely intelligent sampling and path optimization. To the author's knowledge, there were no previous studies examined the effect of combining the PRM algorithm with several variations of the RRT algorithm to produce non-holonomic vehicle path planning algorithms that had low computational time and provided asymptotic optimal solutions [20]

The contribution of this paper was to propose a new SBP algorithm based on the integration of PRM and RRT algorithms and could be used to build local path planning for autonomous vehicles. The proposed algorithm combined the advantages of PRM and RRT algorithms so that they had low computational time and provided optimal asymptotic solutions. This process was done by first running the RRT algorithm several times to obtain several suboptimal paths. Then, the optimal solution was built based on the suboptimal path using the PRM-Djikstra path optimization algorithm. To obtain a smooth curved path, then after the algorithm produced the final path, continued the application of smoothing techniques using the Reed Sheep Planner algorithm. The effects of using several variations of the RRT algorithm were also compared in this study. The proposed algorithm was then tested in the problem of non-holonomic vehicle motion planning.

2. Method

In this research explained the basic idea of the integration of the RRT algorithm and the PRM algorithm. The basic PRM algorithm consisted of two main processes. The first process was generating sample points and roadmap construction as shown by algorithm 1 in Fig. 1. The second process was generating the optimal path shown by algorithm 2 in Fig. 2.

Algorithm 1 : Generate Sample Point and Roadmap Construction Algorithm
Input :
n : number of nodes to put
k : number of closest neighbors to examine for each configuration
Output :
A roadmap $T = (V, E)$

```
1.  V ← ∅
2.  E ← ∅
3.  while |V| < n do
4.      repeat
5.          q ← a random configuration in Q
6.      Until q is collision-free
7.      V ← V ∪ {q}
8.  end while
9.  for all q ∈ V do
10.     N_q ← the k closest negihbors of q chosen from V according to dist
11.     for all q' ∈ N_q do
12.         if (q, q') is collision-free then
13.             E ← E ∪ {(q, q')}
14.         end if
15.     end for
16. end
```

Fig. 1. PRM algorithm to generate sample point and roadmap construction.

Algorithm 2 : Solve Query Algorithm
Input :
q_{init} : the initial configuration
q_{goal} : the goal configuration
k : the number of closest neighbors to examine for each configuration
$T = (V, E)$: the roadmap computed by algorithm 1
Output :
A path from q_{init} to q_{goal} or failure

1. $N_{q_{init}} \leftarrow$ the k closest neighbors of q_{init} from V according to $dist$
2. $N_{q_{goal}} \leftarrow$ the k closest neighbors of q_{goal} from V according to $dist$
3. $V \leftarrow \{q_{init}\} \cup \{q_{goal}\} \cup V$
4. set q' to be the closest neighbor of q_{init} in $N_{q_{init}}$
5. **repeat**
6. **if** $\Delta(q_{init}, q') \neq$ NIL **then**
7. $E \leftarrow (q_{init}, q') \cup E$
8. **else**
9. set q' to be the next closest neighbor of q_{init} in $N_{q_{init}}$
10. **end if**
11. **until** a connection was successful or the set $N_{q_{init}}$ is empty
12. set q' to be the closest neighbor of q_{goal} in $N_{q_{goal}}$
13. **repeat**
14. **if** $\Delta(q_{goal}, q') \neq$ NIL **then**
15. $E \leftarrow (q_{goal}, q') \cup E$
16. **else**
17. set q' to be the next closest neighbor of q_{goal} in $N_{q_{goal}}$
18. **end if**
19. **until** a connection was successful or the set $N_{q_{goal}}$ is empty
20. $P \leftarrow shortest\ path(q_{init}, q_{goal}, T)$ using Djikstra Algorithm
21. **if** ***P*** is not empty **then**
22. **return** *P*
23. **else**
24. **return** failure
25. **end if**

