Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/79/20241597

An improved DWA algorithm in agricultural robot

Yucheng Zhang

Mechatronics and Robotics, University of Leeds, Lanzhou, China

ml223yz@leeds.ac.uk

Abstract. The study of robot research necessitates the exploration of path planning. The goal of this paper is to enable multiple robots with different functions to reach the designated place quickly and accurately in the orchard environment. In comparison, avoiding collisions, deadlocks, and other complications. This paper has chosen to employ the upgraded A-star algorithm in the global layer as its strategy. An improved DWA algorithm is used in the local layer. In addition, the Voronoi graph method is introduced to limit the search area and solve the obstacle avoidance problem of multiple robots. The left-turn method is applied to the deadlock problem of multiple robots. Finally, the obstacle avoidance efficiency in a multi-obstacle environment is simulated to verify the algorithm's effectiveness.

Keywords: path planning, Improved DWA algorithm, dynamic obstacle avoidance, AGV.

1. Introduction

Agricultural products are becoming increasingly sought after annually [1]. At the same time, the world is facing a labor shortage. As the world's population increases, so does the need for food. With the development of artificial intelligence, agricultural automation can help ease the burden on farmers [2]. Fruit harvesting is one of the most laborious tasks in the agricultural industry [3]. Mobile robots can move autonomously without the help of humans. When the robot needs to perform a particular action, a control system coordinates the robot to complete all the steps. Mobile robotics includes the fields of movement, perception, cognition, and navigation [4]. Many countries around the world have developed cucumbers, peppers, apples, oranges, peaches, and other picking robots [5]. The use of robots for fruit picking has been studied for three decades. One of the first results in this field was the identification of fruit by a black-and-white camera for picking [6]. In the path aspect, the path planning is realized by wheel track. At present, mobile robot path planning has made great progress. However, for agricultural robots, the path planning development is not yet mature due to the more complex environment [7].

Since 1980, developed countries in Europe and Japan have conducted much research on fruit and vegetable-picking equipment [8]. A tomato-picking robot developed by Japanese researchers in 1980 was moved by wheeled tracks [9]. The picking speed is 15s/ piece. In 2019, the cucumber-picking robot developed by Yang Changhui's team at China Agricultural University had a picking rate of 80.51% and an obstacle avoidance success rate of 75.79% [10]. Although much research has been done in many countries, the efficiency of fruit-picking robots is still low. Consequently, Agricultural Robot motion planning has recently become a highly sought-after area of study [11].

Fruit-picking robots must have a well-thought-out plan in place. The picking efficiency of the robot is significantly impacted. The orchard's security is also compromised [12]. Path planning is not only

DOI: 10.54254/2755-2721/79/20241597

necessary to help the robot find the optimal picking path. In orchards, the distribution of fruit is usually uneven. Multiple robots must work together in an orchard for tasks such as picking and transporting. If the robot picks randomly, it will waste time and energy [13]. Therefore, the path planning task for fruit-picking robots requires the following points. First, find the shortest and fastest picking route. Secondly, the robot must have the ability to avoid obstacles. Obstacles mainly include prohibited obstacles and dynamic obstacles. Forbidden obstacles are mainly composed of environmental elements, including fruit trees, fences, rocks, pools etc. [14]. A fruit-picking robot entering a pool, rocks, or trees can cause damage to the robot. Dynamic obstacles mainly include robots completing other tasks and workers moving freely. Collisions between robots can cause damage to machines. Simultaneously, it is essential to guarantee the protection of laborers in the orchard [15].

This paper contains six parts. We introduce the background and importance of agricultural robots and path planning. Then we introduce several related and important path-planning algorithms' advantages, disadvantages, and background. Next, we introduces the mathematical derivation, process, and principles related to our algorithm. The next part obtains the results of different algorithms and actual operation results through simulation testing. Then we summarize and compares the existing algorithms and our improved algorithm. Finally, we summarize our results and concludes with an introduction to our future work.

2. Related Work

A multitude of factors hinder the development of path-planning algorithms. There are two types of early path planning. One is the Dijkstra algorithm proposed by Edsger W. Dijkstra in 1956 [16]. However, Dijkstra's algorithm has some limitations. For example, Dijkstra's algorithm can only calculate correctly in graphs with non-negative weights. In addition, dynamic graphs are recalculated every time, dramatically affecting work efficiency.

Furthermore, Dijkstra's algorithm cannot consider more complex requirements for practical applications [17]. In 1958, Richard Bellman and Lester Ford independently created the Bellman-Ford algorithm, which was one of the earliest approaches to the problem [18]. Bellman-Ford addresses the issue of Dijkstra's algorithm's inability to accommodate negative weights. Dynamic programming forms the foundation of the Bellman-Ford algorithm. Therefore, it is much easier to implement than Dijkstra's priority queue. It can also be used for preprocessing of more complex algorithms. However, like Dijkstra, Bellman-Ford's algorithm only works in dynamic environments. Because when the environment changes, it needs to recalculate the path. It will undoubtedly waste computing power and reduce efficiency. To sum up, the Bellman-Ford algorithm is superior to the Dijkstra algorithm in an environment with negative weight. If there is no negative weight, Dijkstra's algorithm is more efficient.

A star algorithm was proposed by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968 [19]. Dijkstra's algorithm is integrated with a heuristic function in a star algorithm. It surpasses Dijkstra's algorithm in terms of efficiency. Dijkstra's algorithm prioritizes breadth, while the search lacks visibility. Moreover, always find the overall optimal path. However, A star algorithm depends on selecting a heuristic function, and setting parameters will significantly affect the search efficiency. In addition, the above algorithms are limited to raster maps.

