Path Planning Based on Mixed Algorithm of RRT and Artificial Potential Field Method

Shunyu Huang
School of Artificial Intelligence and Automation,
Huazhong University of Science & Technology,
Wuhan, Hubei 430074, China
*Corresponding author.
2330572994@qq.com

2021 4th International Conference on Intelligent Robotics and Control Engineering (IRCE) | 978-1-6654-1348-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/IRCE53649.2021.9570910

***Abstract*—Aiming at the problems of random sampling and low efficiency of the path planning algorithm of rapidly-exploring random trees (RRT), an improved algorithm combined with the artificial potential field method (APF) is proposed. This method first introduces a probability value in the expansion step of the random tree in the basic RRT algorithm to speed up the convergence of the random tree to the target node and adds a gravitational component to the random tree to guide the tree to grow towards the target point to speed up the search process. Establish a repulsion field around obstacles to limit the search area between obstacles and reduce the randomness of the path. In this research, in addition to the RRT algorithm, the RRT* algorithm will also be used for improvement. The simulation experiment results show that the proposed method is significantly optimized in time, path length and the number of iterations.**

Keywords—path planning, RRT algorithm, RRT* algorithm, artificial potential field

I. INTRODUCTION

Path planning can be divided into global and local path planning. Global path planning is based on the more comprehensive and macro perspective and in accordance with specific rules. Find a path that can avoid all obstacles by obtaining the distribution information of all obstacles on the map. Task planning and decision-making play an essential role and are inseparable from and complementary to global path planning. Local path planning is based on a more one-sided and microscopic perspective. It performs a limited range of local searches around robots or unmanned vehicles to find a good path and the location of obstacles in the map. When a change occurs, the location information of the obstacle can be grasped by the sensor, and the movement of the obstacle can be judged to avoid the obstacle and find a feasible path [1]. Therefore, from the perspective of whether obstacles are moving, local path planning belongs to dynamic path planning, and global path planning belongs to static path planning.

Nowadays, path planning algorithms are developing rapidly and vigorously due to the rapid development of unmanned vehicles and autonomous mobile robots. A large number of new and improved algorithms have emerged. Path planning algorithms can be divided into the following categories: (1) Path planning algorithms based on bionics. (2) Path planning algorithms based on maps. (3) Path planning algorithms based on sampling. In this research, the third category of path planning algorithm will be used [2-5].

A. Related Work

The mainstream sampling-based algorithms include the RRT algorithm and the Probabilistic Roadmap Method (PRM). Only the RRT algorithm is mentioned. The RRT algorithm is proposed by Steven. M. LaValle [6]. Its most significant advantage is that there is no need to model the planning space. Therefore, it is also a random sampling algorithm. However, the basic RRT algorithm also has several shortcomings when planning the path: (1) Since it is a random sampling algorithm, the search path is random and biased. (2) The random tree is not oriented when searching the path. (3) The convergence rate is tardy, resulting in low search efficiency.

Many studies have improved the shortcomings of the basic RRT algorithm, and therefore many RRT series algorithms have appeared. The RRT-Connect algorithm was proposed by LaValle and Kuffner [7], which improves the search speed by generating a random tree from the starting node and the goal node, respectively. Considering the complexity of the environment, Melchior N A is committed to improving the search efficiency of random trees in complex environments and thus proposes a particle RRT algorithm [8]. Varol and Adiyatov proposed the RRT* algorithm, which optimizes the path's non-optimal problem generated by the original algorithm [9]. In addition, many improved RRT algorithms continue to appear, for example, Dynamic-RRT algorithm, Extend-RRT algorithm, Fast Marching Trees method (FMT*), and Batch Informed Trees method (BIT*) [10-13].

Song Jinze et al. developed an improved RRT algorithm by combining the two-way multi-step search RRT algorithm with non-holonomic constraints in terms of combining improved algorithms. This algorithm improves the search efficiency of the basic RRT algorithm and makes the quality of the planned route better. Song Xiaolin and others improved the RRT algorithm by using Gaussian distribution to describe random sampling points and introducing a heuristic search mechanism, the quality of the actual path planning of the search tree in the local path planning of automobile obstacle avoidance is improved, and the path planning time is shortened. Lu Yafei et al. proposed a UAV penetration path planning algorithm based on artificial potential field and RRT algorithm. This algorithm makes it possible to distinguish between the characteristics of threats and obstacles in the penetrating algorithm [14-16].

Nowadays, although various path planning methods for robots are emerging one after another, many path planning algorithms are applied in a static environment, and there is room for improvement in dynamic path planning algorithms.

