IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 9, 2019, accepted May 27, 2019, date of publication June 3, 2019, date of current version August 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2920420

Path Planning Based on ADFA* Algorithm for Quadruped Robot

LI ZHE[1], LI YIBIN[1], RONG XUEWEN[1], AND ZHANG HUI[2]
[1]School of Control Science and Engineering, Shandong University, Jinan 250061, China
[2]School of Electrical Engineering and Automation, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China

Corresponding author: Li Yibin (liyb@sdu.edu.cn)

This work was supported by the General Program of National Natural Science Foundation of China under Grant U1613223, National Key Research and Development Project under Grant 2017YFC0806505 and Grant 2015AA042201.

ABSTRACT At present, the path-planning algorithm based on the grid map is generally adopted in the field of quadruped robot and the obtained environmental information is represented by a standardized grid map. In this paper, the ADFA* algorithm introduces a dilation factor based on the DFA* to solve the path planning problem under the constraint of computing time and provide a path search result related to the time limit. Path-planning algorithms based on raster maps often equate robots with particles, causing problems, such as path blocking. The FA* algorithm adds raster tolerance to expand obstacles. DFA* uses a path-splitting approach that, such as the DA* algorithm, has better dynamic environment processing capabilities than the FA*. However, during the actual operation of the robot, the environmental information acquired is extremely frequent due to its instability. The robot itself is often in a relatively static state. Therefore, compared with obtaining the shortest path, it is more practical to improve the path search efficiency under dynamic map environment. ADFA* will gradually optimize the path and eventually get the optimal solution when time is sufficient. When the time is limited, ADFA* will search for the current optimal solution under the specified search time but may not be able to obtain the shortest path, which is called the second best solution.

INDEX TERMS Quadruped robot, path planning, A* algorithm.

I. INTRODUCTION

Traditional crawler robots usually need to travel in a gentle surface environment, however, legged robots need only discrete footfalls to operate in rough terrain. It is obvious that the legged robot has more environmental adaptability and flexibility. In the field of legged robotics, autonomic motions in complex environment have been one of the most important research problems. The core of solving this kind of problem is terrain recognition and path planning. Since BigDog [1], many relevant research institutes around the world have carried out relevant research on the path planning of legged robot in complex environment.

BigDog visual system including planar laser scanner and binocular camera, visual system realizes the functions including environmental information collection, identification and construct, path planning, with the help of the A* algorithm [2]. Yuntai planar laser scanner with binocular cameras vision system of LS3 [3], the visual system contains the lighting equipment, realize the night environment recognition, and to express environment information, using the elevation drawing. HyQ [4] used a binocular camera and a Kinect camera to reconstruct the environment map. The traditional Dijkstra [5], A* and ARA* [6] algorithms were used to achieve path planning. The DLR-Crawler uses binocular cameras for environmental perception only and uses D* Lite [7] algorithm for path planning. Scalf [8], [9] uses IEA* for Path Search and Barrier Grid Expansion.

Significant progress has been made in the rugged terrain navigation of quadruped robots. It has been proven to improve exercise efficiency when walking on rugged terrain [26]. Elevation maps have been used for characterization to plan a foothold for optimal stability and obstacle avoidance. The motion of the robot can be made more dynamic by optimizing the dynamics of the robot. [27] However, these methods do not consider the collision with the robot body, which is necessary in a narrow space.

The above path planning algorithm are mostly based on improved A* algorithm. In order to solve the path planning in consumption, fidelity, long path planning and obstacle

The associate editor coordinating the review of this manuscript and approving it for publication was Rui Xiong.

avoidance in A row, A* algorithm needs to be improved. This paper improved the core evaluation function, and grid map path planning method is proposed. This method can meet the demand of robot operation. Experiments proved that the algorithm advantages, when it comes to long distance for obstacle avoidance, consumption.

