Journal of Engineering Science and Technology
Vol. 17, No. 6 (2022) 3742 - 3754

THE DEVELOPMENT OF A PATH PLANNING ALGORITHM COMBINING THE RAPIDLY-EXPLORING RANDOM TREE ALGORITHM AND THE PARTICLE SWARM OPTIMIZATION ALGORITHM

ADAM MALIK, MUHAMMAD ARIA RAJASA POHAN*

Electrical Engineering Department, Universitas Komputer Indonesia,
Jl. Dipati Ukur 102-116, Bandung 40132, Indonesia
*Corresponding Author: muhammad.aria@email.unikom.ac.id

Abstract

This study proposes a path-planning algorithm that combines the Rapidly-exploring Random Tree (RRT) and Particle Swarm Optimization (PSO) algorithms. This research aims to achieve near-optimal values with fast convergence using the proposed algorithm. The RRT algorithm can generate a faster path, but it is not optimal. The RRT algorithm is hybridized with the PSO algorithm to improve its performance and produce a near-optimal path. Each iteration of the combination process involves spinning a roulette wheel. The outcome of the spinning roulette wheel determines whether the following sampling process is carried out randomly, around the local best path, or around the global best path. Consequently, as the quality of a path increases, the probability of sampling along that path also increases. The PSO-RRT algorithm is the name given to the proposed algorithm. The PSO-RRT algorithm is tested and compared to the RRT* and informed RRT* algorithms. The test is based on simulation and employs various benchmarks such as clutter, multiple-narrow, square field, and tough passage environments. The results of this test, discussed in the result and discussion section, show that the proposed algorithm outperforms the RRT* and the informed RRT*.

Keywords: Fast convergence, Particle swarm optimization, Path planning, Rapidly-exploring random tree.

1. Introduction

Finding a path in a configuration area starting from an initial location to a destination location while fulfilling a set of rules is known as a path planning problem [1]. Path planning algorithms are used in various applications, such as autonomous vehicles [2], graphic animation [3], medical applications [4], robotic surgery [5], and cell transport [6]. The Rapidly-exploring Random Tree (RRT) algorithm is one of the most widely used path planning algorithms [7-10]. The RRT algorithm was developed by LaValle [11, 12]. The RRT algorithm can quickly solve problems in multidimensional systems [13]. However, it has the drawback of only offering a sub-optimal answer [14]. The RRT algorithm was then developed into RRT* [15, 16], which gives the optimal solution [17]. However, it has the disadvantage of slow convergence speed.

Several techniques have been developed to improve the performance of the RRT and RRT* algorithms. Kuffner and Lavalle introduced RRT-Connect [18] as a dual-tree version of RRT. Mashayekhi et al. [15] developed the RRT*-Connect algorithm, which combines RRT-Connect and RRT*. Ma et al. [19] introduced the Informed RRT* algorithm, which uses informed sampling on the RRT* after the first solution was found. Mashayekhi et al. [20] proposed the Informed RRT*-Connect algorithm, which works similarly to RRT*-Connect until the first path is discovered. After the first solution is found, the sampling area of Informed RRT*-Connect is limited in the same way it is in Informed RRT*.

In its development, the RRT algorithm has been hybridized with other algorithms to get better performance, such as Viseras et al. [21], who proposed a hybrid path planning algorithm by proposing a hybrid of the RRT algorithm with ACO. Implementing the RRT and ACO algorithms requires discretization from the configuration space, reducing the algorithm's performance in the high-dimensional configuration space.

Pohan et al. [22] developed a path planning algorithm that combines the RRT and the Ant Colony System (ACS) algorithms. RRT-ACS is the name of the algorithm. Then Aria proposed a path planning algorithm using the hybrid method between the informed RRT*-Connect algorithm and the local search algorithm, where the results of this algorithm proposal obtained an algorithm with a better speed of convergence value than the RRT* algorithm. Nevertheless, to the author's knowledge, no previous studies have tried to improve the quality of the RRT algorithm using the combination of the RRT algorithm with the PSO algorithm.

This study focuses on developing a path planning algorithm based on the RRT algorithm and combining it with the PSO algorithm to create an algorithm capable of rapidly reaching near-optimal values with a fast convergence speed. In each iteration of the combination process, a roulette wheel is spun. The outcome of the roulette wheel determines whether the following sampling process is done at random, around the local best path, or around the global best path. As a result, as the quality of a path increases, so does the likelihood of sampling along that path.

