Artificial Potential Field-Guided RRT Path Planning Algorithm in Dynamic Environments: Postprint

Authors: Situ Huajie, Lei Haibo, Zhuang Chungang

Date: 2020-09-28T00:00:00+00:00

Abstract

Most existing dynamic RRT path planning algorithms cannot keep the planned path sufficiently away from obstacles, which may result in insufficient time for the robot to avoid obstacles. To address this issue, we propose an improved RRT algorithm called APFG-RRT (artificial potential field guided RRT) that utilizes artificial potential fields to guide the rapid-exploring random tree to grow toward the goal region while staying away from obstacles. Then, to further accelerate the algorithm' s convergence speed and help it escape local minima, a strategy of selecting the goal point as a sample point with adaptive probability is introduced. Finally, for dynamic environments, a method combining global planning with local replanning is adopted to improve the algorithm's real-time performance. Simulation experiments demonstrate that, compared to the original RRT and Goal-bias RRT, APFG-RRT exhibits higher computational efficiency, lower memory requirements, and the searched path can effectively stay away from obstacles, thereby improving the success rate of dynamic path planning.

Full Text

Preamble

Artificial Potential Field Guided RRT Algorithm for Path Planning in Dynamic Environments

Situ Huajie, Lei Haibo, Zhuang Chungang
(School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: Most existing dynamic RRT path planning algorithms cannot maintain a safe distance between the planned path and obstacles, which may leave

insufficient time for robots to avoid collisions. To address this problem, this paper proposes an improved RRT algorithm called APFG-RRT (Artificial Potential Field Guided RRT) that utilizes artificial potential fields to guide the rapid exploration random tree to grow toward the goal region while staying away from obstacles. To further accelerate convergence and help the algorithm escape local minima, an adaptive probability strategy is introduced to select the goal point as a sampling point. Finally, for dynamic environments, a method combining global planning with local replanning is adopted to improve real-time performance. Simulation experiments demonstrate that compared with the original RRT and Goal-bias RRT, APFG-RRT achieves higher computational efficiency, lower memory requirements, and produces paths that effectively stay away from obstacles, thereby improving the success rate of dynamic path planning.

Keywords: path planning; RRT; artificial potential fields; dynamic environments; local replanning

0 Introduction

Path planning is the process of finding a collision-free path from an initial position to a target position in a given environment, with widespread applications in robotics systems [**?**], medicine [**?**], computer animation [**?**], modern industry, and many other fields. Depending on the level of environmental information available, path planning can be categorized into global path planning and local path planning. Local path planning algorithms include neural networks [**?**], artificial potential field methods [**?**], and genetic algorithms [**?**], among which the artificial potential field method has attracted extensive attention due to its simplicity, good real-time performance, and smooth planned paths. However, the artificial potential field method may fall into local minima during path planning and is prone to oscillation near the target point, which significantly limits its practical application. Global path planning methods include ant colony algorithms [**?**], RRT, visibility graph methods [**?**], etc., where probabilistic roadmap methods [**?**] and RRT explore the configuration space through random sampling, offering high computational efficiency and probabilistic completeness—meaning that as planning time approaches infinity, the probability of finding a feasible path approaches 1.

To further improve the convergence speed of RRT, many researchers have proposed enhancements. Kalisiak et al. introduced RRT-blossom [**?**], which uses regression constraint functions to generate new nodes, reducing the probability of the random tree repeatedly exploring the same region in early stages. Shkolnik et al. developed Ball Tree [**?**], replacing the original volume-less nodes with hyperspheres and rejecting sampling points within nodes to focus the random tree on unexplored areas. Lavalle et al. proposed Bidirectional RRT [**?**], which accelerates exploration by simultaneously expanding random trees from both the initial and target positions. Subsequently, Lavalle also proposed Goal-bias

RRT [**?**], which uses target point information to bias random tree growth toward the goal. Zhou Fang et al. introduced a greedy growth strategy [**?**], employing the largest possible step size at each growth step to quickly cover more area. However, none of these algorithms utilize environmental information.