In summary, path planning and unmanned driving technology are inseparable. Many researchers have proposed many planning algorithms and achieved specific results to make path planning more accurate and rapid. In addition, different algorithms have different characteristics, and their individual characteristics or advantages can be obtained by combining algorithms. Multi-algorithm combination effectively avoids the disadvantages of each algorithm and improves the efficiency of algorithm planning. Therefore, it is necessary to adjust the algorithm in time for different environments and conditions to improve the efficiency of path planning.

B. Research Content

The main research contents are as follows:

(1) Model the basic algorithm through MATLAB, analyse and determine their shortcomings.

(2) Aiming at the algorithm's shortcomings, using the concept of Goal-Biased algorithm, adding probability value p to the basic algorithm for improvement, analyse the simulation results.

(3) Use the APF method to improve the algorithm again, analyse and compare with other algorithms.

II. BACKGROUND

A. Basic RRT Algorithm

The algorithm uses a random expansion method to construct a tree from the initial node to the target node in the search space. The starting point q_{rand} in the state space is used as the root node, and a point q_{rand} is randomly selected from the search space. If this point falls in the non-obstacle interval, it will traverse T to find the nearest node n to this random point. If the line between q_{rand} and $q_{nearest}$ does not collide with obstacles, and the distance between q_{rand} and $q_{nearest}$ is less than the set expansion step, q_{rand} will be added to the random tree as a leaf node q_{new}. If the distance between q_{rand} and $q_{nearest}$ is longer than the set expansion step length, then on the line between q_{rand} and $q_{nearest}$, take the point from the expansion step length of $q_{nearest}$ as q_{new} and add it to the random expansion tree, then the node closest to q_{new} is called the parent node of q_{new}. Repeat the above iterative process until the target node becomes a leaf node or the search ends when the set number of iterations is exceeded. From the target node back to the starting point, the planned path can be obtained. Table 1 shows the pseudo-code of the algorithm search process, and the node expansion process is shown in Figure 1.

TABLE I: BASIC RRT ALGORITHM.

1.	*Tree* ← *Initializetree*( );
2.	*Tree* ← *InsertNode*(∅,q_{init},*Tree*);
3.	*For i=0 to i=n do*
4.	q_{rand} ← *Sample* (*i*);
5.	$q_{nearest}$ ← *Nearest*(*Tree*, q_{rand})
6.	(q_{new}, U_{new}) ← *Steer*($q_{nearest}$, q_{rand})
7.	*If Obstaclefree* (q_{new}) *then*
8.	*Tree* ← *InsertNode*(q_{min}, q_{new},*Tree*);
9.	*End if*
10.	*End for*
11.	*End Tree*

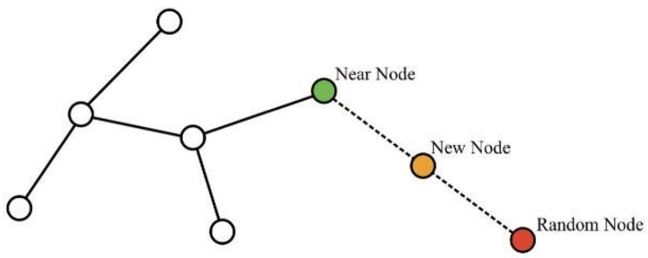


Fig. 1. Node expansion process.

B. RRT Algorithm*

The RRT* algorithm adds two improvements based on the traditional RRT algorithm, called search in adjacent space and reconnection of the tree. The search of the adjacent space means that when the parent node is selected, the new node will use all nodes in the circular space of the predefined radius r as the parent node, calculate the distance required to reach the new node, and take the node with the smallest distance as the parent node of the new node. Tree reconnection refers to using the newly generated node as the parent node to calculate the distance to other nodes within the circle. If the new distance value is short than the original distance to the node, the newly generated node will be used as the parent node of the node reconnect. The pseudo-code of this algorithm search process is shown in the table below.

TABLE II: RRT* ALGORITHM.