The proposed algorithm is dubbed the PSO-RRT algorithm. The PSO-RRT algorithm is tested and compared to the RRT* and informed RRT* algorithms. The test is based on simulation and includes benchmarks such as clutter, multiple-narrow, square field, and tough passage environments. The results of this test show that the proposed algorithm outperforms the RRT* and the informed RRT*.

2. Method

Algorithm 1 in Fig. 1 illustrates the PSO-RRT algorithm. Each iteration will involve spinning a roulette wheel based on the values of w, φ_1, and φ_2. If the roulette wheel's output value is less than w, the next sampling process will be carried out randomly. If the roulette wheel's output value is between w and φ_1, the next sampling process will be carried out around the local best path. Meanwhile, if the roulette wheel's output value is between φ_1 and φ_2, the next sampling process will be carried out around the global best path.

The value of φ_1 is the fitness value of the best path generated by a particle. The value of φ_2 is the fitness value of the best path generated by all particles. Thus, the higher the fitness value of the local and global paths, the higher the possibility of performing path-biased sampling on these paths. The length of the local and global paths will determine the fitness value of the path. The fitness value will increase as the path length decreases.

Algorithm 1: $X^{bs} \leftarrow$PSO-RRT(map)

```
% ========== Initialization
T ← InitializeTree()
T ← InsertNode(∅, q_init, T)
s ← 0
while termination condition not met do
  for k = 1 to number_of_particles do
    while s = 0 do
      c_best ← CalculateShortestPath(X^bs)
      n ← RouletteWheel[w, φ1, φ2]
      if n ≤ w
        (T, s) ← RRT*(T, map)
      else if w < n ≤ φ1
        (T, s) ← PathBiased(X_local(k), map)
      else
        (T, s) ← PathBiased(X_global, map)
      end if
    end while
    X(k) ← MakePath from T
    if fitness(X(k)) < fitness(X_local(k))
      X_local(k) = X(k)
      if fitness(X(k)) < fitness(X_global)
        X_global = X(k)
        PruneTree(T)
      end if
    end if
  end for
end while
```

Fig. 1. PSO-RRT algorithm.

The sampling process will be carried out using Algorithm 2 in Fig. 2 if the roulette wheel results determine that the next sampling process will be carried out randomly based on RRT* algorithm principle. The sampling process will be carried out using Algorithm 3 in Fig. 3 if the roulette wheel results determine that the next sampling process is based on the sampling around the local best path or global best path. Algorithm 3 used path-biased sampling, in which the sampling process is carried out around the best local or best global path to obtain a better path. The

sampling opportunity will be greater in the path area closer to the goal node in the path-biased sampling process.

The PSO-RRT algorithm has several advantages because it uses two types of sampling process methods. A sampling process that imitates the RRT* method, ensures that this algorithm has optimal asymptotic properties and avoids being trapped in the optimal local solution. A path-biased sampling will accelerate the convergence to better paths during planning [23].

Algorithm 2: $(T, s) \leftarrow \boldsymbol{RRT}^*(T, map)$

$q_{rand} \leftarrow InformedSample(k)$
$Q_{\boldsymbol{near}} \leftarrow NearNodes(T, q_{rand})$
$q_{parent} \leftarrow NearestNeighbor(q_{rand}, Q_{near}, T)$
$q_{new} \leftarrow Steer\ (q_{nearest}, q_{rand}, \Delta q)$
if $Obstaclefree(q_{new}, q_{nearest}, map)$ **then**
  $Q_{\boldsymbol{near}} \leftarrow Near(T, q_{new})$
  $q_{min} \leftarrow ChooseParent(q_{new}, Q_{near}, q_{nearest})$
  $T \leftarrow InsertNode(q_{min,} q_{new}, T)$
  if $d(q_{new}, q_{goal}) \leq \Delta q$ **and** $Obstaclefree$ **then**
    $T \leftarrow InsertNode(q_{goal} q_{new}, T)$
    $s \leftarrow 1$
  end if
end if

Fig. 2. The next sampling process will be carried out randomly based on RRT* algorithm principle.

Algorithm 3 : $X_{sol(new)} \leftarrow \boldsymbol{PathBiased}(X_{sol}, map)$

$q_{rand} \leftarrow SamplingNearBestPath(k)$
$Q_{\boldsymbol{near}} \leftarrow NearNodes(T, q_{rand})$
$q_{parent} \leftarrow NearestNeighbor(q_{rand}, Q_{near}, T)$
$q_{new} \leftarrow Steer\ (q_{nearest}, q_{rand}, \Delta q)$
if $Obstaclefree(q_{new}, q_{nearest}, map)$ **then**
  $Q_{\boldsymbol{near}} \leftarrow Near(T, q_{new})$
  $q_{min} \leftarrow ChooseParent(q_{new}, Q_{near}, q_{nearest})$
  $T \leftarrow InsertNode(q_{min,} q_{new}, T)$
  if $d(q_{new}, q_{goal}) \leq \Delta q$ **and** $Obstaclefree$ **then**
    $T \leftarrow InsertNode(q_{goal} q_{new}, T)$
    $s \leftarrow 1$
  end if
end if

Fig. 3. The next sampling process is based on the sampling around the local best path or global best path.