To incorporate environmental information into RRT guidance, some researchers have explored combining RRT with artificial potential fields. References [**?**, **?**] combine the two algorithms through switching: when the artificial potential field method falls into local minima, it switches to RRT planning, and switches back when escaping local minima. References [**?**, **?**] propose improved RRT* algorithms that offset random sampling points toward the goal direction, where the offset magnitude relates to the minimum distance between the sampling point and obstacles, implicitly introducing repulsive force magnitude. Reference [**?**] extends the algorithm from [**?**] to B-RRT. *Since these algorithms are based on RRT*, they focus on searching for optimal paths, suffer from poor real-time performance, and are unsuitable for dynamic path planning with high real-time requirements. Moreover, these algorithms are not directly transferable to dynamic RRT algorithms that only need to plan a path, because they utilize the magnitude of attractive and repulsive forces to offset sampling points. When the random tree has few nodes, sampling points are generally far from the node to be grown, and the potential field around the sampling point cannot represent the environmental information around the node to be grown. The guiding effect of repulsive force is minimal, only becoming significant when random tree nodes are densely distributed in the search space. Furthermore, these algorithms do not incorporate the direction of repulsive force, which only slows random tree growth toward obstacles but cannot make the tree grow away from obstacles. This may cause planned paths to closely follow dynamic obstacle surfaces, creating collision risks. Additionally, these algorithms use fixed step sizes that cannot adapt to different local environments.

To overcome these problems, this paper proposes an improved dynamic path planning algorithm called APFG-RRT, which uses potential fields to guide node growth. By constructing attractive and repulsive potential fields, the random tree is made to approach the goal while staying away from obstacles, with adaptive step sizes for different local environments. Simultaneously, an adaptive probability sampling bias growth strategy is proposed, which significantly reduces repeated calculations in local minima regions. For dynamic environments, a local replanning strategy is employed. Simulation experiments demonstrate that compared with the original RRT and Goal-bias RRT, APFG-RRT achieves higher computational efficiency and fewer nodes in various environments, including narrow and dynamic scenarios.

1 Artificial Potential Field Method

The artificial potential field method, proposed by Khatib, is a virtual force approach that establishes a virtual potential field with minimum potential energy in the target region around the robot. It uses gradient descent to guide the robot to the target region. Specifically, an attractive potential field is constructed in the target region and a repulsive potential field in obstacle regions, causing the robot to be attracted to the target while being repelled by obstacles. The attractive potential function and attractive force function are typically:

$$U_{att}(x) = \begin{cases} \frac{1}{2}K_a d^2(x, x_{goal}) & d(x, x_{goal}) > d_g^* \\ K_a d_g^* d(x, x_{goal}) - \frac{1}{2}K_a d_g^{*2} & d(x, x_{goal}) \leq d_g^* \end{cases}$$

$$F_{att}(x) = \begin{cases} K_a(x_{goal} - x) & d(x, x_{goal}) > d_g^* \\ K_a d_g^* \frac{x_{goal}-x}{d(x, x_{goal})} & d(x, x_{goal}) \leq d_g^* \end{cases}$$

where K_a is the attractive constant, x and x_{goal} are the robot control point position and target position respectively, $d(x, x_{goal})$ represents the Euclidean distance between the two positions, and d_g^* is a set distance threshold. When the distance between the robot control point and target point exceeds d_g^*, the attractive potential energy is proportional to the square of the distance, enabling rapid approach to the target. When the distance is less than d_g^*, the attractive potential field forms a cone centered at the target, mitigating oscillation near the target.

The repulsive potential function and repulsive force function are:

$$U_{rep}(x) = \begin{cases} \frac{1}{2}K_r \left(\frac{1}{d(x, x_{obs})} - \frac{1}{d_{obs}^*} \right)^2 & d(x, x_{obs}) \leq d_{obs}^* \\ 0 & d(x, x_{obs}) > d_{obs}^* \end{cases}$$

$$F_{rep}(x) = \begin{cases} K_r \left(\frac{1}{d(x, x_{obs})} - \frac{1}{d_{obs}^*} \right) \frac{1}{d^2(x, x_{obs})} \frac{\partial d(x, x_{obs})}{\partial x} & d(x, x_{obs}) \leq d_{obs}^* \\ 0 & d(x, x_{obs}) > d_{obs}^* \end{cases}$$

where K_r is the repulsive constant, x_{obs} is the position of the obstacle' s closest point, d_{obs}^* is the repulsive potential field' s radius of influence, and $\frac{\partial d(x, x_{obs})}{\partial x}$ represents the unit vector from the obstacle to the robot.