1.	*Tree* ← *Initializetree*( );
2.	*Tree* ← *InsertNode*(∅,q_{init},*Tree*);
3.	*For i=0 to i=n do*
4.	q_{rand} ← *Sample* (*i*);
5.	$q_{nearest}$ ← *Nearest*(*Tree*, q_{rand})
6.	(q_{new}, U_{new}) ← *Steer*($q_{nearest}$, q_{rand})
7.	*If Obstaclefree* (q_{new}) *then*
	q_{near} ← *Near*(*Tree*, q_{new},|*Tree*|)
	q_{min} ← *Chooseparent*(q_{near}, $q_{nearest}$, q_{new});
8.	*Tree* ← *InsertNode*(q_{min}, q_{new},*Tree*);
	Tree ← *Review*(*T*, q_{near}, q_{min}, *Tree*)
9.	*End if*
10.	*End for*
11.	*End Tree*

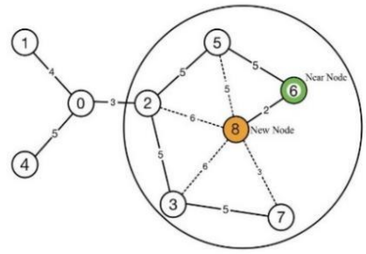


Fig. 2. The addition of new nodes in the RRT* algorithm.

Figure 2 generates a random node 8, the distance from node 6 is 2. Compared with other nodes in the extended tree, the distance to node 6 is the closest, and the connection with node six will not collide with obstacles. Therefore, node eight is added to the expanded tree as a new node. Compared to the basic RRT algorithm, this algorithm increases the search in the adjacent space when selecting the parent node. Draw a circle with node eight as the centre and r is the radius. The node with the closest distance from the starting point to node eight is searched among the nodes within the circle.

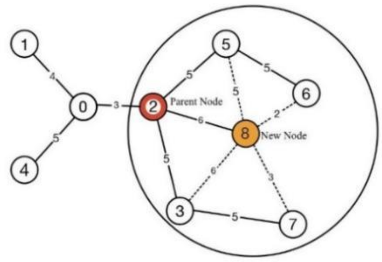


Fig. 3. Search in adjacent space.

As shown in Figure 3, the circle includes nodes 2, 3, 5, 6, and 7, and the initial expansion path of node 8 is 0→2→5→6→8, and the length of the path is 15. But it can be found that path 0→2→8 to node eight is the shortest. Therefore, reconnect node eight and no longer use node six as the parent node, but node two as the parent node.

In addition, the RRT* algorithm adds the feature of tree reconnection. After determining the parent node of node 8, the other nodes 3, 5, 6, 7 in the circle can take node eight as the parent node. Using node 3, 5, 6, and 7 as the parent node, respectively, the path length is calculated, and it is found that when node 6 has node eight as the parent node, the path length is 11, which is less than the path length 13 with node five as the parent node. Similarly, when node 7 takes node eight as the parent node, its path length is 12, less than the path length 13, with node three as the parent node. Therefore, node eight is the parent node of nodes 6 and 7 to reconnect the extended tree. The process is shown in Figure 4.

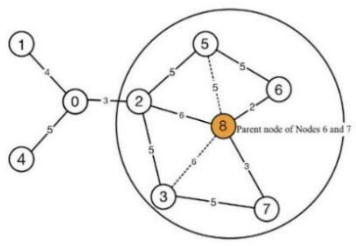


Fig. 4. Reconnection of RRT* algorithm tree.

C. Goal-Biased RRT Algorithm

In order to speed up the search, a search algorithm based on probability appears. Firstly, the goal-biased RRT algorithm presents a target partial probability threshold named p_{target}, then obtain a random probability value p between 0 and 1. When $p > p_{target}$, a random state named q_{rand} is obtained in the search space, otherwise q_{rand} is equal to q_{target}. This algorithm maintains the characteristics of the original algorithm and accelerates the convergence speed to the target node. Table 3 shows the pseudo-code of the algorithm. The Extended functions in the table return to steps 5 to 11 in Table 2.

TABLE III: GOAL-BIASED RRT ALGORITHM.

1:	*Tree* ← *InsertNode*(∅,q_{init},*Tree*);
2:	*for i=0 to i=n do*
3:	*if* $p>p_{target}$ *then*
4:	q_{rand} ← *RandomState()*;
5:	*else*
6:	$q_{rand} = q_{goal}$;
7:	*Extended* (*Tree*,q_{rand});
8:	*Return Tree*

D. Artificial Potential Field Method (APF)

The APF is a local path planning algorithm. Its method is to regard the robot motion space as a box of artificial force fields; that is, the target node has a gravitational component to the robot in the potential field, and the obstacle is in the potential field. There is a repulsive force component to the robot. Therefore, the robot moves under the combined force of the target gravitational component and the obstacle repulsion component. In the traditional APF method, the robot x is assumed to move under the combined force of the target gravitational component and the obstacle repulsion component. Its total potential field U(x) is expressed as:

$$U(x) = U_{target}(x) + U_{obstacle}(x) \qquad (1)$$

In Equation 1, $U_{target}(x)$ represents the gravitational field of the target point on the robot x, and $U_{obstacle}(x)$ represents the repulsion field of the obstacle on the robot x. The potential field force is expressed as:

$$F(x) = F_{target}(x) + F_{obstacle}(x) \qquad (2)$$

In Equation 2, $F_{target}(x)$ represents the gravitational force of the target node, $F_{obstacle}(x)$ represents the repulsion force of the obstacle on the robot x, and $F(x)$ represents the resultant force.