II. DA* ALGORITHM

A* algorithm is a common grid map path planning method, which was proposed by Nilsson in 1980. It can search the optimal path to target point by contrast evaluation function. The core of it is to add heuristic search part based on Dijkstra algorithm. Its evaluation function is

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

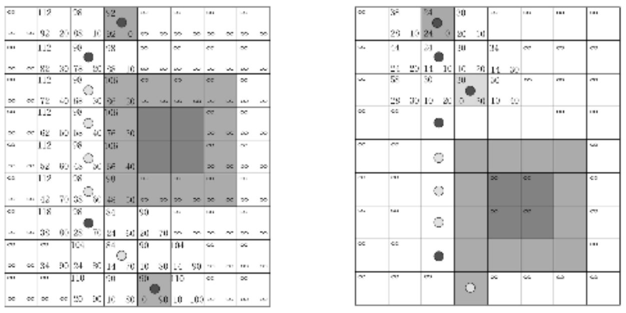
f(n) is the evaluation function, g(n) represents the path cost from the initial point n to any node. h(n) said heuristic evaluation price, from the node n to the target point. After comparing traversal grid evaluation function, A* algorithm program gives the optimal path of the current environment. It can be seen from its core evaluation function that A* algorithm is a path planning algorithm for static grid maps, and its operation process does not consider changes of environmental information with time. Therefore, when the map environment changes, the A* algorithm will discard the existing path information and perform path search again. During the actual operation of the robot, the robot is in a relatively stationary position in the grid map due to the slower traveling speed. Using the original A* algorithm results in a partial overlap between the updated path and the path already obtained. If we can amend this part of the overlapping path, it will significantly improve the efficiency of path planning. At the initial moment, DA* obtains the optimal path under the current environment through the A* algorithm and adds the marking parameters for all the way points D,

$$D = \begin{cases} 1, & \Delta(\Delta g(n)) \neq o \\ 1, & \Delta(\Delta h(n)) \neq 0 \\ 0, & else \end{cases} \tag{2}$$

where $\Delta(\Delta g(n))$ represents the second-order difference of the n-values of the pathway raster sequence and $\Delta(\Delta h(n))$ corresponds to the second-order difference of the n-values of the pathway raster sequences. When D = 1, the corresponding grid is the relay point at the current moment. At the initial time, the optimal path is obtained through the A* algorithm from the starting point to the end. When the environment map changes, the optimal path is coordinate translation, vary according to the position of the robot, eventually get the updated evaluation function,

$$\begin{aligned} g(n') &= g(n) - g(n^*) \\ h(n') &= |x_t - x_{n'}| + |y_t - y_{n'}| \\ f(n') &= g(n') - h(n') \end{aligned} \tag{3}$$

where n^* represents the grid node in the robot's current position, n represents the grid node that has not been reached,



(a) The selection of the relay point (b) The path spliced

FIGURE 1. Path segmentation and combination based on DA*.

(x_t, y_t) represents the coordinate of the target node, and (x'_n, y'_n) represents the coordinates of the path node that has not arrived. At the same time, the DA* algorithm searches for the path relay $\mathcal{S}$ with a smaller f-number in the set of relay points n_D starting from the end of the updated path, satisfying

$$f(s) = \min_{n \in nD} \{f(n)\} \tag{4}$$

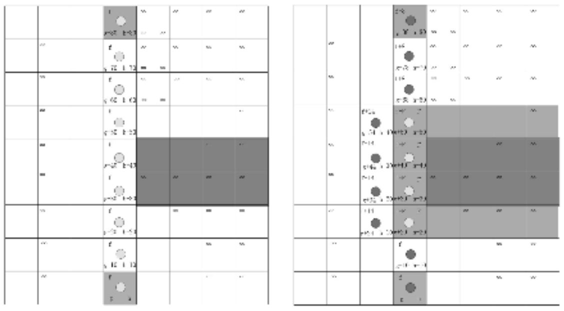
Then the A* algorithm performs routing planning, starting from the relay node and ending with the goal node. The obtained path and the path after the initial point is shifted to the relay point $\mathcal{S}$ are spliced to obtain the complete path R. Since the path is not planned from the initial starting point, it can not be guaranteed to be the shortest path, which is called the second best path.