3. Results and Discussions

The output performances analysed in this research are the quality of the resulting path or solution cost, computation time, and the number of iterations required. The test is based on simulation using LabVIEW software and uses various benchmarks such as clutter, multiple-narrow, square field, and tough passage environments. The test results will be compared to the informed RRT* and RRT* algorithms to

determine whether the proposed algorithm outperforms them. The test consisted of 50 trials, with the iteration value in each environment set to 5000.

3.1. The performance of the PSO-RRT algorithm in a clutter environment

Table 1 displays the results obtained from testing in a clutter environment. The average cost value obtained by PSO-RRT in the clutter environment is 306.97, with an average iteration of 122.02 and an average time required of 327.66 s. The highest PSO-RRT cost value was obtained with a value of 311.23 and a time requirement of 631.61s. The informed RRT* algorithm's average cost value is 306.13, with an average iteration of 4101.54 and a time required of 1533.76 s. With a value of 306.6 and a time requirement of 1340.90 s, the highest informed RRT* solution cost value was obtained. The RRT* algorithm's average cost value is 309.25, with an average iteration of 3606.94 and a time required of 316.62 s. The highest RRT* solution cost value was 313.737, with a time requirement of 123.98 s.

Table 1. Average results from 50 trials in a cluttered environment.

Method	$\bar{x}$ Solution cost score	$\bar{x}$ Iterations required	$\bar{x}$ Time (s)	Smallest Solution Cost Value
RRT* algorithm	309.25	3606.94	316.62	306.63
Informed RRT* algorithm	306.13	4101.54	1533.76	305.67
PSO-RRT algorithm	306.97	122.02	327.66	305.71

Figure 4 depicts the convergence graph of each algorithm's solution costs after 50 trials. Figure 5 depicts the path with the lowest solution cost produced by each algorithm. In terms of iteration and computation, the PSO-RRT algorithm outperforms RRT* and informed RRT* in the clutter environment. In terms of solution costs, informed RRT* performs slightly better than PSO-RRT, but informed RRT* requires more time and iterations than PSO-RRT. PSO-RRT also produces the smallest solution cost of all trials, with the smallest solution, iterations, and computation time compared to RRT*.

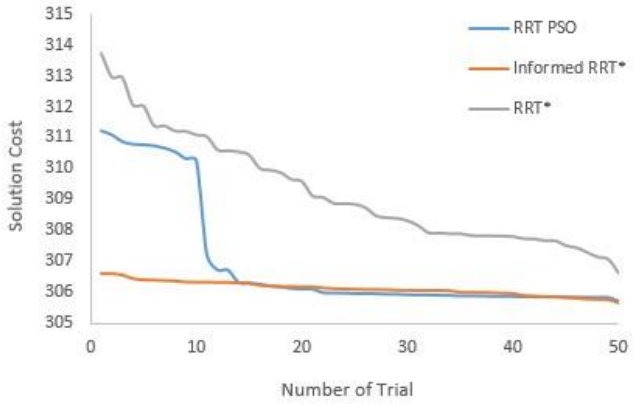

Fig. 4. The convergence graph of each algorithm's solution costs after 50 trials.