Artificial potential field has several shortcomings in path planning: (1) When there are many obstacles in the movement space, the planning success rate will decrease. (2) When the distance between the obstacle and the target is less than a certain threshold, it is easy to oscillate and make it impossible to reach the target point. (3) When $F(x) = 0$, it will fall into a local minimum and cause a local infinite loop. Therefore, an improved RRT algorithm combined with APF is proposed.

E. Improved RRT Algorithm

1) Introducing the gravitational component

The concept of APF is introduced into the Goal-Biased algorithm to increase the gravitational component of the random node and control the random tree to grow toward the target point to reduce the randomness. The schematic diagram is shown in Figure 5. The core idea is to introduce a gravitational component to each new node generated. In this study, a gravitational function $G(n)$ is introduced in each new node n to change the resultant force of the growth direction. It can be expressed as:

$$F(n) = Rd(n) + G(n) \qquad (3)$$

In Equation 3, $F(n)$ represents the function of the random tree to determine the new node in the search process, $Rd(n)$ is the random growth function of the new node, and $G(n)$ is the increased target gravitational function.

The target gravitational function $G(n)$:

$$G(n) = \rho \times g \times \frac{x_{goal} - x_{near}}{|x_{goal} - x_{near}|} \qquad (4)$$

In Equation 4, ρ is the step size, which can be adjusted by programming, g is the gravitational gain coefficient, x_{goal} is the target position vector, and $|x_{goal} - x_{near}|$ represents the absolute value of the geometric distance between node x_{goal} and x_{near}.

The random growth function of the new node:

$$Rd(n) = \rho \times \frac{x_{goal} - x_{near}}{|x_{goal} - x_{near}|} \qquad (5)$$

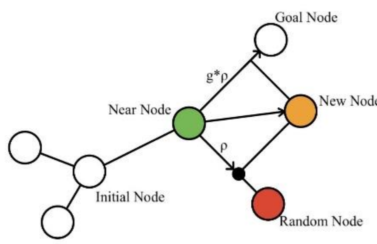


Fig. 5. Constructing RRT by increasing gravitational component.

2) Introduce the repulsion component

The concept of obstacle repulsion in APF is introduced into the Goal-Biased RRT algorithm to guide the local random tree to grow away from the obstacle. The expansion diagram of the RRT algorithm to increase the repulsion component is shown in Figure 6. The core idea is to introduce a repulsion component to each new node generated during the expansion of the local random tree search. In this study, an obstacle repulsion function $T(n)$ is introduced at the new node n generated, which can be expressed as:

$$F(n) = Rd(n) + T(n) \tag{6}$$

In Equation 6, $F(n)$ represents the function of the random tree to determine the new node in the search process, $Rd(n)$ is the random growth function of the new node, and $T(n)$ is the increased obstacle repulsion function.

The obstacle repulsion function $T(n)$:

$$T(n) = \begin{cases} 0, & p(x) > p_0 \\ \rho k_{rep}\left(\frac{1}{p(x)} - \frac{1}{p_0}\right)\frac{1}{p(x)^2}\frac{\partial(x_{near} - x_{obstacle})}{\partial x_{near}}, & p(x) \le p_0 \end{cases} \tag{7}$$

In Equation 7, k_{rep} is the repulsive force gain coefficient, $p(x)$ is the shortest distance from the random node to the obstacle, p_0 is the influence distance of the obstacle on the node, and $x_{obstacle}$ is the position vector of the obstacle.

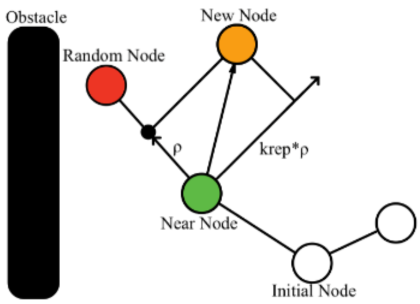


Fig. 6. Constructing RRT by increasing repulsion component.