Fig. 2. PRM algorithm to generate the final path

The results of the sample point generation process are shown in Fig. 3(a). The sample point distributed randomly over a large area (according to algorithm 1 line 5). This considered a weakness because planners had a high probability of taking samples from large areas (did not contribute effectively to the preparation of the final path). Fig. 3(a) shows a sample point that away from the goal node but must still be checked by the algorithm. For the sample point to spread only along the path to the goal node, the sample point was generated based on a combination of several suboptimal path solutions. The illustrations are shown in Figs. 3(b)-(f). Figures 3(b)-(d) show the suboptimal path solution produced sequentially, Fig. 3(e) is the distribution of sample points generated based on the combination of three suboptimal paths, and Fig. 3(f) is the optimal path result resulting from the combination of several optimal sub-paths from path 3(b)-(d). The RRT algorithm that in algorithm 3 is in Fig. 4. The basic idea was to run the algorithm 3 several times to get sample points spreading only on the path to the goal point. Then, algorithm 2 was applied to get the optimal final path from the distribution of sample

points that made. Thus, the input algorithm 2 in the form of a roadmap T was generated by the output of algorithm 3 (not by algorithm 1).

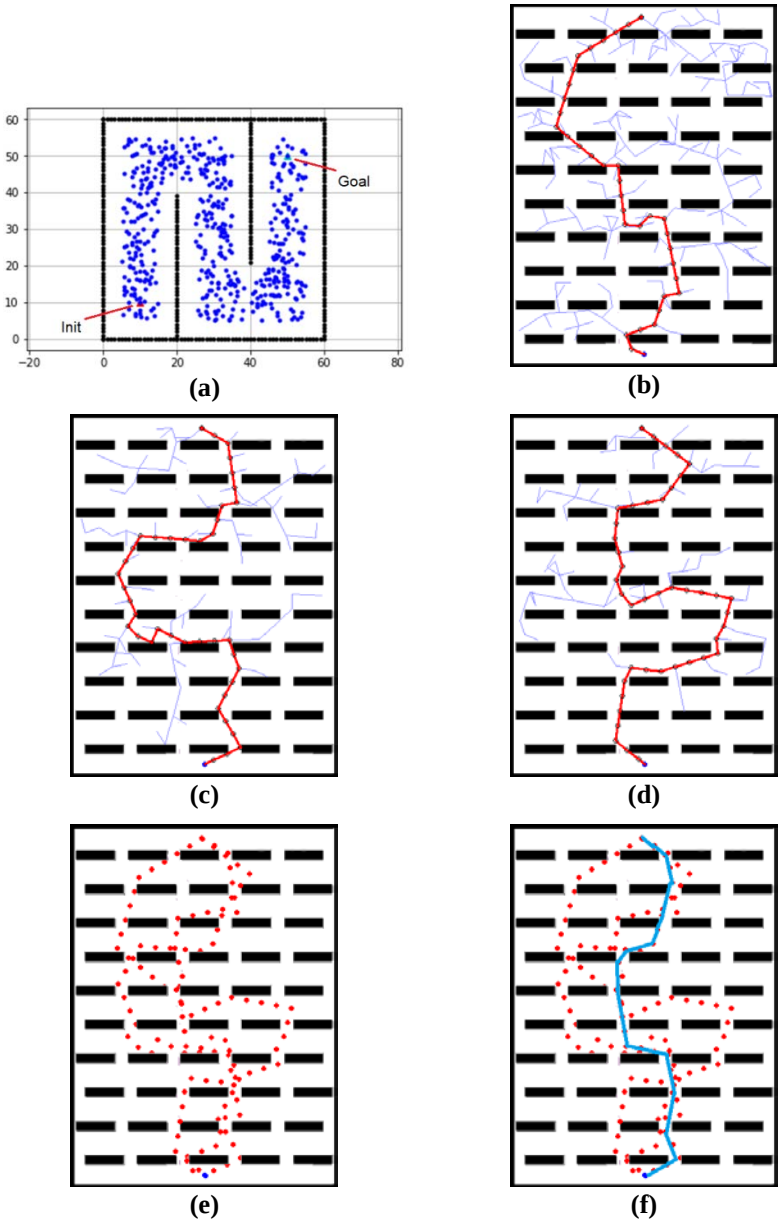


Fig. 3. The basic idea of the integration of the RRT and PRM algorithm: (a) Example sample points generated by PRM; (b) First suboptimal path solution produced by RRT; (c) Second suboptimal path solution produced by RRT; (d) Third suboptimal path solution produced by RRT; (e) The distribution of sample points is generated based on the combination of three suboptimal paths generated by the RRT; (f) Example path solutions generated by the proposed algorithm use a sample point by the RRT.