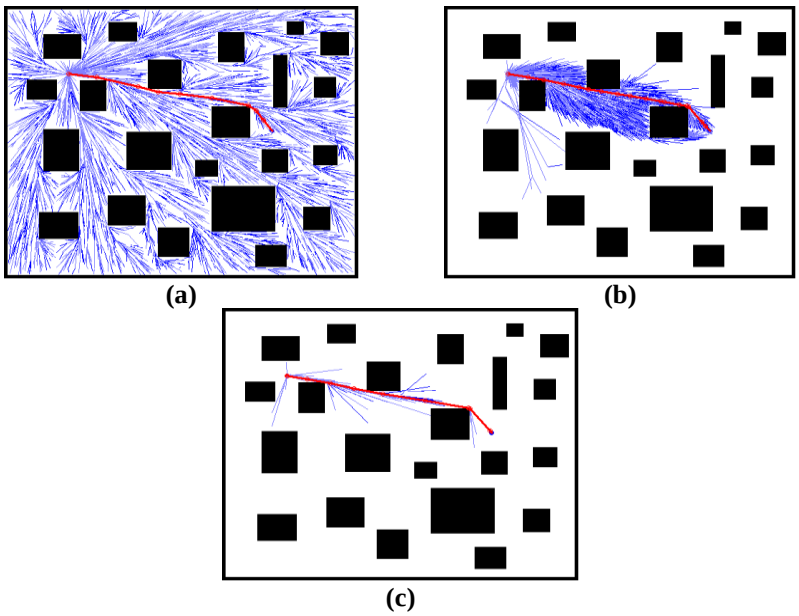

Fig. 5. The path with the lowest solution cost produced by each algorithm: (a) RRT*, (b) informed RRT*, and (c) PSO-RRT algorithms in a clutter environment.

3.2. The performance of the PSO-RRT algorithm in a multiple narrow environment

The results obtained in multiple narrow environments are shown in Table 2. The average cost value obtained by PSO-RRT in the narrow environment is 464.41, with an average iteration of 315.2 and an average time required of 1205.41 s. The highest PSO-RRT cost value was obtained with a value of 464.71 and a time requirement of 1298.12 s. The informed RRT* algorithm's average cost value is 468.26, with an average iteration of 4395.43 and a time required of 269.37 s. With a value of 495.53 and a time requirement of 151.08 s, the highest informed RRT* solution cost value was obtained. The RRT* algorithm's average cost value is 473.31, with an average iteration of 3901.65 and a time required of 131.51 s. The highest RRT* solution cost value was 530.21, with a time requirement of 94.91 s.

Table 2. Average outcomes from the 50 trials in the multiple narrow environment.

Method	$\bar{x}$ Solution cost score	$\bar{x}$ Iterations required	$\bar{x}$ Time (s)	Smallest Solution Cost Value
RRT* algorithm	473.31	3901.65	131.51	465.12
Informed RRT* algorithm	468.26	4395.43	269.37	464.44
PSO-RRT algorithm	464.41	315.2	1205.41	464.38

Figure 6 depicts the convergence graph of each algorithm's solution costs after 50 trials. Figure 7 depicts the path with the lowest solution cost produced by each algorithm. In multiple narrow environments, PSO-RRT produces the smallest average solution cost and the smallest number of iterations required from all experiments, although the informed RRT* and RRT* algorithms compute faster than PSO-RRT. PSO-RRT takes about 200 seconds longer to compute than RRT* and informed RRT*. On the other hand, PSO-RRT produced solutions for all trials, whereas the informed RRT* and RRT* could only produce solutions in some tests.

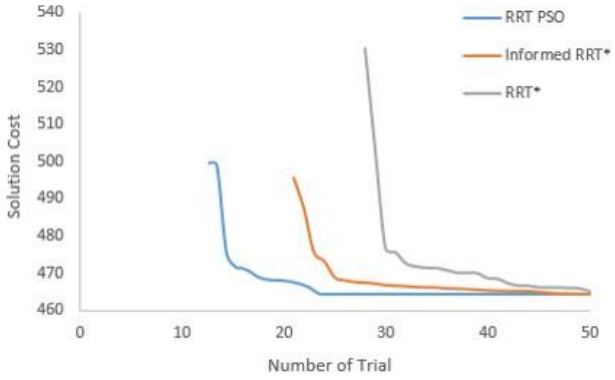

Fig. 6. The test result corresponds from the highest to the lowest value of (a) solution cost, (b) iteration, and (c) time computation algorithms in the multiple narrow environment of 50 trials.