III. FA* ALGORITHM

In the grid-based path planning algorithm, the robot is usually equated to a particle, and then the obstacle is expanded according to the size of the robot so that the initial barrier grid and the expanded barrier grid together form a non-passable area. For the legged robot, the shape of the road is rectangular, which means that the distance from the centered of the robot to the edge of the contour is not a constant, which greatly increases the selection difficulty for the obstacle grid expansion radius.

Based on the A* algorithm, the FA* algorithm first expands the barrier grid according to the shortest distance from the center of the robot to the edge of the contour and updates the grid map information, and then introduces an obstacle level F for each grid point to represent the ruggedness of the surface where the grid is located. A further expansion of the barrier grid is achieved, in such a way that the resulting path is prevented from being obstructed by the narrow channel due to the expansion of the barrier grid, and the planned path is further removed from the barrier grid. In this way, the secondary expansion of the barrier grid in the path planning is achieved without changing the initial grid map information.

Grid points with high obstacle levels have higher evaluation function values f and are less likely to be selected as pathways. By adjusting the relationship between obstruction level and expansion, the problem of path congestion can be improved.



(a) The path gerated by the A* algorith

(b) The path expected by the FA* algorithm

FIGURE 2. Path modification based on FA*.

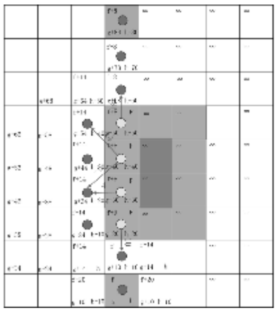


FIGURE 3. The difference value of g and h.

The original path generated by A* is shown in Figure 2(a). If the path of the FA* algorithm is far away from the initial barrier, as shown in Figure 2(b), it can be seen from the core evaluation function that the evaluation function of the new path and the original path needs to satisfy $f\left(R'\right) < f(R)$. According to the annotation in the diagram, the path update condition is expanded to,

$$f + 14 = (g + \Delta g) + (h + \Delta h) < g + h + F = f + F \quad (5)$$

where Δg represents the amount of change in g required for a new path point and Δh represents the amount of change in h. In a broad sense, it can be assumed that the risk level F for extending a grid should be greater than the increment of the value of f for a new path node, i.e., the sum of g, h.

According to the path shown in Figure 3, we can see, $|\Delta g| = 4, 6, 10, 14, 18 |\Delta h| = 10, 20$. Considering the more general case, when $\Delta h = 20$, the path will be far away from the target point, so $\Delta h = 10$ in the normal state. Considering $\Delta g = -10, 4, 18$, the new path contains a common edge to the grid in which the old path resides, whereas the new path contains a common point with the old path at $\Delta g = -6, 14$. From the figure we can see that the expected expansion path contains public side, so the desirable $\Delta g = -4, 10, 18$ can meet the expected path requirements. But a larger F-value means that robots will cost more to get through the grid, with the result that they can cause obstruction to the narrowed channel. So take $F > 14$ to meet the requirements.

In the actual operation, if we strictly follow the above rules, selecting the F value of extended barrier grid is essentially the same as that of scaling grid barrier directly. In order to dynamically improve its passability and to highlight the characteristics of extended grid pay, it is necessary to compensate for the level of obstacle F, which corresponds to the change in the position of the obstacle. Closer to the interior of the barrier, the higher its cost; the more away from the barrier center, the lower the cost.

When the obstacle extension grid is located at the edge, inside or corner, the Δg is different. Judging whether an obstacle grid is at the edge of the obstacle depends on the number l of the barrier rasters belonging to the barrier. When $l = 4$, then the barrier is divided into obstacles within the barrier; when $l = 3$, then the barrier attached to the edge of the barrier, when $l < 3$, then the obstacle attached to the obstacle barrier angle. The reason why the F-value of the light-colored barrier area can not be increased indefinitely is that since the light-colored barrier area is not an unacceptable obstacle, it is a rugged area that is different from a white-freely-accessible area. F-value is the quantification of its passing cost.

$$F = \begin{cases} \infty, & l = 4 \\ F_0, & F_0 > 14,\ l = 3 \\ F_1, & 0 < F_1 < 6,\ l < 3 \end{cases} \quad (6)$$

For the case of expanding to 2 grids or more, since the judgment of F is directly related to l, it does not affect the conclusion.