In 1980, with the development of robotics, path planning became a hot issue again and broke through the form of raster maps that could be applied to high-dimensional Spaces. In the year 2000, a technique of random selection was suggested. Rapidly exploring Random Trees (RRT) and Probabilistic Roadmaps (PRM) are two classical methods of random sampling. RRT algorithm can explore ample space quickly and be applied to high dimensional space [20]. Nevertheless, the RRT algorithm usually cannot find the optimal path. Furthermore, the results obtained from each calculation may be different and are random. LE Kavraki first proposed the PRM algorithm, P Svestka et al. in 1996 [21]. The PRM algorithm first sets some sampling points in the environment. Then, remove the issue in the obstacle. Next, connect the sampling points with line segments. Then, drag the line of the farther point and the line through the obstacle. Finally, the Dijkstra or The star algorithm determines the distance between the starting point and the endpoint through calculations. This method simplifies planning space so paths

DOI: 10.54254/2755-2721/79/20241597

can be found quickly and efficiently. The PRM algorithm is suitable for high-latitude robot planning. The PRM algorithm requires a considerable amount of time to construct a map, and the resulting maps may take up a considerable amount of storage capacity. Due to the complexity of the orchard environment, dynamic map planning ability needs to be improved.

The Voronoi diagram is also called Dirichlet tessellation. In 1908, Ukrainian mathematician Georgy Voronoi began research and proposed the foundation of Voronoi [22]. With the development of computer technology, Voronoi diagrams have also made outstanding contributions to robot path planning. Lloyd's algorithm takes advantage of the non-overlapping nature of the buffered Voronoi cell (BVC) and uses sensors to collect data in a convex environment [23]. Typically, the seed resides within the BVC due to the inherent uncertainty of the robot. Ding designates the robot's location as a zone of unpredictability (UV)[24]. This approach confines the robot to the Veno cell to ensure a safety buffer between the robot and the obstacle. Subsequently, Chen suggests the utilization of the convex uncertain Voronoi (CUV) diagram to guarantee the robot's security [25].

3. Problem Formulation

In the 1990s, two algorithms, the Dynamic Window Approach (DWA) and the Space-Time A* algorithm, were developed to address the challenge of dynamic environment planning and moving obstacles in intricate settings [26]. The DWA algorithm is a component of the local path planning algorithm. The DWA algorithm establishes a velocity space (v,ω) based on the present location and speed of the mobile robot. The robot's path at these velocities is subsequently computed. The trajectory is assessed by the evaluation function. Subsequently, the velocity that aligns with the ideal path is chosen as the velocity of the robot. In the end, continue cycling until the mobile robot arrives at the intended location. The DWA algorithm is capable of swiftly responding to any hindrances in its surroundings.

Therefore, the DWA algorithm suits local path planning in an unknown environment. However, DWA is primarily concerned with local navigation. So, DWA is only sometimes able to find the global optimal path. Moreover, falling into deadlocks or obstacle avoidance difficulties in complex environments is easy. Space-Time A* (STA*) uses a time-space grid [27]. Each represents a specific location and time. If another robot arrives at position X at time T. This location will not be available to other robots. STA* avoids obstacles at specific locations at specific times. However, the computing requirements are often more significant. STA* is typically used in multi-robot systems or highly dynamic environments. Orchard environments are complex but do not require many robots working simultaneously. Therefore, STA* is not suitable for orchard environments. The Voronoi diagram algorithm offers a method for circumventing obstacles in order to maintain the highest possible distance between the robot and the obstacle. The Voronoi graph algorithm initially utilizes the boundary points of obstacles within the surroundings to produce Voronoi graphs. The robot will traverse the periphery of the Voronoi diagram. Subsequently, employ either Dijkstra's algorithm or A star algorithm to determine the most efficient route connecting the initial and final points. The Voronoi graph algorithm fails to determine the most direct route. Despite this, he is able to maintain a safe distance between the robot and any hindrances. By making the path strategy smoother, we can get good results.

3.1. A star

The heuristic function is utilized by a star algorithm. The actual consumption from the node to node *n* is denoted by *g(n)*, while the estimated consumption from the node to the target node is represented by *h(n)* [28]. This calculation is more effective than Dijkstra's algorithm [28].

$$f(n) = g(n) + h(n) \tag{1}$$

A heuristic search is employed by a star algorithm to regulate the expansion of nodes. The fundamental concept of this approach is to determine the path nodes by means of the evaluation function. Slowly advance towards the desired location by enlarging the subordinate nodes. Go from the beginning to the end to create a plan of action. A star algorithm is depth-first to reduce search volume. The advantages of the A star algorithm are simple, fast, and adaptable to global static programming. However,

DOI: 10.54254/2755-2721/79/20241597

there is room for improvement in path smoothness, locally optimal path, and path planning in dynamic environments. The detailed steps of the A star algorithm is shown in Figure 1.