III. SIMULATION EXPERIMENT ANALYSIS

To confirm whether the improved algorithm can search for target points and generate paths more efficiently at the starting point in the space, it is necessary to build simulation experiments on the improved RRT algorithm. Use Matlab to build the simulation environment and complete the simulation experiment of the improved RRT algorithm. The simulation results and basic algorithm simulation results are simultaneously compared and analysed with the Goal-Biased RRT improved algorithm.

In this study, only the situation in a two-dimensional space is considered, and the size of the simulation map is 1000*1000. To compare the results, the simulation parameters of each algorithm are unified. The specific parameters are as follows: the starting node is (0, 0), the target node is (999, 999), the starting point and target point are marked in pink and blue, respectively, and the step length is 30.

A. Simulation Results and Analysis of Basic Algorithm

Figure 7 and Figure 8 are the simulation diagrams of the basic RRT algorithm and RRT* algorithm, respectively.

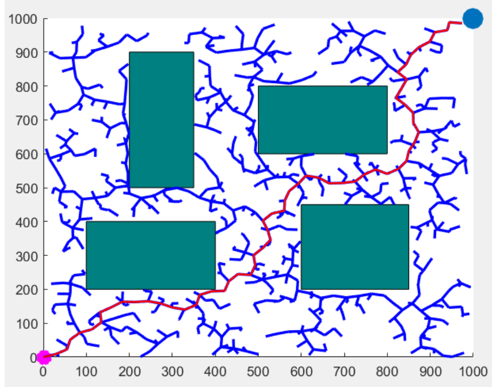


Fig. 7. Experimental result of basic RRT algorithm.

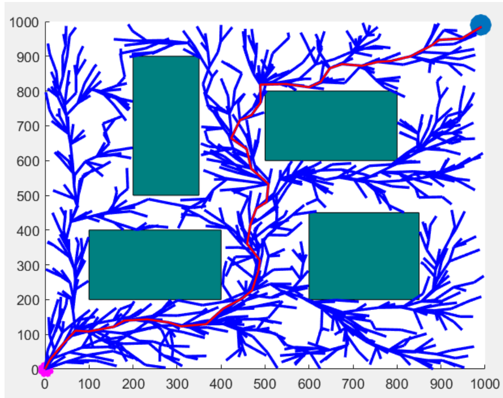


Fig. 8. Experimental result of RRT* algorithm.

In the two figures, blue represents the generated random tree, and red represents the feasible path generated by the algorithm in two-dimensional space. It can be seen from these two figures that the basic RRT algorithm and basic RRT* algorithm have no guiding line in the search process, and the path is randomly generated. The random tree performs a global search in the space, and the random generation of nodes is biased.

B. Simulation Results and Analysis of Goal-Biased RRT Algorithm and Goal-Biased RRT Algorithm*

The core concept of the Goal-Biased algorithm is used to improve the basic algorithm for the first time. Figure 9 and Figure 10 are the simulation diagrams of the Goal-Bias RRT algorithm and the Goal-Biased RRT* algorithm, respectively. In this study, for all the improved algorithms of Goal-Biased, the target partial probability threshold p_{target} is 0.4.

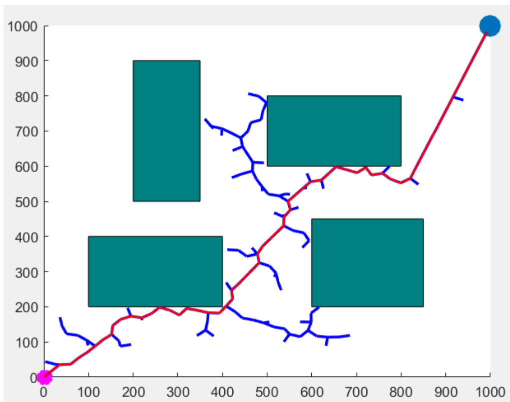


Fig. 9. Experimental result of Goal-Bias RRT algorithm.

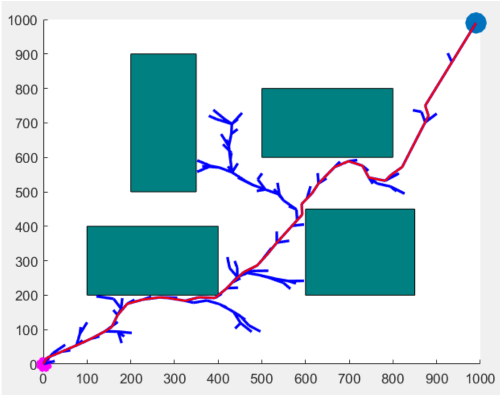


Fig. 10. Experimental result of Goal-Bias RRT* algorithm.