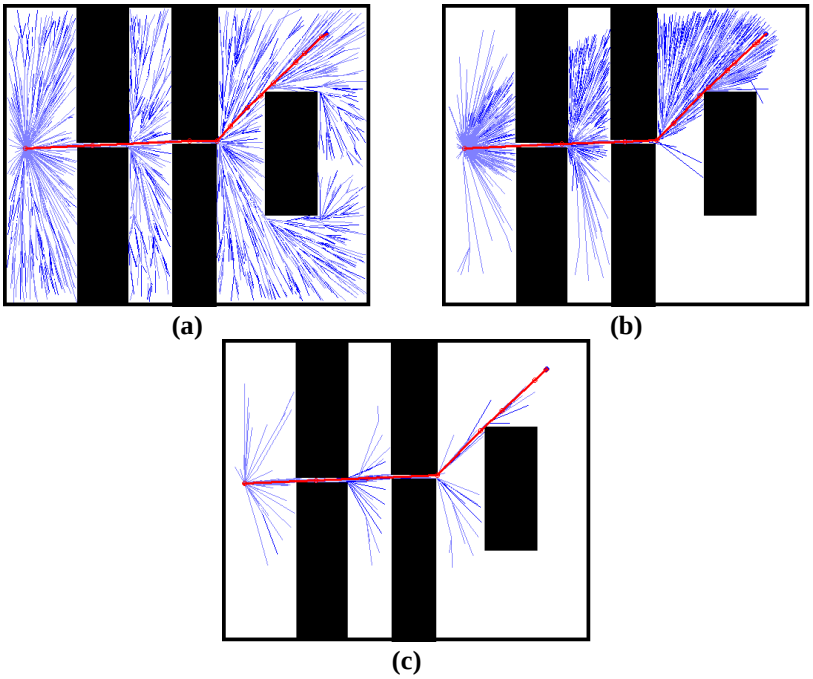

Fig. 7. The path with the lowest solution cost produced by each algorithm: (a) RRT*, (b) informed RRT*, and (c) PSO-RRT algorithms in a multiple narrow environment.

3.3. The performance of the PSO-RRT algorithm in a square field environment

The results obtained in square field environments are shown in Table 3. The average cost value obtained by PSO-RRT in the narrow environment is 325.60, with an average iteration of 122.46 and an average time required of 331.07 s. The highest PSO-RRT cost value was obtained with a value of 333.58 and a time requirement of 61.53 s. The informed RRT* algorithm's average cost value is 325.41, with an average iteration of 4278.24 and a time required of 958.78 s. With a value of 325.743 and a time requirement of 1162.20 s, the highest informed RRT* solution cost value was obtained. The RRT* algorithm's average cost value is 328.16, with an average iteration of 3404.24 and a time required of 414.93 s. The highest RRT* solution cost value was 331.458, with a time requirement of 749.18 s.

Table 3. Average outcomes from the 50 trials in the multiple square field.

Method	$\bar{x}$ Solution cost score	$\bar{x}$ Iterations required	$\bar{x}$ Time (s)	Smallest Solution Cost Value
RRT* algorithm	328.16	3404.24	414.93	326.52
Informed RRT* algorithm	325.41	4278.24	958.78	325.14
PSO-RRT algorithm	325.60	122.46	331.07	325.17

Figure 8 depicts the convergence graph of each algorithm's solution costs after 50 trials. Figure 9 depicts the path with the lowest solution cost produced by each algorithm. The same results are obtained in the square environment as in the clutter environment. PSO-RRT also has the lowest solution cost of all trials, with the fewest iterations and computation time compared to RRT* and informed RRT*. However, informed RRT* generates a very small difference in solution cost. Informed RRT* yields a solution cost value of 325.**1**4, while PSO-RRT yields 325.**1**7, a difference of 0.03.

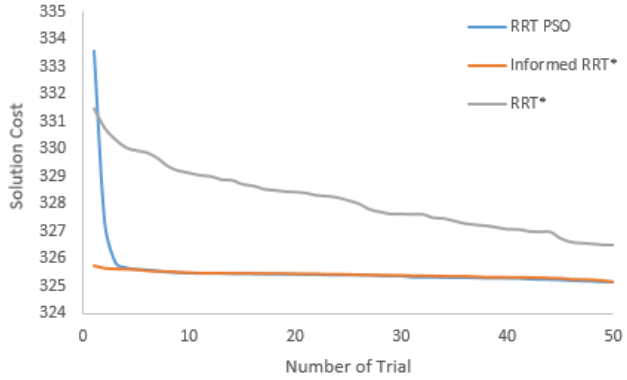

Fig. 8. The test result corresponds from the highest to the lowest value of (a) solution cost, (b) iteration, and (c) time computation algorithms in the square field environment of 50 trials.

3.4. The performance of the PSO-RRT algorithm in a tough passages environment

Table 4 displays the results obtained from testing in a tough passages environment. The average cost value obtained by PSO-RRT in the tough passages environment is 220.74, with an average iteration of 243.9 and an average time required of 665.65 s. The highest PSO-RRT cost value was obtained with a value of 222.10 and a time requirement of 916.33 s. The informed RRT* algorithm's average cost value is 221.94, with an average iteration of 4334.62 and a time required of 794.15 s. With a value of 223.80 and a time requirement of 520.31 s, the highest informed RRT* solution cost value was obtained. The RRT* algorithm's average cost value is 229.35, with an average iteration of 3971.7 and a time required of 346.72 s. The highest RRT* solution cost value was 236.02, with a time requirement of 184.46 s.