Algorithm 3 : $T = (V, E) \leftarrow RRT(q_{init})$

1. $T \leftarrow InitializeTree()$
2. $T \leftarrow InsertNode(\emptyset, q_{init}, T)$
3. **for** $k \leftarrow 1$ **to** N **do**
4. $\quad q_{rand} \leftarrow RandomSample(k)$
5. $\quad q_{nearest} \leftarrow NearestNeighbor(q_{rand}, Q_{near}, T)$
6. $\quad q_{new} \leftarrow Steer\ (q_{nearest}, q_{rand}, \Delta q)$
7. $\quad$ **if** $Obstaclefree(q_{new}, q_{nearest})$ **then**
8. $\quad\quad T \leftarrow InsertNode(q_{min,} q_{new}, T)$
9. **End**

Fig. 4. RRT algorithm

3. Results and Discussion

First, this study compared the performance of the proposed algorithm against the performance of the RRT and PRM algorithms. Then, this study compared the effects of using several variations of the RRT algorithm to produce sample points for the PRM algorithm. Some variations of the RRT used in this study were the RRT-Biased, RRT, RRT* without rewire operations and Bi-RRT [14-16]. For this experiment, a simulation program had been made. For the test case, it was taken from several benchmark cases from RRT research papers. The benchmark cases used in this study are shown in Fig. 5.

RRT * algorithm had an advantage over RRT because it had chooseparent and rewiring operations. The RRT-biased algorithm had the characteristic of not just making a new node one time step towards a random node. But the new node made several steps until the random node reached or obstacles block the path. The Bi-RRT-biased algorithm used a bidirectional version of the search for the RRT-biased algorithm. One search tree started from the initial node, and another search tree started from the destination node.

Path cost comparison between RRT, PRM and the proposed algorithm is shown in Fig. 6. Computation time comparison between the three algorithms is shown in Fig. 7. It appeared that the PRM algorithm could produce a better cost path than the RRT algorithm, but required higher computational time. This was the same as reported by other reports [19]. And, the RRT and PRM integration algorithm produced the smallest cost path, even though it had the highest computational time. By repeating the RRT algorithm several times, several alternative paths to the destination node obtained. In each of these alternative paths, it was possible to find sub-paths that had better cost paths than sub-paths on other alternative paths. By combining several sub-paths that had a minimal cost path, the proposed algorithm could build a better path. Because the proposed algorithm repeated the RRT algorithm several times, it could be seen that the computational time of the proposed algorithm was never smaller than the computational time of the RRT algorithm.

The next discussion compared the effects of using various variations of the RRT algorithm in generating sample points for the PRM algorithm. Path cost comparison between RRT-Biased, RRT*, RRT* algorithms without *rewire* function and Bi-RRT-Biased is shown in Fig. 8. Computation time comparison between the five algorithms is shown in Fig. 9. It can be seen that the RRT-Biased algorithm produced routes with an average distance shorter than RRT* with the

computational time that was relatively equal with RRT*. Although the Bi-RRT-Biased algorithm always guarantees the smallest distance, the computational time of the Bi-RRT-Biased algorithm could be very large. Also shown in the figure was that the cost path of the RRT * algorithm was smaller than the RRT. This was consistent with what was reported by other researchers [21]. And the Bi-RRT algorithm had a smaller cost path than RRT. This is consistent with what was reported in literature [22].