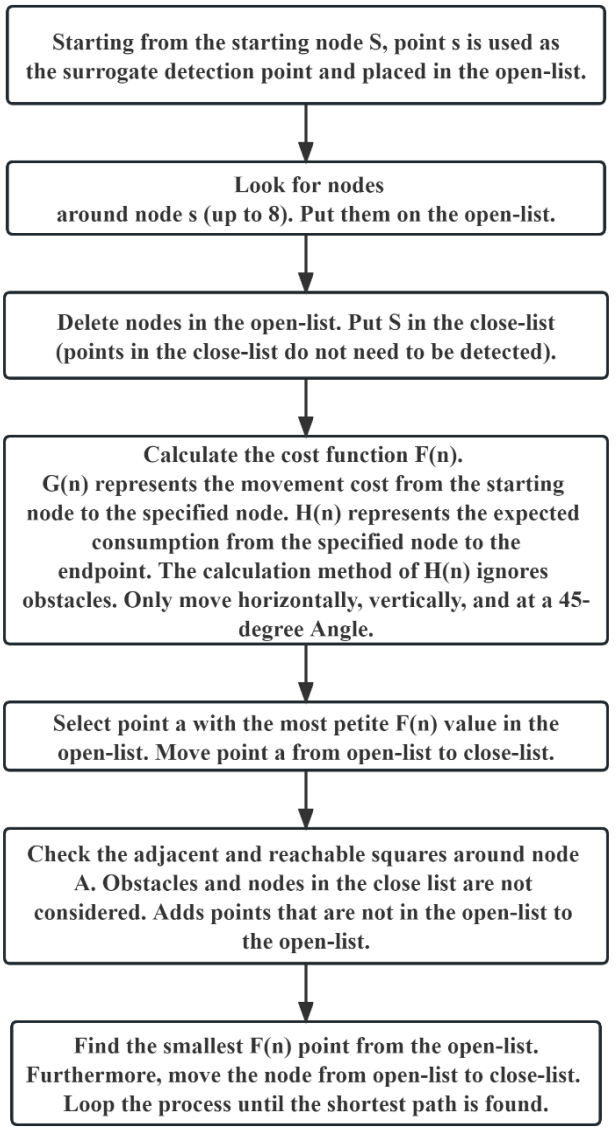

Figure 1. A Star flow chart

3.2. Rapidly exploring Random Trees (RRT)

As illustrated in Figure 2. The RRT algorithm considers the starting point of the root node and performs spatial dot sampling. Select a point X (rand) at random from the given space and ascertain the nearest node X (near) of X (rand); the branch's growth direction involves connecting two points. Subsequently, establish a new starting point X (new) by determining the step size, which is the termination point of the step size. This process is repeated until all obstacles are bypassed and the target point is reached [29].

DOI: 10.54254/2755-2721/79/20241597

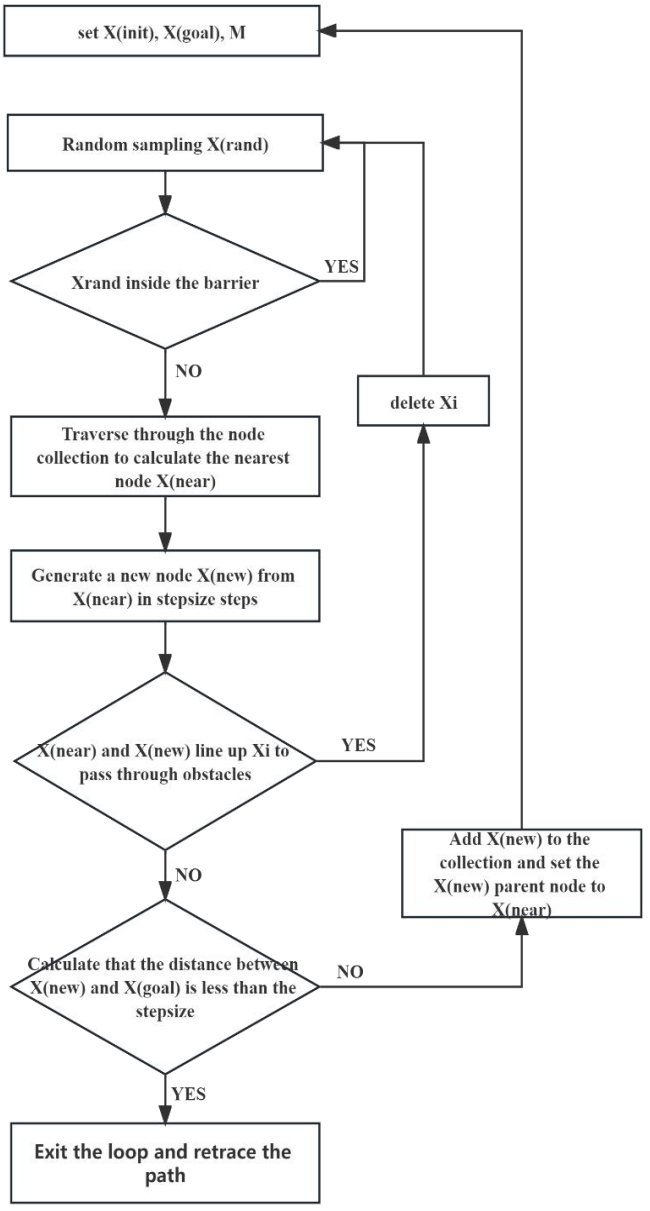

Figure 2. RRT flow chart

3.3. Dynamic window Approach

The Dynamic Window Approach (DWA) is a technique that allows for the selection of paths at a local level. DWA converts the robot's movement into velocity space. Regional path planning is performed while following constraints. Despite this, the DWA algorithm rapidly descends into the local optimal dilemma and cannot locate the global optimal path. Another disadvantage is that it is susceptible to the environment. Recalculation is required when circumstances change. Therefore, it is not efficient in complex environments.

The DWA algorithm first collects real-time velocity samples of the robot. The motion trajectories generated in a period under the sample velocity are predicted and evaluated. The ideal path is chosen to guide the robot's movement in accordance with the optimal path. The present linear and angular velocity of the robot determines the kinematic depiction of the robot. DWA algorithm mainly has three steps: velocity sampling, trajectory prediction, estimation, and evaluation [30].

DOI: 10.54254/2755-2721/79/20241597

3.3.1. Velocity sampling. The environment and the speed of the robot motor mainly limit the robot's speed. The robot has some limitations in velocity sampling space V_s (v, ω).