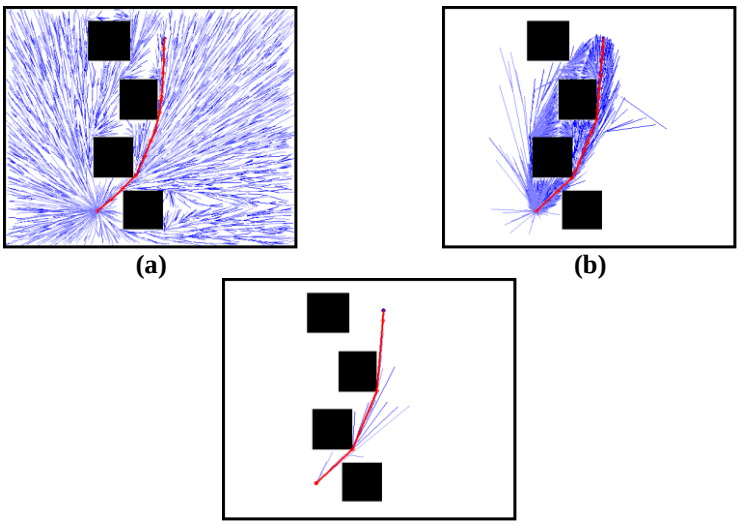

Fig. 9. The path with the lowest solution cost produced by each algorithm: (a) RRT*, (b) informed RRT*, and (c) PSO-RRT algorithms in a square field environment.

Table 4. Average outcomes from the 50 trials in the multiple narrow environment.

Method	$\bar{x}$ Solution cost score	$\bar{x}$ Iterations required	$\bar{x}$ Time (s)	Smallest Solution Cost Value
RRT* algorithm	229.35	3971.7	346.72	222.13
Informed RRT* algorithm	221.94	4334.62	794.15	220.23
PSO-RRT algorithm	220.74	243.9	665.65	219.93

Figure 10 depicts the convergence graph of each algorithm's solution costs after 50 trials. Figure 11 depicts the path with the lowest solution cost produced by each algorithm. The same results are obtained in the tough passage environment as in multiple narrow environments. Although RRT* computes faster than PSO-RRT,

PSO-RRT produces the smallest solution cost and iteration of all trials compared to RRT* and informed RRT*.

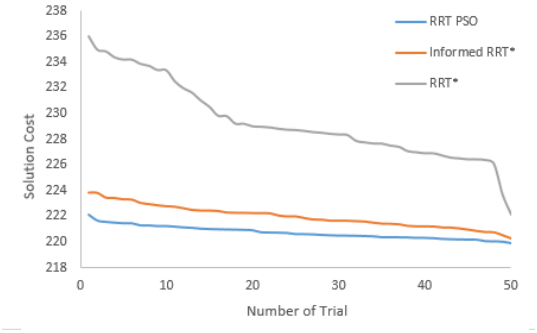

Fig. 10. The test result corresponds from the highest to the lowest value of (a) solution cost, (b) iteration, and (c) time computation algorithms in the tough passages environment of 50 trials.

Based on the results of the tests, it is clear that the informed RRT* algorithm outperforms the RRT* algorithm. This is consistent with Gammell et al. [24] and Zhang et al. [25]. The PSO-RRT algorithm, which combines the RRT* and PSO algorithms, outperforms the RRT* and Informed RRT* algorithms. The superior performance of the PSO and RRT hybridization algorithms is consistent with Shami et al. [26], who claim that the hybridization process can produce a new algorithm that outperforms the individual algorithms. Chai et al. [27], Sayah and Hamouda [28], as well as Kiran et al. [29] also hybridized an algorithm with the PSO algorithm and all reported obtaining a new algorithm that is better than the individual algorithms.