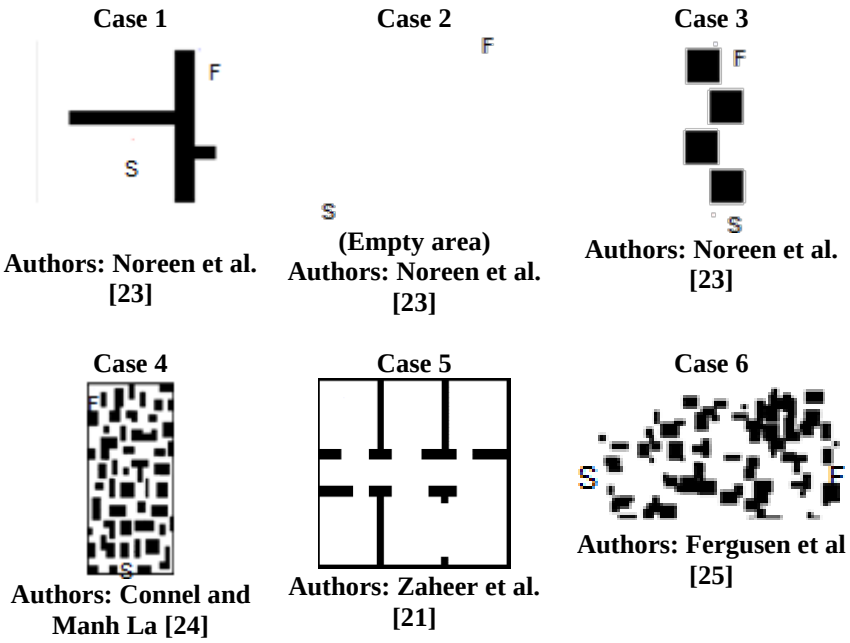


Fig. 5. The benchmark case used in simulation testing.

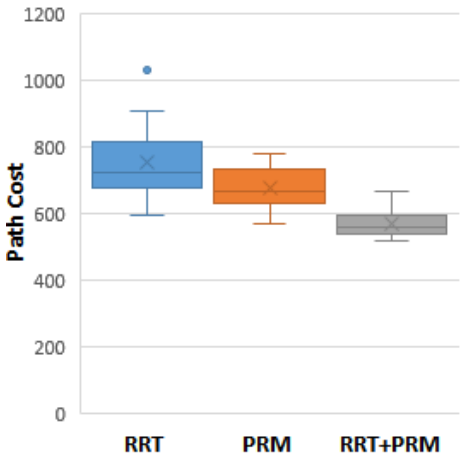


Fig. 6. Cost comparison of the path produced by RRT, PRM and integrated RRT and PRM.

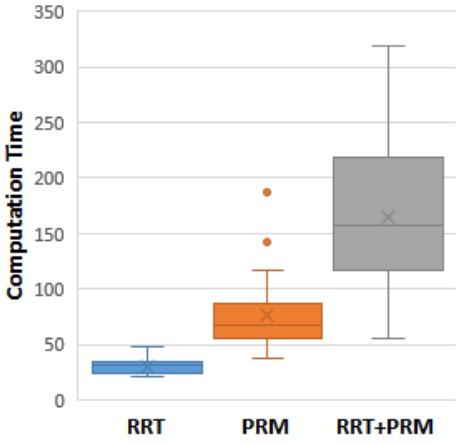


Fig. 7. Computation time comparison between RRT, PRM and integrated RRT and PRM.

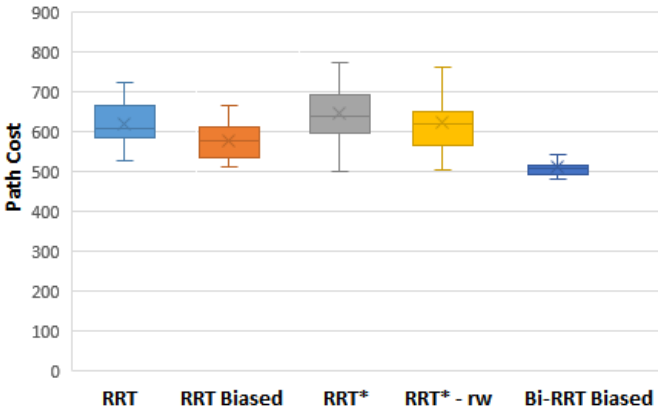


Fig. 8. Cost comparison of the path produced by each algorithm.