IV. DFA* ALGORITHM

The principle of DFA* algorithm is similar to that of DA*, that is, path searching is performed by FA* algorithm at the initial time to obtain path R. After the environment information is changed, the initial path R and the environment map information are updated according to the change of the robot position, and the path R' which the robot has not arrived is generated, and then the evaluation function of the path is updated.

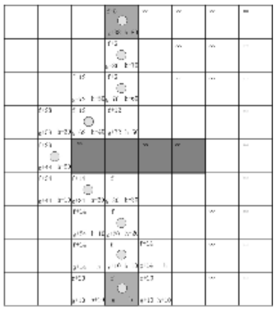
$$\begin{aligned} g(r^{'})^{'} &= g(r^{'})^{'} - d(r^{'}) \bullet \Delta F - g(rb^{'})^{'} + d(r_b^{'})^{'} \bullet \Delta F \\ h(r^{'}) &= \left|x_g - x_{r'}\right| + \left|y_g - y_{r'}\right| \\ f(r^{'}) &= g(r^{'}) + h(r^{'}) \end{aligned} \quad (7)$$

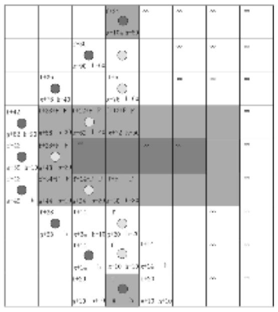
In the equation, r' represents the remaining part of the grid node after path shift, ΔF indicates the F-value increment corresponding to the amount of raster shift, r_b represents the starting point of r', and $\left(x_g, y_g\right)$ represents the coordinates of the target grid node. $(x_{r'}, y_{r'})$ represents the coordinates of the path node r'.

Meanwhile, the DFA * algorithm searches for the relay point s with a small f value from the end of the translated path to satisfy

$$f(s) = \min_{n \in nD} \{f(n)\} \quad (8)$$

The updated path from the path point to the target point and the previously obtained path from the starting point to the


(a) The path gerated by the A* algorith


(b) The path generated by the FA* algorithm

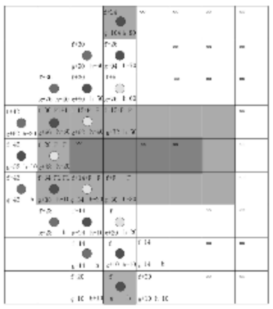
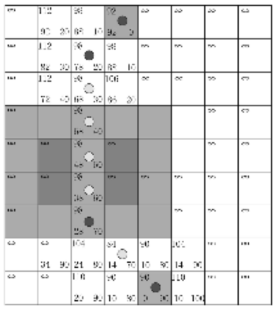
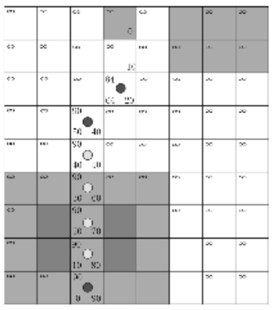

(c) The path of improved FA* algorithm

FIGURE 4. Path planning based on FA*.


(a) Selection of the current Path relay point


(b) Path update after environment information is updated

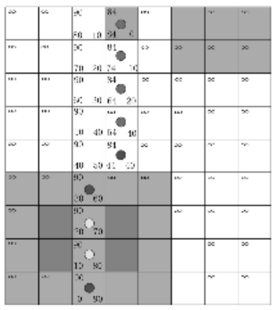

(c) The spliced new path

FIGURE 5. Path planning based on DFA*.

path node are spliced together to form a complete path. This path is also just an increment of the path at the last moment rather than a complete FA* path plan under the new context information. Therefore, it can not guarantee the shortest path distance, so it is called the second best path.