The robot' s total potential energy and resultant force are:

$$U_{total} = U_{att} + U_{rep}$$

$$F_{total} = F_{att} + F_{rep}$$

The small-step gradient descent algorithm for artificial potential fields proceeds as follows, where x_0 is the robot' s initial position, ϵ is the set step size, and the function `PotentialGradient` calculates the resultant force according to equation (6).

Algorithm 1: Artificial Potential Field Gradient Descent Search

```
1: Initialize x $\leftarrow$ x$_{0}$
2: while  U_{total}(x) $\neq$ 0 do
3:     F_{total} $\leftarrow$ PotentialGradient(x)
4:     x $\leftarrow$ x +  · F_{total}
5: end while
```

---

2 Goal-bias RRT

RRT is an incremental sampling-based path planning algorithm that establishes a root node at the robot's starting position and guides the random tree to explore uncovered regions by sampling in the robot' s configuration space, eventually finding a path to the target. However, due to the completely random sampling of the initial RRT in configuration space, its search is aimless, resulting in low efficiency. To address this, many heuristic RRT algorithms have been proposed, among which Goal-bias RRT is a simple and effective approach.

Goal-bias RRT incorporates target point information to make RRT' s search more purposeful and improve search efficiency. However, it does not consider obstacle information, which is available in most scenarios. To enhance the algorithm' s perception of the surrounding environment and guide the search using obstacle information, the proposed APFG-RRT integrates artificial potential fields into RRT. Based on target point and surrounding obstacle information, artificial potential fields guide RRT search, enabling the extended random tree to more effectively avoid obstacles and quickly find safe paths to the goal.

In RRT, each node growth increment is determined solely by the direction and step size ϵ from the nearest node $x_{nearest}$ toward the random sampling point x_{rand}. In APFG-RRT, the growth increment is also determined by the attractive and repulsive forces experienced by the robot at $x_{nearest}$. The new node generation formula for APFG-RRT is:

$$x_{new} = x_{nearest} + \epsilon \cdot \phi_{rand} + \phi_{total}(x_{nearest})$$

where $\phi_{total}(x_{nearest})$ is the resultant force of attraction and repulsion experienced by the robot at $x_{nearest}$, ϕ_{rand} is the random component factor, and ϕ_{total} is the potential field component factor. Note that to maintain the algorithm' s probabilistic completeness and enable the random tree to reach any collision-free pose in configuration space, ϕ_{rand} and ϕ_{total} must satisfy $\|\phi_{rand}\| \geq \|\phi_{total}\|$.

Unlike most RRT algorithms with equal step sizes, APFG-RRT determines growth distance through both random and potential field components. When a node is close to obstacles, the potential field component' s direction is primarily determined by repulsive force, resulting in smaller growth steps toward obstacle regions, which helps the random tree pass through narrow areas. When the node is far from obstacles, the potential field component' s direction is mainly determined by attractive force, enabling larger growth steps toward the target region for rapid approach.

The attractive and repulsive functions in APFG-RRT are respectively:

$$F_{att}(x) = K_a \frac{x_{goal} - x}{d(x, x_{goal})}$$

$$F_{rep}(x) = \begin{cases} K_r \left(\frac{1}{d(x,x_{obs})} - \frac{1}{d^*_{obs}} \right) \frac{x - x_{obs}}{d(x,x_{obs})} & d(x, x_{obs}) \le d^*_{obs} \\ 0 & d(x, x_{obs}) > d^*_{obs} \end{cases}$$

where K_a and K_r are the attractive and repulsive constants respectively, k is a shape coefficient, d^*_{obs} is the repulsive potential field' s radius of influence, and x_{obs} is the nearest point on the obstacle to x.

In artificial potential field methods, the attractive force magnitude is set as in equation (2) to enable rapid target approach at long distances while preventing overshoot near the target. Since RRT plans the complete path at once and can directly connect the node to the target when near it without overshooting, APFG-RRT sets the attractive force magnitude as a constant K_a.