Analysing Figure 9 and Figure 10, the Goal-Biased RRT algorithm eliminates useless search areas far away from the target area by introducing probability values and guides the random tree to grow toward the target point as much as possible. Dramatically reduces the number of iterations and nodes of the basic RRT algorithm. However, the resulting red path is still random.

C. Simulation Results and Analysis of Improved Algorithm Based on Goal-Bias Algorithm and APF

Figure 11 and Figure 12 are the simulation diagrams of the APF RRT improved algorithm and APF RRT* improved algorithm, respectively.

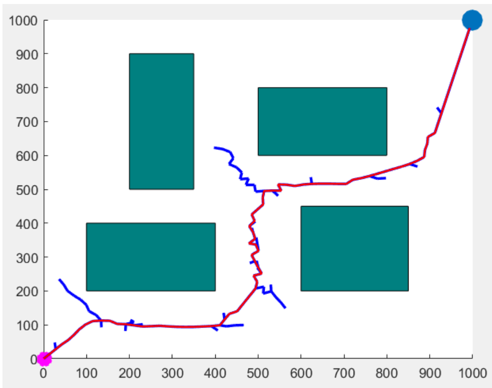


Fig. 11. Experimental result of APF RRT improved algorithm.

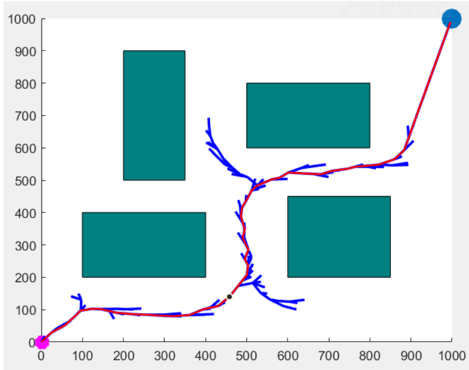


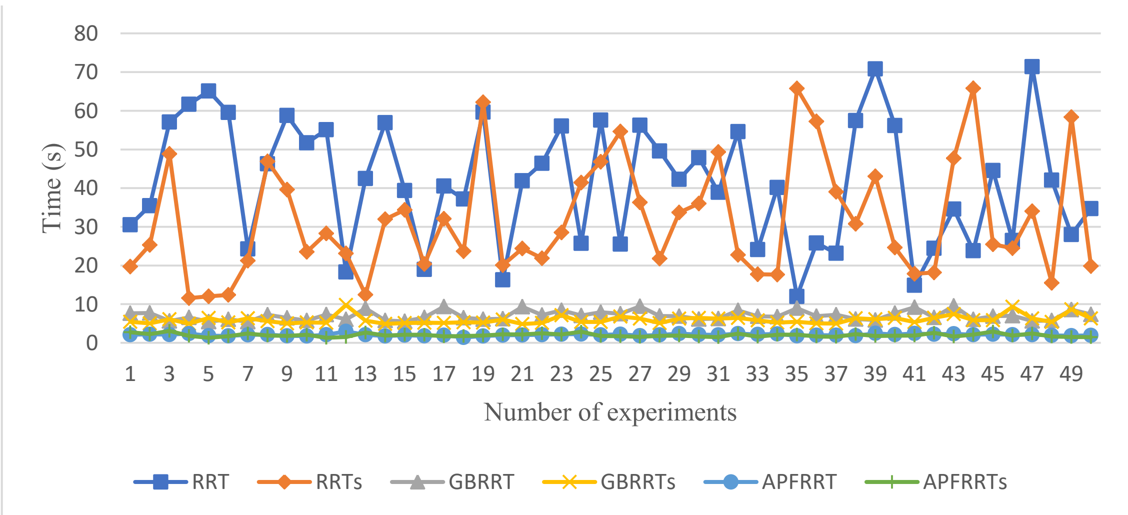
Fig. 12. Experimental result of APF RRT* improved algorithm.

By introducing an appropriate gravitational coefficient, the path can be more confined to the area connecting the starting node and the target node, which reduces the search area compared to the Goal-Bias RRT algorithm. Due to the existence of the obstacle repulsion field, it can be regarded as a specific range of invisible obstacles around the visible obstacles, which significantly limits the search area of the path between the obstacles. Therefore, it can be found that the path between the obstacles is almost a straight line, which can reduce the randomness of the path search and keep the searched path at a proper distance from the obstacle.