V. ADFA* ALGORITHM

In solving the problem of robot path planning in the actual operation, the search time is usually limited to a limited range. Full-time search is very adaptable to solve this kind of problem. In the process of specific path planning, we first obtain a more suitable path quickly and then perfect the path within the search time constraints. The ADFA* algorithm is based on this notion, introducing an achievable search time constraint on a concrete path search based on DFA*. When the search time is sufficient, the algorithm can get a fully ideal path. Based on DFA* incremental panning, when the surrounding environment changes, ADFA* does not re-route search from the starting point like the A* algorithm, but rather optimizes the existing route search results to update the current Path to reduce the amount of computation.

As usual, the heuristic function h (s) should be a constant input, and meet

$$\begin{cases} h(s) \leq c(s, s^{'}) + h(s^{'}), & s \neq s_{goal} \\ h(s) = o, & s = s_{goal} \end{cases} \tag{9}$$

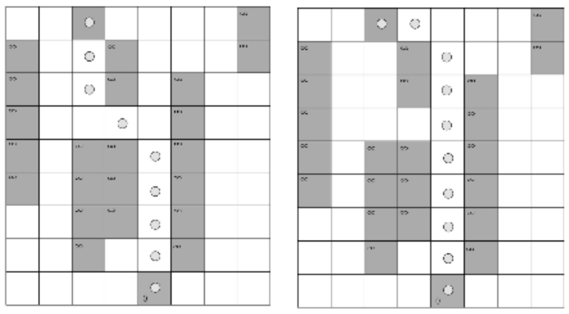
where $c\left(s, s'\right)$ represents the cost of the value needed to reach the node s' from s, which is usually a positive value.

During the operation, in order to ensure that the heuristic function can work normally and obtain the shortest path planning result, the admissibility principle of the heuristic function is introduced, that is, the value of $h(s)$ can not be greater than the true cost required. If we introduce the expansion factor ε to rewrite the weight of the heuristic function, the expansion of the state points will be reduced during the operation, so as to improve the operation efficiency. However, once the weight of heuristic function is increased, the principle of admissibility of the heuristic function will be undermined, which will result in that the obtained path is not the optimal path. In contrast to the DFA* algorithm, ADFA* introduces the expansion factor ε and rewrites the cost evaluation function f as

$$f = F + g(s) + \varepsilon \cdot h(s, s_{goal}) \tag{10}$$

Set ε to a larger value at the beginning of the search and then gradually lower the factor each time it is optimized until $\varepsilon = 1$.

Figure 6 shows a simple simulation under the grid map ε, values were 2.5,1.5,1. The blue grid area corresponds to the obstacle, the green grid area corresponds to the starting point, and the red grid area corresponds to the target point. When the value ε of is set to 1, the path planning result will coincide with the original DFA* planning result because the inspiration part of the evaluation function is not expanded at this time, and the planning result at this time may be called the optimal path. For $\varepsilon > 1$, the suboptimal path is a series of ε-constrained paths. The specific length of the obtained path and the operation time consumption satisfy the following


(a) ε=2.5 (b) ε=1.5

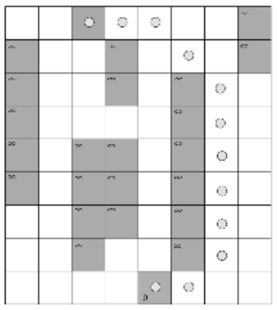

(c) ε=1

FIGURE 6. Path planning based on ADFA* with decreasing ε.