The repulsive function in APFG-RRT is equation (9). When the minimum distance between x and obstacles approaches 0, the repulsive force magnitude approaches K_r/d^*_{obs}; when the minimum distance approaches d^*_{obs}, the repulsive force magnitude approaches 0. Compared with equation (4), the parameters in equation (9) have more intuitive meaning and more reasonable repulsive force distribution. Testing shows that setting the repulsive function as equation (9) yields better algorithm performance.

If the probability of selecting the target point as the sampling point is set to 100% for both Goal-bias RRT and APFG-RRT, their performance in simple scenarios is illustrated in Figure 1. Goal-bias RRT grows directly toward the target and easily falls into local minima on obstacle surfaces, requiring random sampling with probability $1 - P$ to escape. In contrast, APFG-RRT with integrated artificial potential fields incorporates a repulsive mechanism, giving it the ability to bypass obstacles.

Figure 1: Schematic diagrams of path planning with Goal-bias RRT and APFG-RRT [Figure 1: see original paper]

3.2 Adaptive Probability Sampling

The strategy of probabilistically selecting the target point as the sampling point in Goal-bias RRT not only guides the random tree toward the goal but also has the characteristic of preferentially growing a single branch, which facilitates rapid path planning. However, Goal-bias RRT uses a fixed probability P for selecting the target point as the sampling point. When the random tree is trapped in local minima, extensions using the target point as the sampling point are generally ineffective, wasting substantial computational resources. Therefore, this paper proposes a probability-adaptive sampling method. When extensions using the target point as the sampling point become ineffective, the random tree is considered trapped in local minima, and P is set to 0, allowing the algorithm to focus on escaping local minima through random growth. As the number of random growth attempts increases, the likelihood of escaping local minima grows, so P correspondingly increases with the number of random growth attempts. The value of P when the random tree is trapped in local minima is obtained by:

$$P = P_{max}\,(1 - e^{-a \cdot n})$$

where P_{max} is the probability of selecting the target point as the sampling point when the random tree is not trapped in local minima, a is a shape coefficient, and n is the set maximum number of attempts. When the number of random growth attempts approaches n, P gradually recovers to P_{max}.

3.3 APFG-RRT Algorithm for Static Environments

The specific process of APFG-RRT is as follows: First, establish the root node of the random tree at the robot' s initial position, setting the probability of selecting the target point as the sampling point to P_{max}. When the random number is less than P, select the target point as the sampling point; when greater than P, sample randomly in the robot' s configuration space. Then select the nearest node $x_{nearest}$ on the random tree to x_{rand}, calculate the nearest obstacle point x_{obs}, compute x_{new} according to equation (7), and check if x_{new} collides with obstacles. If collision occurs and the sampling point is x_{goal}, consider the random tree trapped in local minima; if no collision, add x_{new} as a new node to the random tree, and if the sampling point is x_{goal}, consider the random tree not trapped in local minima or having escaped local minima, resetting P to P_{max}. If the random tree is in a local minima state, update P according to equation (10). Repeat these steps until the random tree grows into the target neighborhood or iterations exceed the set maximum. The pseudocode for APFG-RRT in static environments is:

Algorithm 3: APFG-RRT for Static Environments

```
1: InitializeTree() $\leftarrow$ T_{init}
2: P $\leftarrow$ P_{max}, local_{minima} $\leftarrow$ False, n $\leftarrow$ 0
3: for i = 1 to K do
```

```
    if random(0,1) < P then
        x_{rand} $\leftarrow$ x_{goal}
    else
        x_{rand} $\leftarrow$ Sample()
    end if
    x_{nearest} $\leftarrow$ NearestNode(x_{rand}, T)
    x_{obs} $\leftarrow$ NearestObstacle(x_{nearest}, Obstacle)
    x_{new} $\leftarrow$ Steer(x_{nearest}, x_{rand}, x_{goal}, x_{obs})
    if CollisionFree(x_{new}) then
        T.add_{node}(x_{new})
        T.add_{edge}(x_{nearest}, x_{new})
        if d(x_{new}, x_{goal}) <   then
            return T
        end if
        if x_{rand} == x_{goal} then
            P $\leftarrow$ P_{max}, local_{minima} $\leftarrow$ False, n $\leftarrow$ 0
        end if
    else if x_{rand} == x_{goal} then
        local_{minima} $\leftarrow$ True
        UpdateProbability(P, n)
        n $\leftarrow$ n + 1
    end if
end for
```