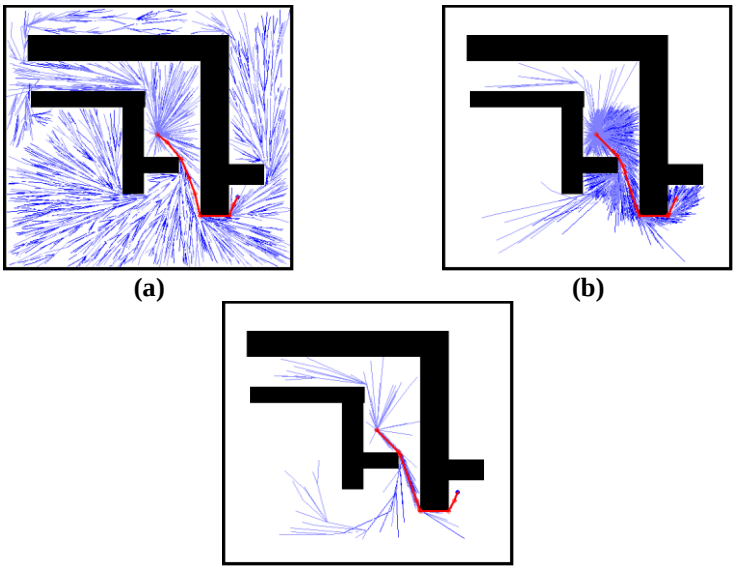

Fig. 11. The path with the lowest solution cost produced by each algorithm: (a) RRT*, (b) informed RRT*, and (c) PSO-RRT algorithms in tough passages environment.

4. Conclusion

The path planning algorithm was created using a combination of the RRT and PSO algorithms and was tested using simulation using LabVIEW software. In each environment that has been tried, the results have been different.

The proposed algorithm outperforms the RRT* algorithm and the informed RRT* algorithm by obtaining a small cost value, iteration, and computational time in the clutter and square field environments. In multiple narrow environments, the PSO-RRT algorithm provides a small cost value and iteration cost, but the PSO-RRT algorithm takes considerable computational time.

Furthermore, unlike the RRT* and informed RRT* algorithms, the PSO-RRT algorithm can obtain a cost solution value in each trial. For 50 trials, the RRT* and informed RRT* algorithms can only produce 30 cost solution values.

In tough passage environments, the PSO-RRT algorithm outperforms the informed RRT* and RRT* algorithms. The RRT* algorithm is slightly faster in terms of computational time, but the PSO-RRT algorithm has the lowest solution cost and iteration value compared to the RRT* algorithm.

Therefore, using the solution cost value, iteration, and computational time generated by each test, it can be concluded that the PSO-RRT algorithm outperforms the RRT* algorithm and the informed RRT* algorithm.

Acknowledgement

This research was funded by the grant from the Ministry of Education, Culture, Research, and Technology Republic of Indonesia, with contract numbers between LLDIKTI4 and Universitas Komputer Indonesia No. 002/SP2H/RT-MONO/BATCH 2/LL4/2022 in the Applied Research scheme for the fiscal year 2022.

References

1. Paden, B.; Čáp, M.; Yong, S.Z.; Yershov, D.; and Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1), 33-55.
2. Li, H.; Liu, W.; Yang, C.; Wang, W.; Qie, T.; and Xiang, C. (2021). An optimization-based path planning approach for autonomous vehicles using dynEFWA-artificial potential field. *IEEE Transactions on Intelligent Vehicles*, 7(2), 263-272.
3. Jin, B.; and Hong, L. (2015). Improved artificial bee colony algorithm and application in path planning of crowd animation. *International Journal of Control and Automation*, 8(3), 53-66.
4. Zhao, B.; Shao, S.; Lei, L.; Wang, X.; Yang, X.; Wang, Q.; and Hu, Y. (2022). Curve fitting-based dynamic path planning and tracking control for flexible needle insertion. *IEEE Transactions on Medical Robotics and Bionics*, 4(2), 436-447.
5. Frisken, S.; Luo, J.; Haouchine, N.; Pieper, S.; Wang, Y.; Wells, W. M.; and Golby, A.J. (2021). Incorporating uncertainty into path planning for minimally invasive robotic neurosurgery. *IEEE Transactions on Medical Robotics and Bionics*, 4(1), 5-16.