The first is the speed boundary limit, according to the maximum motor speed of the robot and environmental restrictions. The velocity sampling space can then be described as V_m.

$$V_m = \{ (v, \omega) \mid v \in [v_{min}, v_{max}], \omega \in [\omega_{min}, \omega_{max}] \} \tag{2}$$

In the formula, $v_{min}$and $v_{max}$are the robot's minimum and maximum linear velocities, respectively. Wmin and Wmax are the minimum and maximum angular velocities, respectively.

The second is acceleration limitation, also due to the robot's motor limitations. The robot's linear and angular acceleration are both constrained by boundaries. When the robot's acceleration and deceleration are the same, the velocity sampling space V_d can be described as the following formula.

$$V_d = \{(v, \omega) \mid \begin{matrix} v \in [v_c - a_{v_{max}} \times \Delta t, v_c + a_{v_{max}} \times \Delta t] \\ \omega \in [\omega_c - a_{w_{max}} \times \Delta t, \omega_c + a_{w_{max}} \times \Delta t] \end{matrix} \} \tag{3}$$

The final is the environmental obstacles limit. The robot must have real-time dynamic obstacle avoidance ability in the orchard environment. Therefore, the constraint condition that the robot does not collide with the surrounding environment at a particular time is expressed as V_a.

$$V_a = \begin{Bmatrix} v \in \left[v_{min}, \sqrt{2 \cdot dist(v, \omega) \cdot a_{vmax}}\right] \\ \omega \in \left[\omega_{min}, \sqrt{2 \cdot dist(v, \omega) \cdot a_{\omega max}}\right] \end{Bmatrix} \tag{4}$$

In the formula, $dist(v, \omega)$ indicates the gap between the simulated path and the environmental barrier in the present situation. When the robot is very close to the environmental obstacle, it will safely avoid it with minimum speed. To put it succinctly, the robot's velocity sampling space is the point where three velocity Spaces intersect.

$$V_s = V_m \cap V_d \cap V_a \tag{5}$$

3.3.2. Generate Motion Candidates. After determining the sampling space V_s, the DWA algorithm reads the speed data at a specific sampling frequency. The sampling frequencies E_v and E_w of the linear and angular velocities are set in the velocity space, respectively. The angular velocity can also be calculated mathematically. The number of data sets of the sampling speed can be determined. As shown below, the upper and lower limits of the velocity space for v_{high}, v_{low}, w_{high}, w_{low}.

$$n = \left[\left(v_{high} - v_{low}\right)/E_v\right] \cdot \left[\frac{\left(w_{high} - w_{low}\right)}{E_w}\right] \tag{6}$$

After sampling a group of(v, ω), trajectory prediction is made by the robot kinematics model.

3.3.3. Select Best Motion. After determining the range of velocity constraints, many simulated trajectories obtained are evaluated and selected. Choose an optimal path corresponding to the speed of the drive. This path is considered optimal in the current environment while satisfying the robot's dynamic constraints. The evaluation formula is as follows.

$$G(v, \omega) = \sigma\big(\alpha \cdot heading(v, \omega)\big) + \sigma\big(\beta \cdot dist(v, \omega)\big) + \sigma\big(\gamma \cdot velocity(v, \omega)\big) \tag{7}$$

In this equation heading(v, ω)is the azimuth evaluation function. Its function is to evaluate the error $\Delta\theta$ of the Angle between the direction of the end position of the trajectory and the line of the target point generated at the current sampling speed. And$dist(v, \omega)$ is the distance evaluation function measures the proximity between the simulated trajectory and the obstacle at the present velocity. The $velocity(v, \omega)$ is a velocity evaluation function. It represents the magnitude of the current velocity,

DOI: 10.54254/2755-2721/79/20241597

which can be expressed directly in terms of the magnitude of the current linear velocity. α, β, and γ are the coefficients of the evaluation function.

3.4. Buffered Voronoi Diagram

Voronoi diagrams can divide the plane according to a specific set of points. The Euclidean distance between any two points p and q can be expressed as follows.

$$dist(p,q) = (p_x - q_x)^2 + \left(p_y - q_y\right)^2 \tag{8}$$

Let S be a set of distinct points p_i in the plane, where $i = 1,2,\ldots,n$. The Voronoi region $R(p_i)$ for a point p_i is defined as:

$$R(p_i) = \{q \in \mathbb{R}^2 \mid \forall p_j \in S \backslash \{p_i\}, d(q,p_i) \leq d\left(q,p_j\right)\} \tag{9}$$