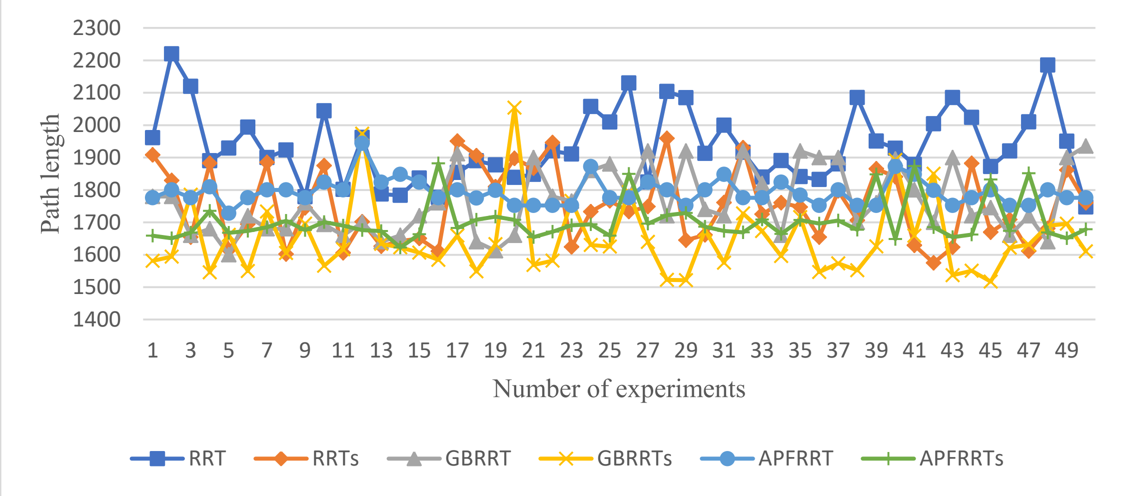
D. Data Analysis of Simulation Results

In this research, the improved Goal-Biased algorithm, the improved APF RRT algorithm and the basic algorithm compare the number of iterations, search time and path length for 50 simulations on the premise of the same coefficients as shown in Figure 13. The average search time, the average path length and the average number of iterations of the simulation are computed and shown in Table 4.

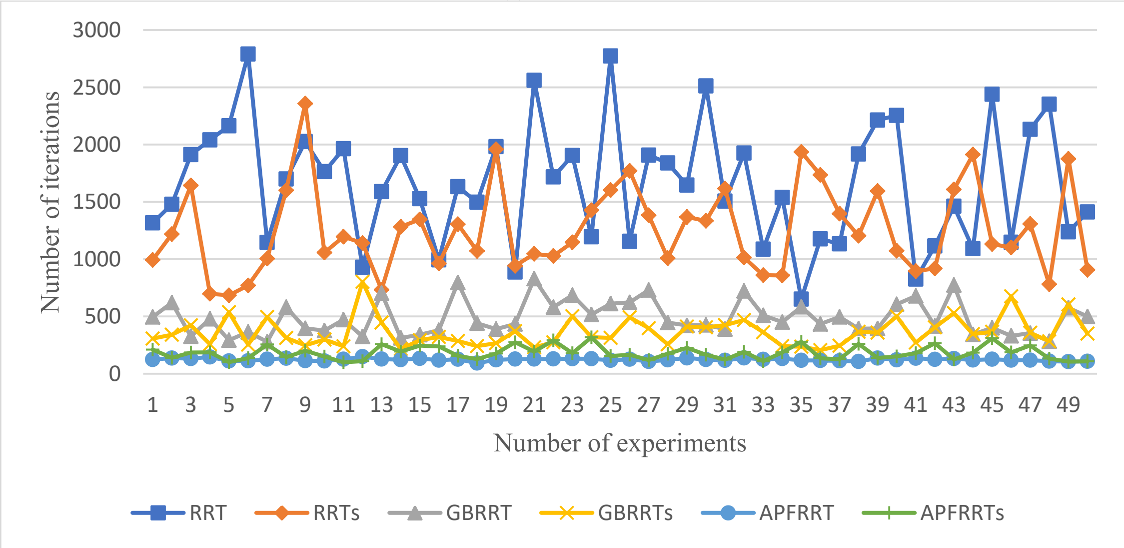
Analyzing Figure 13 and Table 4, the Gaol-Biased improved algorithm significantly reduces the search time of the path search and the number of iterations compared to the basic algorithm. The APF RRT improved algorithm further reduces the number of iterations and the calculation time. In terms of path length, due to the limitation of the repulsive force field, the path length of the APF RRT improved algorithm has not been decreased. Still, from Figure 13(b), it can be seen that the improved algorithm ensures that the same path can be searched more stably, and the path randomness is reduced.