3.4 APFG-RRT Algorithm for Dynamic Environments

The difficulty of path planning in dynamic environments lies in obstacles potentially moving onto previously planned paths, requiring the algorithm to replan a collision-free path within extremely short time, thus demanding high real-time performance. Additionally, considering the need to leave sufficient avoidance time and space for the robot, the planned path should stay away from obstacles. Traditional obstacle inflation methods can maintain a safety distance between planned paths and obstacles, but require adjusting inflation parameters for different scenarios—for instance, choosing smaller inflation in narrow environments to avoid hindering random tree expansion. APFG-RRT, due to its potential field-guided growth mechanism, achieves faster convergence and generates paths relatively far from obstacles, demonstrating good performance in dynamic environments.

To further improve real-time performance in dynamic environments, this paper adopts a local replanning strategy, referencing the method in [**?**], by only replanning the "dangerous" portion of the unexecuted path within a certain range of the current position, thereby reducing computation. Specifically, it first searches for the continuous path p_{near} to be executed within a radius R of the current position, identifies nodes with minimum distance to obstacles within d_{safe} as "dangerous" nodes, and uses APFG-RRT to replan and connect the

shortest path segment on p_{near} containing all dangerous nodes. The algorithm pseudocode is:

Algorithm 4: APFG-RRT for Dynamic Environments

```
1: Obstacle $\leftarrow$ UpdateObstacles()
2: T $\leftarrow$ APFG-RRT(x_{init}, x_{goal}, p, Obstacle)
3: x_{current} $\leftarrow$ x_{init}
4: while x_{current} $\neq$ x_{goal} do
5:     Obstacle $\leftarrow$ UpdateObstacles()
6:     x_{current} $\leftarrow$ GetCurrentState()
7:     p_{near} $\leftarrow$ SelectNearPath(x_{current}, R)
8:     x_{start}, x_{end} $\leftarrow$ DangerousPath(p_{near}, d_{safe})
9:     if x_{start} $\neq$ NULL then
10:        p_{rest} $\leftarrow$ DeletePath(p, x_{start}, x_{end})
11:        p_{changed} $\leftarrow$ APFG-RRT(x_{start}, x_{end}, p, Obstacle)
12:        p $\leftarrow$ AddPath(p_{rest}, p_{changed})
13:    end if
14: end while
```

4 Experimental Results and Analysis

To test APFG-RRT' s convergence speed, this paper conducted simulation experiments in various environments and compared it with the original RRT and Goal-bias RRT, keeping corresponding parameters identical across the three algorithms. Considering RRT' s inherent randomness, path planning from start to goal was repeated 100 times on maps corresponding to general and narrow environments, with average values taken as test results.

Figures 2 [Figure 2: see original paper] and 3 [Figure 3: see original paper] show the performance of the original RRT, Goal-bias RRT, and APFG-RRT in general and narrow scenarios, while Tables 1 and 2 list the corresponding experimental data. The tables reveal that Goal-bias RRT and APFG-RRT significantly outperform the original RRT, with APFG-RRT superior to Goal-bias RRT across all metrics. In Map 1 (general scenario), APFG-RRT' s average planning time is 49.3% of Goal-bias RRT' s, and its average node count is 41.5% of Goal-bias RRT' s. In Map 2 (narrow scenario), APFG-RRT' s average planning time is 47.2% of Goal-bias RRT' s, and its average node count is 42.9% of Goal-bias RRT' s. This is because although artificial potential field introduction increases per-iteration computation, it gives APFG-RRT the ability to bypass obstacles, reducing invalid nodes from obstacle collisions. The adaptive probability sampling strategy further reduces invalid nodes from repeated calculations due to local minima. From the data, Goal-bias RRT' s average invalid node counts in Maps 1 and 2 are 5189 and 5555 respectively, while APFG-RRT' s are only 164 and 12, substantially reducing time wasted on invalid iterations. Moreover, as the number of nodes in the random tree increases, the time con-