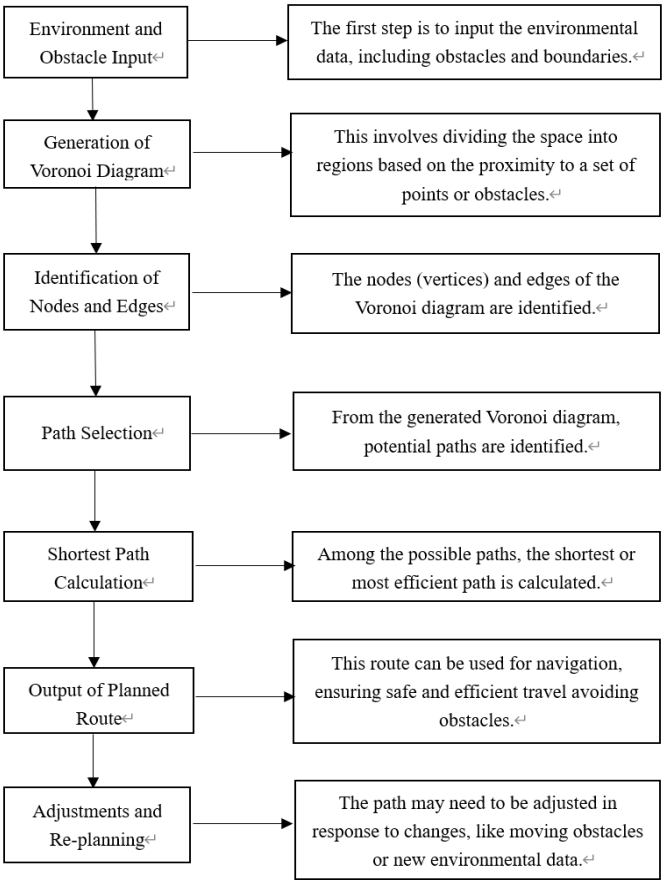

Figure 3. Several steps for path planning using Voronoi diagrams

As shown in Figure 3. There are several steps for path planning using Voronoi diagrams. To begin with, the robot's movement setting and the mobile robot's condition are established. The initial action is to devise a route for every mobile robot, commencing from the initial point and concluding at the endpoint, while partitioning the buffered Voronoi cells to form the map. Subsequently, the mobile robot commenced its journey along the designated path. Once the unit time has elapsed, all robots come to a halt and proceed to redraw the buffered Voronoi diagram. Subsequently, the robots recommence their movement and replicate the procedure.

DOI: 10.54254/2755-2721/79/20241597

4. Simulation

4.1. A star

In Figure 4, the obstacle is represented by black, the robot's starting point is indicated by the red dot, the robot's target position is indicated by the green dot, and the robot's trajectory is indicated by the green line. There are two routes in the picture. The route above is shorter. When the path is clear, the robot chooses the upper path. The robot determines the following path when we add an obstacle that blocks the path. The A star algorithm is able to find the optimal path according to the simulation results. A reliable path can be obtained by setting an appropriate heuristic function. This path can be used as a reference path to participate in the evaluation function of the subsequent DWA algorithm.

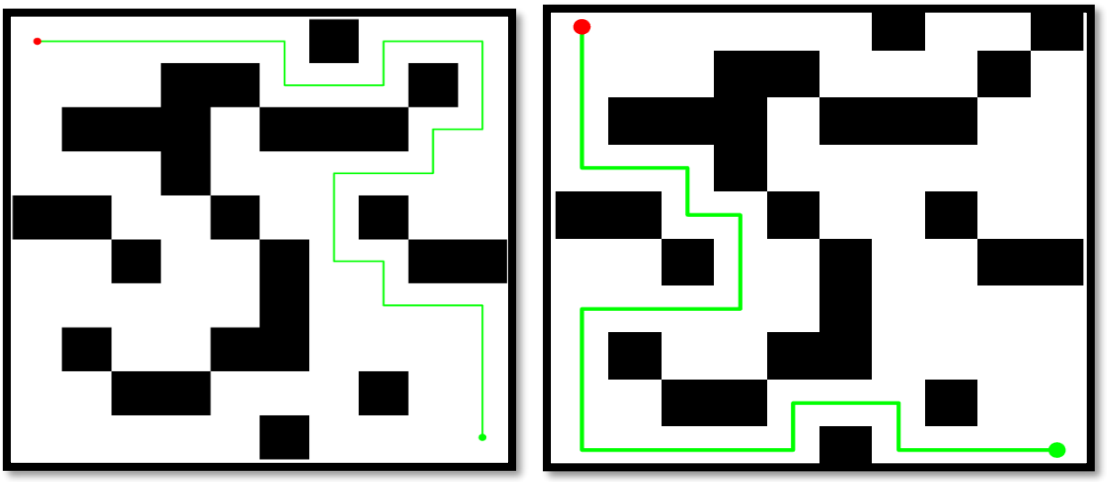

Figure 4. Simulation results of the A-star algorithm. The obstacle is represented by the black, the starting point by the red dot, the end point by the green dot, and the planned path by the green line.

4.2. Rapidly exploring Random Trees

Likewise, in Figure 5, the red dot signifies the robot's initial location, the green dot denotes the robot's desired destination, the black dot represents the obstacle's boundary, the blue dot indicates the randomly generated endpoint, the blue line signifies the developed branch, and the red line signifies the ultimate planned path. In the comparison of the two figures in Figure 5, we can see the comparison of the planned route of the robot when there are obstacles and when there are no obstacles. The path of the robot is not optimal when there are no obstacles. When there are obstacles, the robot can quickly get the general direction of action. After smoothing the path curve, it can be used as an influence factor of the DWA evaluation function. The approximate direction of the path obtained quickly can reduce the range of the simulated way in DWA algorithm.

DOI: 10.54254/2755-2721/79/20241597

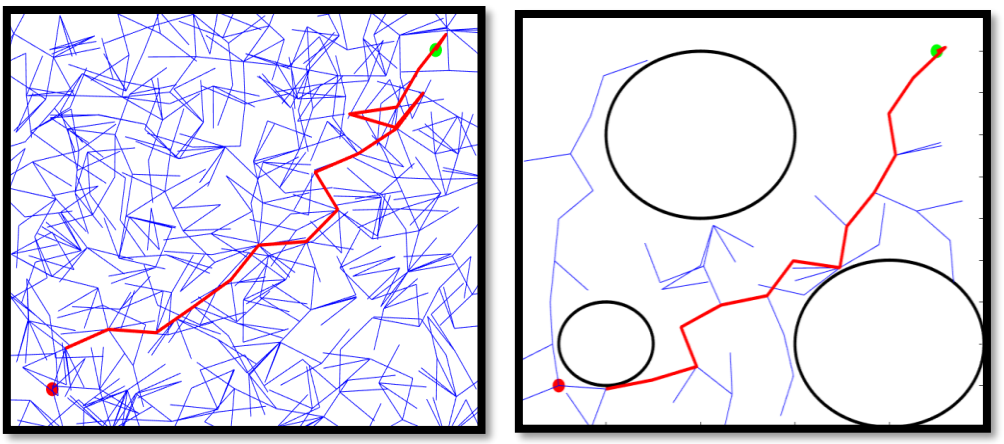

Figure 5. RRT algorithm simulation results. The black circle represents the hindrance, the blue line signifies the connection between arbitrary points, the red line signifies the intended route, the red point signifies the commencement and the green point signifies the conclusion.