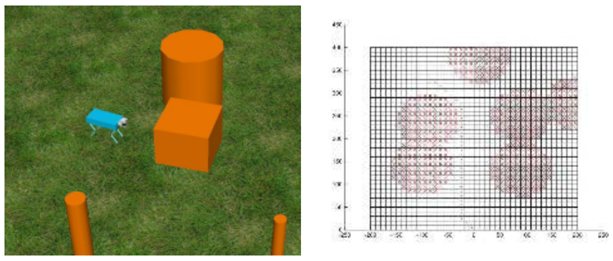

(a) Webots (b) Matlab

FIGURE 7. Simulation environment with Webots and Matlab.

formula,

$$\begin{cases} L_{s_s s_g} \leq \varepsilon \cdot l_{s_s s_g} \\ T_\varepsilon < T \end{cases} \tag{11}$$

where $L_{s_s s_g}$ denotes the total length of the suboptimal path, $l_{s_s s_g}$ denotes the total length of the optimal path, ε denotes the expansion coefficient, T_ε denotes the operation time after the expansion factor ε is introduced, and T denotes the maximum available operation time.

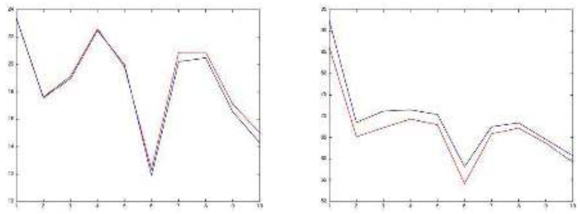
VI. SIMULATION AND EXPERIMENT

In order to evaluate the performance of the algorithm mentioned in this article, the algorithm is tested in Webots and Matlab simulation software respectively. The simulation environment is shown in the Figure 7.

In order to test the planning efficiency of DA* algorithm, this section compares the path length of A* and DA* algorithm and the number of traversal grid in the same map environment respectively.

TABLE 1. Path length and time cost based on A* and DA*.

Num	Total Length /m		Time cost /s	
	A*	DA*	A*	DA*
1	23.40	23.37	86.20	92.40
2	17.63	17.58	65.21	68.56
3	19.13	18.96	67.33	71.20
4	22.60	22.49	69.34	71.52
5	19.84	20.01	68.01	70.42
6	12.32	11.90	54.21	58.17
7	20.87	20.21	65.90	67.56
8	20.89	20.48	67.24	68.45
9	17.11	16.56	63.72	64.59
10	15.03	14.29	59.24	60.75


(a) Path length (b) Time cost

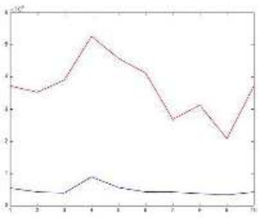

(c) The amount of passed grid

FIGURE 8. Effectiveness comparison between A* and DA*.

The red data in the above figure shows the total length of the planned path of the A* algorithm, the time consumption, and the number of traversed grids. The blue data represents the length of the DA* algorithm planning path, the time consumption, and the number of traversal grids. The comparative analysis shows that although the path length of A* is almost the same as that of DA *, the number of traversal grids of DA* algorithm is obviously smaller than that of A* algorithm, and the extra time consumption is not significant. If we consider only traversing the grid number, the DA* algorithm has more than 6 times the search efficiency of the A* algorithm.

In the Figure 8, the gray grid indicates the obstacle, the red grid indicates the extended grid, the blue path indicates the FA* algorithm operation result, and the green path is the smoothed path. It can be seen that increasing the pass-through cost F can achieve the expansion of the barrier grid.

Based on the A* algorithm, this paper presents the FA* algorithm, with additional tolerance F. Through path splicing, DFA* algorithm is implemented. Based on the DFA* algorithm, ε parameters are introduced to restrict the time condition, making the algorithm more suitable for practical operation. Compared with the original A* algorithm, the

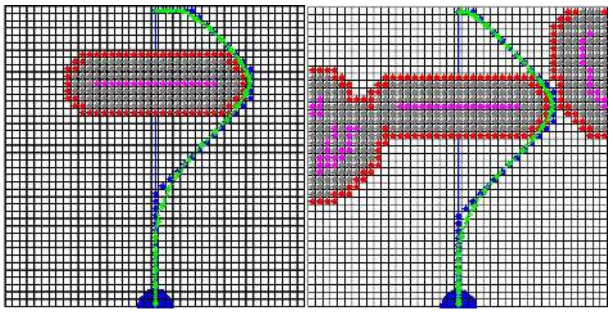


FIGURE 9. Path planning simulation based on FA*.