6. Khadem, M.; Rossa, C.; Usmani, N.; Sloboda, R.S.; and Tavakoli, M. (2017). Robotic-assisted needle steering around anatomical obstacles using notched steerable needles. *IEEE journal of biomedical and health informatics*, 22(6), 1917-1928.
7. Connell, D.; and Manh La, H. (2018). Extended rapidly exploring random tree–based dynamic path planning and replanning for mobile robots. *International Journal of Advanced Robotic Systems*, 15(3), 1-15.
8. Wang, X.; Luo, X.; Han, B.; Chen, Y.; Liang, G.; and Zheng, K. (2020). Collision-free path planning method for robots based on an improved rapidly-exploring random tree algorithm. *Applied Sciences*, 10(4), 1381.
9. Liu, B.; Feng, W.; Li, T.; Hu, C.; and Zhang, J. (2020). A variable-step RRT* path planning algorithm for quadrotors in below-canopy. *IEEE Access*, 8, 62980-62989.
10. Tian, L.; Zhang, Z.; Zheng, C.; Tian, Y.; Zhao, Y.; Wang, Z.; and Qin, Y. (2021). An improved rapidly-exploring random trees algorithm combining parent point priority determination strategy and real-time optimization strategy for path planning. *Sensors*, 21(20), 6907.
11. Lavalle, S.M.; and Kuffner, J.J. (2001). Randomized kinodynamic planning, *The International Journal of Robotics Research*, 20(5), 378-400.
12. Becerra, I.; Suomalainen, M.; Lozano, E.; Mimnaugh, K. J.; Murrieta-Cid, R.; and LaValle, S.M. (2020). Human perception-optimized planning for comfortable VR-based telepresence. *IEEE Robotics and Automation Letters*, 5(4), 6489-6496.
13. Katrakazas, C.; Quddus, M.; Chen, W.H.; and Deka, L. (2015). Real-time motion planning methods for autonomous on-road driving: state-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60, 416-442.
14. Karaman, S.; and Frazolli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7), 846-894.
15. Mashayekhi, R.; Idris, M.Y.I.; Anisi, M.H.; and Ahmedy, I. (2020). Hybrid RRT: A semi-dual-tree RRT-based motion planner. *IEEE Access*, 8, 18658-18668.
16. Xinyu, W.; Xiaojuan, L.; Yong, G.; Jiadong, S.; and Rui, W. (2019). Bidirectional potential guided RRT* for motion planning. *IEEE Access*, 7, 95046-95057.
17. Salzman, O.; and Halperin, D. (2016). Asymptotically near-optimal RRT for fast, high-quality motion planning. *IEEE Transactions on Robotics*, 32(3), 473-483.
18. Chen, J.; Zhao, Y.; and Xu, X. (2021). Improved RRT-connect based path planning algorithm for mobile robots. *IEEE Access*, 9, 145988-145999.
19. Ma, N.; Wang, J.; Liu, J.; and Meng, M. Q. H. (2021). Conditional generative adversarial networks for optimal path planning. *IEEE Transactions on Cognitive and Developmental Systems*, 14(2), 662-671.
20. Mashayekhi, R.; Idris, M.Y.I.; Anisi, M.H.; Ahmedy, I.; and Ali, I. (2020) Informed RRT*-connect: an asymptotically optimal single-query path planning method, *IEEE Access*, 8, 19842-19852.

21. Viseras, A.; Losada, R.O.; and Merino, L. (2016). Planning with ants: efficient path planning with rapidly exploring random trees and ant colony optimization. *International Journal of Advanced Robotic Systems*, 13(5), 1-16.
22. Pohan, M.A.R.; Trilaksono, B.R.; Santosa S.P.; and Rohman, A.S. (2021) Path planning algorithm using the hybridization of the rapidly-exploring random tree and ant colony systems, *IEEE Access*, 9, 153599-153615.
23. Véras, L.G.D.O.; Medeiros, F.L.L.; and Guimaráes, L.N.F. (2019) Systematic literature review of sampling process in rapidly-exploring random trees, *IEEE Access*, 7, 50933-50953.
24. Gammell, J.D.; Barfoot, T.D.; and Srinivasa, S.S. (2018). Informed Sampling for Asymptotically Optimal Path Planning. *IEEE Transactions on Robotics*, 34(4), 966-984.
25. Zhang, C.; Zhou, L.; and Liu, H. (2019). LaLo-check: A path optimization framework for sampling-based motion planning with tree structure. *IEEE Access*, 7, 100733-100746.
26. Shami, T.M.; El-Saleh, A.A.; Alswaitti, M.; Al-Tashi, Q.; Summakieh, M. A.; and Mirjalili, S. (2022). Particle swarm optimization: a comprehensive survey. *IEEE Access*, 10, 10031-10061.
27. Chai, Z.; Nwachukwu, A.; Zagayevskiy, Y.; Amini, S.; and Madasu, S. (2021). An integrated closed-loop solution to assisted history matching and field optimization with machine learning techniques. *Journal of Petroleum Science and Engineering*, 198, 108204.
28. Sayah, S.; and Hamouda, A. (2013). A hybrid differential evolution algorithm based on particle swarm optimization for nonconvex economic dispatch problems. *Applied Soft Computing*, 13(4), 1608-1619.
29. Kıran, M.S.; Gündüz, M.; and Baykan, Ö.K. (2012). A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum. *Applied Mathematics and Computation*, 219(4), 1515-1521.