4.3. Dynamic window Approach

Figure 6 shows the simulation of the DWA algorithm. The red circle in the figure symbolizes the hindrance, the blue line signifies the trajectory of motion, and the green region signifies the simulated path. By comparing different barriers, it can be observed that when the obstacles are dense and complex, the robot will fall into a dead end and cannot get rid of them.

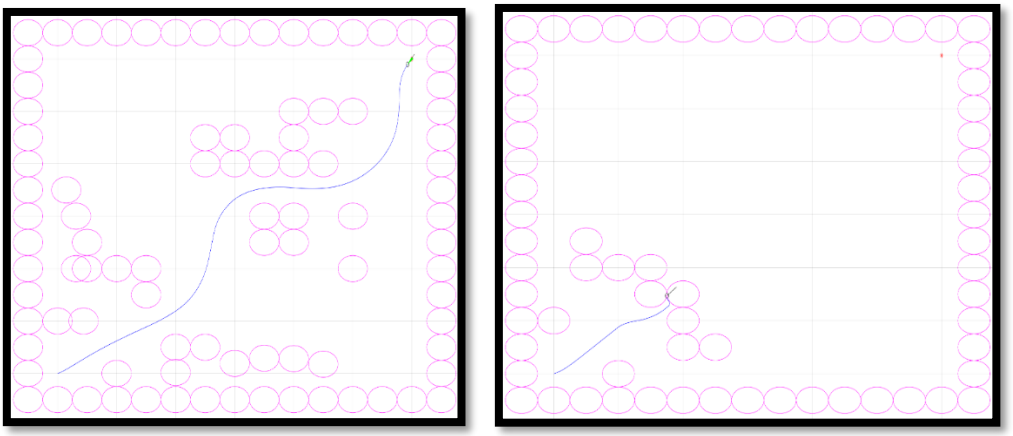

Figure 6. DWA algorithm simulation results. The red circles are obstacles and the blue line is the final path.

However, it has excellent responsiveness to moving obstacles and unknown environments by taking the path of the A star algorithm as the impact factor of the evaluation function. The general direction provided by the RRT algorithm can avoid the situation where the robot is trapped in a local dead end. Although this algorithm requires a lot of memory, it will significantly improve the practical effect of DWA. This approach solves the situation where the DWA would be stuck in a local dead end. It also reduces the number of simulated paths generated to improve speed by providing a strategy of general direction.

4.4. Buffered Voronoi Diagram

As shown in Figure 7. Gray squares are obstacles. Different colored circles represent the robot's position. Rectangles of different colors represent robot buffer distances. Lines of different colors represent the

DOI: 10.54254/2755-2721/79/20241597

robot's final movement trajectory. It can be seen that the buffer area changes when the robot is closer to obstacles or other robots. Constrain the robot's predicted trajectory within the buffer area. This approach facilitates the avoidance of obstacles between robots and robots, as well as between robots and obstacles.

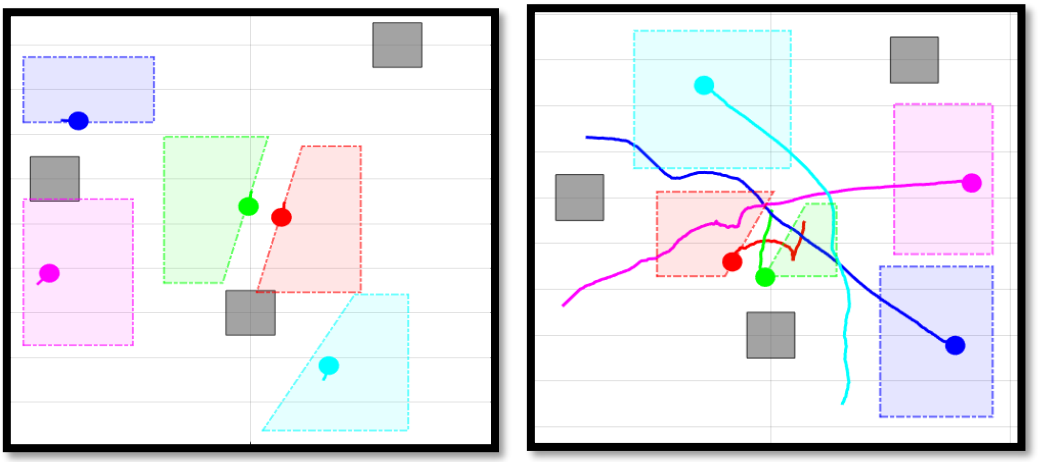

Figure 7. Gray squares are obstacles. Different colored circles represent the robot's position. Rectangles of different colors represent robot buffer distances. Lines of different colors represent the robot's final movement trajectory.