sumed to search for the nearest node to the sampling point grows. Goal-bias RRT samples randomly with probability $1 - P$, while APFG-RRT' s growth direction during probability $1 - P$ is determined by both potential field and random components, making APFG-RRT' s expansion more purposeful and resulting in fewer nodes when a path is found, which reduces average per-iteration time. For narrow scenarios, Goal-bias RRT' s goal-biased growth characteristic easily traps it in cycles where extensions using the target point as sampling point fail near obstacle surfaces, requiring random sampling with probability $1 - P$ to escape. When near narrow passage surfaces, the random tree' s growth direction is restricted, with limited effective sampling space. APFG-RRT, however, tends to grow along the middle of narrow passages where potential energy is lowest, facing less restriction on growth direction and achieving higher effective growth probability. When the random component of growth points toward passage surfaces, the potential field component points away from obstacles, reducing growth step size and further improving effective growth probability.

Table 1: Experimental results of the three algorithms in general scenario

Table 2: Experimental results of the three algorithms in narrow environment

Figure 4 [Figure 4: see original paper] demonstrates APFG-RRT' s performance in dynamic scenarios using the local replanning strategy. Gray objects are moving obstacles traveling back and forth on fixed routes, with arrows indicating their movement direction in each frame. The thick yellow solid line shows the executed robot path, red dotted lines indicate corrected portions of the path planned in the previous frame, and thin blue solid lines show the replanned path in the current frame. Figure 4 shows that with potential field-guided growth, APFG-RRT plans paths relatively far from obstacles, leaving sufficient avoidance space. Notably, local replanning can optimize the original path to some extent—for instance, in the third subfigure of Figure 4, the replanned path is shorter and smoother than the original, and moves the path away from obstacles where it was previously trapped in local minima. This is because the potential field characteristics enable APFG-RRT to perform well in locally simple environments.

Table 3: Experimental results of Goal-bias RRT and APFG-RRT in dynamic environment

In experiments, Goal-bias RRT in dynamic scenarios replans the entire path at each frame. Both algorithms were tested 100 times in the scenario shown in Figure 4, with average planning results presented in Table 3 . Due to Goal-bias RRT' s goal-biased growth characteristic, its searched paths tend to cling to obstacles in local minima regions, potentially causing robot-obstacle collisions. As shown in Figure 5 [Figure 5: see original paper], Goal-bias RRT' s replanned path in the current frame is collision-free but closely follows obstacles. In the next frame, the moving obstacle in its direction of motion will collide with the robot, which explains Goal-bias RRT' s 0% planning success rate in the experimental dynamic scenario. For most other RRT algorithms, lacking a

mechanism to stay away from obstacles may also result in paths closely following obstacles, as seen in the original RRT' s planned path in Figure 2(a).

Since Goal-bias RRT's planning success rate is 0%, its average replanning time in Table 3 is statistically derived from successfully replanned portions. The results show that APFG-RRT with local replanning achieves 100% planning success rate in the tested dynamic scenario, with average replanning time far less than Goal-bias RRT. This is because compared with algorithms that replan the entire global path at each frame, local replanning only requires replanning a small path segment, and may not require any replanning when the path to be executed is safe, significantly reducing computation. The additional computation from introducing the local replanning strategy is comparatively minimal.

---

5 Conclusion

This paper addresses the problem that most existing RRT dynamic path planning algorithms cannot maintain safe distances between planned paths and obstacles. It proposes a heuristic algorithm that uses artificial potential fields to guide RRT growth, enhancing environmental perception capability, and combines adaptive probability sampling with local replanning strategies to further improve RRT search efficiency. Simulation results demonstrate that the proposed improved RRT algorithm exhibits superior performance in both static and dynamic scenarios.

References

[1] Tzafestas, Spyros G. Mobile Robot Control and Navigation: A Global Overview [J]. Journal of Intelligent & Robotic Systems, 2018, 91 (1): 35-58.

[2] Kong Xiangzhan, Duan Xingguang, Wang Yonggui. An integrated system for planning, navigation and robotic assistance for mandible reconstruction surgery [J]. Intelligent Service Robotics, 2016, 9 (2): 113-124.