(a)



(b)



(c)

Fig. 13. Data comparison: (a) Time (b) Path length (c) The number of iterations.

TABLE IV. DATA CONTRAST.

Algorithm	Average search time (s)	Average path length	Average iteration
Basic RRT algorithm	41.44	1936.16	1661.28
RRT* algorithm	31.81	1751.31	1256.28
Goal-Bias RRT algorithm	7.05	1763.59	485.98
Goal-Bias RRT* algorithm	5.94	1648.30	362.90
APF RRT algorithm	2.06	1792.58	122.94
APF RRT* algorithm	1.98	1702.46	182.18

Because of the gravitational component of the APF RRT improved algorithm, all new nodes grow toward the target node to avoid covering the entire search area, reducing the running time. Due to the existence of the repulsive force field, the search path can stably appear in the middle of the two obstacles, which can make it easier for unmanned vehicles or self-service robots to obtain a safe path and avoid collisions.

IV. Conclusion

This research combines the improved RRT algorithm with the artificial potential field. Simulation experiments prove that the hybrid algorithm significantly improves the shortcomings of the basic RRT algorithm. It effectively reduces the number of iterations and the running time in the path search process and reduces the randomness and deviation of the path.

A. *Future Development*

This research did not discuss the issue of path optimization. As shown in Figure 11, it can be found that the path between the obstacles oscillates due to the sizeable repulsive component. Therefore, in the future, the path can be optimized to make the path smoother.

References

[1] You F, Wang R, Zhang R, et al. Lane Changing and Overtaking Control Method for Intelligent Vehicle Based on Backstepping Algorithm[J]. Transactions of the Chinese Society for Agricultural Machinery, 2008, 39(6):42-45.

[2] Ji J, et al. "Path Planning and Tracking for Vehicle Collision Avoidance Based on Model Predictive Control with Multiconstraints." IEEE Transactions on Vehicular Technology (2016):1-1.

[3] Cao H, Song X, Huang Z, et al. Simulation research on emergency path planning of an active collision avoidance system combined with longitudinal control for an autonomous vehicle[J]. Proceedings of the Institution of Mechanical Engineers Part D Journal of Automobile Engineering, 2015:1624–1653.

[4] Abbas, M. A, R. Milman, and J. M. Eklund "Obstacle avoidance in real time with Nonlinear Model Predictive Control of autonomous vehicles." Canadian Journal of Electrical and Computer Engineering 40.1(2017):12-22.

[5] Huang Z, Wu Q, Ma J, et al. An APF and MPC combined collaborative driving controller using vehicular communication technologies[J]. Chaos Solitons & Fractals, 2016, 89:232-242.

[6] Lavalle S M. Rapidly-exploring random trees: A new tool for path planning[J]. Computer ence Dept. Oct, 1998, 98.

[7] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), 2000, pp. 995-1001 vol.2, doi: 10.1109/ROBOT.2000.844730.

[8] Melchior N A, Simmons R. Particle RRT for Path Planning with Uncertainty[C]// IEEE International Conference on Robotics & Automation. IEEE, 2007:1617-1624.

[9] Adiyatov O, Varol H A. A novel RRT*-based algorithm for motion planning in Dynamic environments. IEEE, 2017:1416-1421.

[10] D. Ferguson, N. Kalra and A. Stentz, "Replanning with RRTs," Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., 2006, pp. 1243-1248, doi:10.1109/ROBOT.2006.1641879.

[11] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002, pp. 2383-2388 vol.3, doi: 10.1109/IRDS.2002.1041624.

[12] Janson L, Clark A, Pavone M. Fast Marching Tree: a Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions[J]. 2013.

[13] Gammell, J. D, S. S. Srinivasa, and T. D. Barfoot. "Batch Informed Trees (BIT*): Sampling-based Optimal Planning via the Heuristically Guided Search of Implicit Random Geometric Graphs." Proceedings - IEEE International Conference on Robotics and Automation 2015(2015):3067-3074.

[14] Song J Z, Dai B, Shan E Z, et al. An Improved RRT Path Planning Algorithm[J]. Acta Electronica Sinica, 2010.

[15] Song X, Zhou N, Huang Z, et al. An Improved RRT Algorithm of Local Path Planning for Vehicle Collision Avoidance[J]. Journal of Hunan University (Natural Sciences), 2017.

[16] Lu, Yafei W, Anping C, et al. An Improved UAV Path Planning method Based on RRT-APF Hybrid strategy[C]// 2020.