5. Results and Discussion

As shown in Figure 8, the first picture shows the path generated by the RRT algorithm and the results of the improved DWA algorithm. In the figure, the dark blue line is the path forged by RRT, and the red line is the final planned path of RRT. Light blue is the robot's actual running path. Without departing from the main course of RRT planning, it has good results in both global optimal and local optimal. And benefit from RRT's fast path generation at the beginning of a movement. Robots can run faster when facing complex environments during actual activities. This is reflected in exercise time. When only using the original DWA algorithm, the time to reach the endpoint in this environment is 1 minute and 04 seconds. The running time is 47 seconds, with RRT pre-obtaining the path and factoring it into an evaluation function. Without considering the time required to pre-compute RRT path planning, the efficiency is increased by approximately 36%. The effect of this improvement is pronounced in actual use.

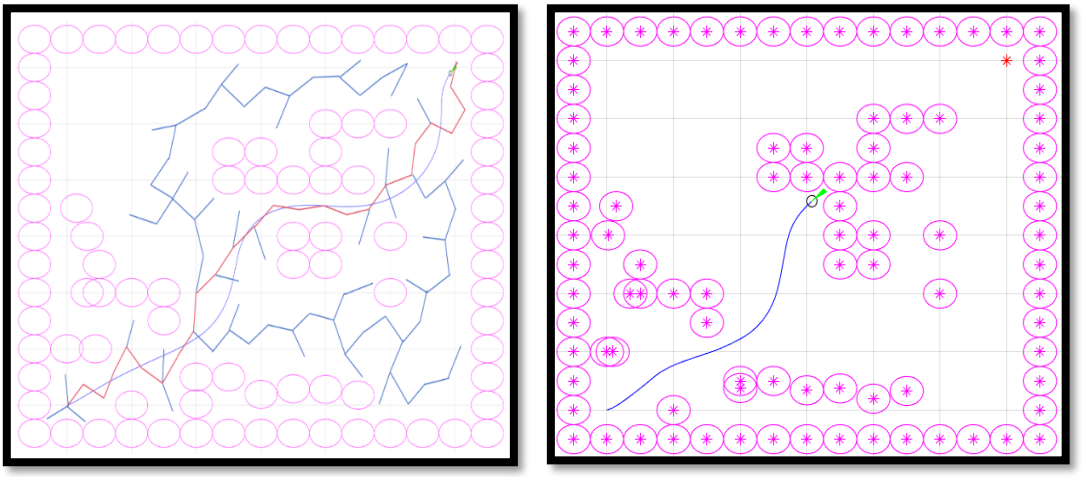

Figure 8. Enhance the outcomes of the DWA algorithm simulation. Obstacles are symbolized by circles, running paths are indicated by blue lines, and paths generated by RRT are represented by red lines.

DOI: 10.54254/2755-2721/79/20241597

6. Conclusion

This paper introduces the development history and prospects of fruit-picking robots. Next, six path-planning algorithms with good results are introduced, and our improved algorithm is proposed. First, the RRT algorithm is used to build a map environment quickly, and the PRM algorithm is used to build a more detailed map environment after reaching a certain threshold. Subsequently, the evaluation function of the DWA algorithm incorporates the path obtained through the A star algorithm, which emphasizes the global shortest path, and the Voronoi graph algorithm, which emphasizes robot safety and enhanced obstacle avoidance efficiency. In practical application, the weights of different influence factors of different robot path-planning evaluation functions can be changed according to the structures and work needed to achieve the robot's safety and efficiency. Finally, our hypothesis is verified by simulation data. The results of the simulation experiment agree with the prediction. Picking robots on the map can efficiently realize path planning and avoid collision problems. This result shows that our proposed path-planning strategy and obstacle avoidance effect are feasible. Simulation experiments preliminarily verify the feasibility of our algorithm in an ideal MATLAB environment. Next, our work is to continue to find the feasibility of this algorithm in practical applications and other influencing factors. Moreover, we improved our algorithm to improve work efficiency. We will continue to improve the path-planning algorithm in the future. At the same time, issues such as global optimality, local optimality, and dynamic obstacle avoidance are considered. Moreover, apply theoretical research to actual agricultural robots to test the results.

References

[1] L. Rosa, D. D. Chiarelli, M. C. Rulli, J. Dell'Angelo, and P. D'Odorico, "Global Agricultural Economic Water Scarcity," Science Advances, vol. 6, no. 18, 2020. doi:10.1126/sciadv.aaz6031

[2] "Artificial Intelligence in agriculture," Data Analytics and Artificial Intelligence, vol. 2, no. 6, 2022. doi:10.46632/daai/2/6/11

[3] I. Charania and X. Li, "Smart farming: Agriculture's shift from a labor intensive to technology native industry," Internet of Things, vol. 9, p. 100142, 2020. doi:10.1016/j.iot.2019.100142

[4] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of Mobile Robots: Concepts, methods, theoretical framework, and applications," International Journal of Advanced Robotic Systems, vol. 16, no. 2, p. 172988141983959, 2019. doi:10.1177/1729881419839596

[5] H. Zhou, X. Wang, W. Au, H. Kang, and C. Chen, "Intelligent robots for fruit harvesting: Recent developments and future challenges," Precision Agriculture, vol. 23, no. 5, pp. 1856–1907, 2022. doi:10.1007/s11119-022-09913-3

[6] P. Li, S. Lee, and H.-Y. Hsu, "Review on fruit harvesting method for potential use of automatic fruit harvesting systems," Procedia Engineering, vol. 23, pp. 351–366, 2011. doi:10.1016/j.proeng.2011.11.2514

[7] W. Mao, Z. Liu, H. Liu, F. Yang, and M. Wang, "Research progress on synergistic technologies of Agricultural Multi-Robots," Applied Sciences, vol. 11, no. 4, p. 1448, 2021. doi:10.3390/app11041448