[3] Elbanhawi M, Simic M. Sampling-Based Robot Motion Planning: A Review [J]. IEEE Access, 2014, 2 (2): 56-77.

[4] Zhang Yinyan, Li Shuai, Guo Hongliang. A type of biased consensus-based distributed neural network for path planning [J]. Nonlinear Dynamics, 2017, 89 (3): 1803-1815.

[5] Bounini F, Gingras D, Pollart H, et al. Modified artificial potential field method for online path planning applications [C]// Proc of IEEE Intelligent Vehicles Symposium (IV). Piscataway, NJ: IEEE Press, 2017: 180-185.

[6] Li Jinghua, Huang Yibin, Xu Zhao, et al. Path planning of UAV based on hierarchical genetic algorithm with optimized search region [C]// Proc of IEEE

International Conference on Control & Automation. Piscataway, NJ: IEEE Press, 2017: 1033-1038.

[7] Liu Yang, Ma Jianwei, Zang Shaofei, et al. Dynamic Path Planning of Mobile Robot Based on Improved Ant Colony Optimization Algorithm [C]// Proc of the 8th International Conference on Networks, Communication and Computing. New York: ACM Press, 2019: 248-252.

[8] Latip N B A, Omar R, Debnath S K. Optimal Path Planning using Equilateral Spaces Oriented Visibility Graph Method [J]. International Journal of Electrical & Computer Engineering, 2017, 7 (6): 3046-3051.

[9] Kumar N, Vámossy Z, Szabó-Resch Z M. Robot path pursuit using probabilistic roadmap [C]// Proc of IEEE International Symposium on Computational Intelligence & Informatics. Piscataway, NJ: IEEE Press, 2016: 139-144.

[10] Kalisiak M, Panne M V D. RRT-blossom: RRT with a local flood-fill behavior [C]// Proc of IEEE International Conference on Robotics and Automation. Piscataway, NJ: IEEE Press, 2006: 1237-1242.

[11] Perez A, Karaman S, Shkolnik A, et al. Asymptotically-optimal path planning for manipulation using incremental sampling-based algorithms [C]// Proc of IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, NJ: IEEE Press, 2011: 4307-4313.

[12] Lavalle S M, Kuffner J J. Randomized kinodynamic planning [C]// Proc of IEEE International Conference on Robotics and Automation. Piscataway, NJ: IEEE Press, 1999: 473-479.

[13] Lavalle S M. Planning Algorithms [M]. Cambridge: Cambridge University Press, 2006.

[14] Zhou Fang, Zhu Qidan, Zhao Guoliang. Path Optimization of Manipulator Based on the Improved Rapidly-exploring Random Tree Algorithm [J]. Journal of Mechanical Engineering, 2011, 47 (11): 30-35.

[15] He Zhaochu, He Yuanlie, Zeng Bi. Obstacle Avoidence Path Planning for Robot Arm Based on Mixed Algorithm of Artificial Potential Field Method and RRT [J]. Industrial Engineering Journal, 2017, 20 (2): 56-63.

[16] Xu Xiaohui, Zhang Jinlong. Hybrid optimization algorithm of improved artificial potential field and TAS-RRT [J]. Application of Electronic Technique, 2018, 44 (10): 88-92.

[17] Qureshi A H, Mumtaz S, Iqbal K F, et al. Adaptive Potential guided directional-RRT [C]// Proc of IEEE International Conference on Robotics and Biomimetics. Piscataway, NJ: IEEE Press, 2013: 1887-1892.

[18] Qureshi A H, Ayaz Y. Potential functions based sampling heuristic for optimal path planning [J]. Autonomous Robots, 2016, 40 (6): 1079-1093.

[19] Zaid T, Qureshi A H, Yasar A, et al. Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments [J]. Robotics and Autonomous Systems, 2018, 108: 13-27.

[20] Adiyatov O, Varol H A. A novel RRT*-based algorithm for motion planning in Dynamic environments [C]// Proc of IEEE International Conference on Mechatronics and Automation. Piscataway, NJ: IEEE Press, 2017: 1416-1421.

Note: Figure translations are in progress. See original paper for figures.