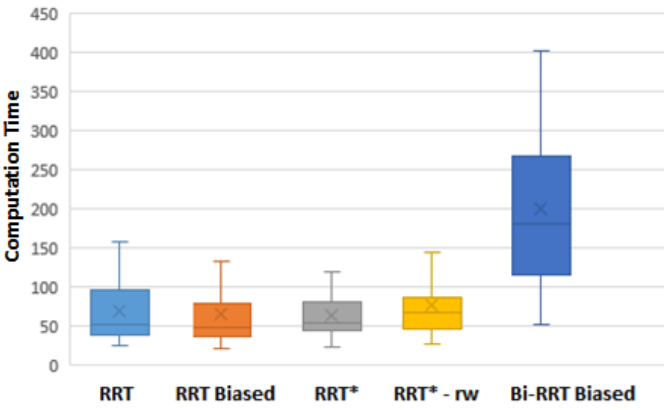
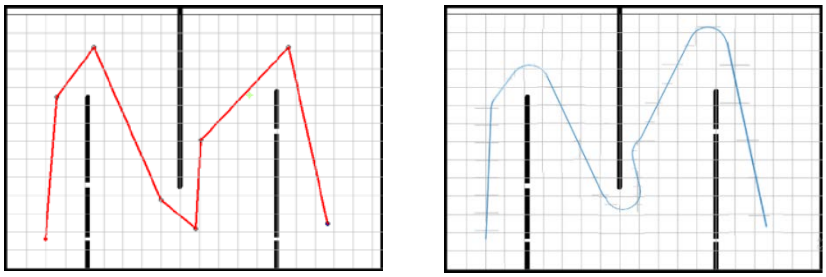


Fig. 9. Computation time comparison of each algorithm.

After the final path was obtained (but it still had sharp curves), the reed sheep path planning algorithm used at each edge. The reed sheep path planning algorithm adjusted the vehicle to be among the three types of maneuvers, namely moving straight, turning left fully (maximum steering angle to the left), and turning rightfully.

An illustration of the process can be shown in Figs. 10(a) and (b). Figure 10(a) is the final path produced by the proposed algorithm. It showed that the path still contained sharp curves. Figure 10(b) is the path produced by the reed sheep path planning algorithm. The reed sheep path planning algorithm formed a path based on two main inputs, namely the starting point (x_s, y_s, θ_s) and the destination point (x_g, y_g, θ_g). Although this data confimed good analysis. Other analyses are still required [26, 27], especially for understanding type of vehicle and aerodynamic.



(a) (b)

Fig. 10. (a) the path produced by the proposed algorithm, (b) the path produced by the reed sheep path planning algorithm.

4. Conclusion

This paper proposed a new SBP algorithm based on the integration of RRT and PRM algorithms used to build local path planning for autonomous vehicles. Using Reed Sheep Planner algorithm, the final path that still had sharp curves is smoothed and formed into a curved line that followed by non-holonomic vehicles. The proposed algorithm produced higher quality output paths than a single RRT or PRM output because the algorithm generated several sub-optimal paths and then combined them. It also compared the effects of using several variations of the RRT algorithm to generate sample points for the PRM algorithm. Some variations of RRT that used in this study were RRT with biasing, RRT*, RRT* without rewire operations and Bi-RRT Biased. The RRT-biased algorithm produced routes with a shorter average distance than RRT* with computational time relatively equal with RRT*. Although the Bi-RRT Biased algorithm always guaranteed the smallest distance, the computational time of the Bi-RRT Biased algorithm was very large.

References

1. Zhao, J.; Liang, B.; and Chen, Q. (2018) The key technology toward the self-driving car. *International Journal of Intelligent Unmanned Systems*, 6(1), 2-20.
2. Garcia, O.; Lemos, R.; and Ferreira, J.V. (2016) Local and global path generation for autonomous vehicles using splines. *Ingeniería*, 21(2), 1-1.
3. Liu, S.; Li, L.; Tang, J.; Wu, S.; and Gaudiot, J.L. (2017) Creating autonomous vehicle system. *Synthesis Lectures on Computer Science*, 6(1), 186.