[8] R. Gai, X. Wang, Z. Chang, and Y. Guo, "Fruit and vegetable picking robot movement planning: A Review," 2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), 2022. doi:10.1109/ithings-greencom-cpscom-smartdata-cybermatics55523.2022.00086

[9] J. Yang, C. Wang, B. Jiang, H. Song, and Q. Meng, "Visual perception enabled industry intelligence: State of the art, challenges and prospects," IEEE Transactions on Industrial Informatics, vol. 17, no. 3, pp. 2204–2219, 2021. doi:10.1109/tii.2020.2998818

DOI: 10.54254/2755-2721/79/20241597

[10] R. Yue et al., "Design and working parameter optimization of pneumatic reciprocating seedling-picking device of automatic transplanter," Agriculture, vol. 12, no. 12, p. 1989, 2022. doi:10.3390/agriculture12121989
[11] L. F. Oliveira, A. P. Moreira, and M. F. Silva, "Advances in agriculture robotics: A state-of-the-art review and Challenges ahead," Robotics, vol. 10, no. 2, p. 52, 2021. doi:10.3390/robotics10020052
[12] X. Cao, X. Zou, C. Jia, M. Chen, and Z. Zeng, "RRT-based path planning for an intelligent litchi-picking manipulator," Computers and Electronics in Agriculture, vol. 156, pp. 105–118, 2019. doi:10.1016/j.compag.2018.10.031
[13] S. G. Vougioukas, "Agricultural Robotics," Annual Review of Control, Robotics, and Autonomous Systems, vol. 2, no. 1, pp. 365–392, 2019. doi:10.1146/annurev-control-053018-023617
[14] J. C. Dagar, G. W. Sileshi, and F. K. Akinnifesi, "Agroforestry to enhance livelihood security in Africa: Research trends and emerging challenges," Agroforestry for Degraded Landscapes, pp. 71–134, 2020. doi:10.1007/978-981-15-4136-0_3
[15] A. Bagheri, N. Emami, M. S. Allahyari, and C. A. Damalas, "Pesticide handling practices, health risks, and determinants of safety behavior among Iranian Apple Farmers," Human and Ecological Risk Assessment: An International Journal, vol. 24, no. 8, pp. 2209–2223, 2018. doi:10.1080/10807039.2018.1443265
[16] A. Fitriansyah, N. W. Parwati, D. R. Wardhani, and N. Kustian, "Dijkstra's algorithm to find shortest path of tourist destination in Bali," Journal of Physics: Conference Series, vol. 1338, no. 1, p. 012044, 2019. doi:10.1088/1742-6596/1338/1/012044
[17] Y. D. Rosita, E. E. Rosyida, and M. A. Rudiyanto, "Implementation of dijkstra algorithm and multi-criteria decision-making for Optimal Route Distribution," Procedia Computer Science, vol. 161, pp. 378–385, 2019. doi:10.1016/j.procs.2019.11.136
[18] A. T. Ameen et al., "Comparison of parallel and serial execution of shortest path algorithms," 2023 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 2023. doi:10.1109/iccike58312.2023.10131705
[19] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968. doi:10.1109/tssc.1968.300136
[20] I. Noreen, A. Khan, and Z. Habib, "Optimal path planning using RRT* based approaches: A survey and Future Directions," International Journal of Advanced Computer Science and Applications, vol. 7, no. 11, 2016. doi:10.14569/ijacsa.2016.071114
[21] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), 2000. doi:10.1109/robot.2000.844107
[22] A. Mackovová and P. Chalmovianský, "Regularity conditions for Voronoi diagrams in hyperbolic space," Lecture Notes on Data Engineering and Communications Technologies, pp. 150–162, 2022. doi:10.1007/978-3-031-13588-0_13
[23] H. Huang, Y. Kang, X. Wang, and Y. Zhang, "Multi-robot collision avoidance based on buffered Voronoi diagram," 2022 International Conference on Machine Learning and Knowledge Engineering (MLKE), 2022. doi:10.1109/mlke55170.2022.00051
[24] E. Stach et al., "Autonomous Experimentation Systems for Materials Development: A Community Perspective," Matter, vol. 4, no. 9, pp. 2702–2726, 2021. doi:10.1016/j.matt.2021.06.036
[25] J. Chen and P. Dames, The convex uncertain Voronoi diagram for safe multi-robot multi-target tracking under localization uncertainty, 2022. doi:10.21203/rs.3.rs-1530901/v1
[26] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of Mobile Robots: Concepts, methods, theoretical framework, and applications," International Journal of Advanced Robotic Systems, vol. 16, no. 2, p. 172988141983959, 2019. doi:10.1177/1729881419839596

DOI: 10.54254/2755-2721/79/20241597

[27] G. Kyprianou, L. Doitsidis, and S. A. Chatzichristofis, “Towards the achievement of path planning with multi-robot systems in Dynamic Environments,” Journal of Intelligent & Robotic Systems, vol. 104, no. 1, 2021. doi:10.1007/s10846-021-01555-3

[28] G. Tang, C. Tang, C. Claramunt, X. Hu, and P. Zhou, “Geometric A-star algorithm: An improved A-star algorithm for AGV Path Planning in a port environment,” IEEE Access, vol. 9, pp. 59196–59210, 2021. doi:10.1109/access.2021.3070054

[29] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, “Anytime motion planning using the RRT*,” 2011 IEEE International Conference on Robotics and Automation, 2011. doi:10.1109/icra.2011.5980479

[30] Z. Lin, M. Yue, G. Chen, and J. Sun, “Path planning of Mobile Robot with PSO-based APF and Fuzzy-based DWA subject to moving obstacles,” Transactions of the Institute of Measurement and Control, vol. 44, no. 1, pp. 121–132, 2021. doi:10.1177/01423312211024798