4. Likhachev, M.; Ferguson, D.; Gordon, G.; Stentz, A.; and Thrun, S. (2008) Anytime search in dynamic graphs. *Artificial Intelligence*, 172(14), 1613-1643.
5. Pivtoraiko, M.; Knepper, R.A.; and Kelly, A. (2009) Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3), 308-333.
6. Sutrisno, A.; and Lee, T.J. (2011). Service reliability assessment using failure mode and effect analysis (FMEA): Survey and opportunity roadmap. *International Journal of Engineering, Science and Technology*, 3(7), 25-38.
7. Wahid, A.S.B.A.; Ahmad, M.Z.; Ahmad, K.A.B.; Taylor, J.P.; Abdullah, A.B.; Al-Shafiq, A.A.B.; and Kitagawa, K. (2018). Demystifying ship operational availability–An alternative approach for the maintenance of naval vessels. *Journal of Engineering Science and Technology*, 13(12), 4326-4346.
8. Garcia, M.P.; Montiel, O.; Castillo, O.; Sepúlveda, R.; and Melin, P. (2009) Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. *Applied Soft Computing*, 9(3), 1102-1110.
9. Belkhouche, F.; and Bendjilali, B. (2011) Reactive path planning for 3-D autonomous vehicles. *IEEE Transactions on Control Systems Technology*, 20(1), 249-256.
10. Manup, B.; and Raja, P. (2016). Collision-avoidance for mobile robots using region of certainty: A predictive approach. *Journal of Engineering Science and Technology*, 11(1), 18-28.
11. Werling, M.; Kammel, S.; Ziegler, J.; and Gröll, L. (2012) Optimal trajectories for time-critical street scenarios using discretized terminal manifolds. *The International Journal of Robotics Research*, 31(3), 346-359.
12. Elbanhawi, M.; and Simic, M. (2014). Sampling-based robot motion planning: A review. *IEEE Access*, 2, 56-77.
13. Kavraki, L.E.; Svestka, P.; Latombe, J.C.; and Overmars, M.H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4), 566-580.
14. LaValle, S.M. (1998). Rapidly-exploring random trees: A new tool for path planning. Retrieved on October 1, 2019 from http://msl.cs.illinois.edu/~lavalle /papers/Lav98c.pdf.
15. Karaman, S.; and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7), 846-894.
16. Karaman, S.; and Frazzoli, E. (2010). Optimal kinodynamic motion planning using incremental sampling-based methods. *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)*. Atlanta, US, 7681-7687
17. Akgun, B.; and Stilman, M. (2011). Sampling heuristics for optimal motion planning in high dimensions. *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Francisco, US, 2640-2645.
18. Kumar, B.S.; Abhimanyu, P.S.; Bharagav, B.V.V.P.; Agarwal, P.; and Krishna, K.M. (2010) RoboCup SSL Team Description. *Proceedings of the India Research Lab (IRL) RC*, India, 1-4.

19. Gammell, J.D.; Srinivasa, S.S.; and Barfoot, T.D. (2014). Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, US, 2997-3004.
20. Nasir, J.; Islam, F.; Malik, U.; Ayaz, Y.; Hasan, O.; Khan, M.; and Muhammad, M.S. (2013). RRT*-SMART: A rapid convergence implementation of RRT. *International Journal of Advanced Robotic Systems*, 10(7), 1-12.
21. Zaheer, S.; Jayaraju, M.; and Gulrez, T. (2015). Performance analysis of path planning techniques for autonomous mobile robots. *In 2015 IEEE international conference on electrical, computer and communication technologies (ICECCT)*. Coimbatore, India, 1-5
22. Klemm, S.; Oberländer, J.; Hermann, A.; Roennau, A.; Schamm, T.; Zollner, J.M.; and Dillmann, R. (2015). RRT∗-Connect: Faster, asymptotically optimal motion planning. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Zhuhai, China, 1670-1677.
23. Noreen, I.; Khan, A.; and Habib, Z. (2016). A comparison of RRT, RRT* and RRT*-smart path planning algorithms. *International Journal of Computer Science and Network Security (IJCSNS)*, 16(10), 20-27.
24. Connell, D.; and La, H.M. (2018). Extended rapidly exploring random tree–based dynamic path planning and replanning for mobile robots. *International Journal of Advanced Robotic Systems*. Miyazaki, Japan, 15(3), 2-15.
25. Ferguson, D.; Kalra, N.; and Stentz, A. (2006). Replanning with RRTS. *Proceedings IEEE International Conference on Robotics and Automation 2006 (ICRA 2006)*. Orlando, US, 1243-1248.
26. Eftekhari, S.; and Al-Obaidi, A.S.M. Investigation of a cruising fixed wing mini unmanned aerial vehicle performance optimization. *Indonesian Journal of Science and Technology*, 4(2), 280-293.
27. Aziz, M.; Huda, M.; Nandiyanto, A.B.D.; and Abdullah, A.G. (2018). Opportunity of frequency regulation using electric vehicles in Denmark. *Journal of Engineering Science and Technology* (*JESTEC*), 13(6), 1700-1712.