TABLE 2. Grid number based on A* and DA*.

Num	Grids Amount		A*/DA*
	A*	DA*	
1	37265	5482	6.80
2	35316	4275	8.26
3	39101	3998	9.78
4	52714	8976	5.87
5	45785	5669	8.08
6	41276	4254	9.64
7	26877	4254	6.32
8	31280	3785	8.26
9	20861	3397	6.14
10	37263	4267	8.73

TABLE 3. Path length and time cost based on A* and ADFA*.

Num	Total Length /m		Time cost /s	
	A*	ADFA*	A*	ADFA*
1	24.45	22.37	59.51	61.14
2	14.88	20.51	58.51	60.32
3	25.05	23.03	72.15	76.80
4	20.72	18.69	56.41	58.92
5	21.85	19.72	58.00	60.90
6	20.33	17.92	55.29	55.94
7	18.77	16.22	51.93	53.06
8	16.74	14.43	48.16	51.13
9	23.15	20.66	56.93	58.17
10	23.17	20.86	57.93	58.84

accurate expansion of the obstacle can be accomplished. In the process of path planning with long obstacle avoidance, the selected relay points can greatly reduce the time consumption and improve the computational efficiency of the algorithm.

As shown in Table 3 and Table 4, multiple sets of experimental data were obtained at specific ε value. The comparative analysis shows that the path length of ADFA* is almost the same as that of A*, the number of traversal grids of ADFA* algorithm is obviously smaller than that of A* algorithm, and the extra time consumption is not significant. If we consider only traversing the grid number, the ADFA*

TABLE 4. Grid number based on A* and ADFA*.

Num	Grids Amount		A*/ADFA*
	A*	ADFA*	
1	47036	7482	6.29
2	41839	5002	8.36
3	35449	4862	7.29
4	33987	3956	8.59
5	36826	3749	9.82
6	23146	3298	7.02
7	35447	3950	8.97
8	38457	3962	9.71
9	27658	3941	7.02
10	30960	3589	8.63

algorithm has more than 7 times the search efficiency of the A* algorithm. The algorithm will be more efficient when a appropriate ε value is selected. At the same time, the obstacle avoidance task was completed more rigorously.

For Quadruped Robot, the ideal algorithm guarantees computational efficiency in a complex environment and gets the exact path within the time allowed. Although the ADFA* algorithm proposed in this paper has been improved for long-distance continuous obstacle avoidance, this method is aimed at the grid map using Euler distance measurement. However, the idea of obstacle avoidance and the method of improving efficiency in long distance can be widely used in the path planning of other forms of maps. To improve Quadruped Robot path planning progress, the next step should be to expand the application of the algorithm, under other forms of map. As algorithms face more complex environments, map information changes at high frequencies, and its dynamic adaptability will be challenged. The next step will be to perfect the algorithm to meet the requirements of the actual operation, when the map information is constantly updated.

VII. CONCLUSION AND PROSPECT

In practical applications, obstacle avoidance problems and operational problems within limited time are all considered. The ADFA algorithm can solve these problems well and has been proved by experiments. The dilation factor has been introduced to DFA* algorithm to solve the path planning problem under the constraint of computing time and provide a path search result related to the time limit. ADFA* will gradually optimize the path and eventually get the optimal solution when time is sufficient. Compared with the traditional path planning algorithm, ADFA* will search for the current optimal solution under the specified search time but may not be able to obtain the shortest path, which is called the second best solution. At the same time, the path obstacle avoidance problem is solved, which takes less time than the traditional algorithm, and has a more easily adjusted scaling coefficient. Dealing the same map information, the number of passed grid is reduced to a quarter of the original requirement, which means increased efficiency when processing continuous and large amounts of terrain data. The future work is to deal with the problem of long distance path planning on the one hand, and the problem of rugged terrain planning on the other hand. More efficient path segmentation methods will be adopted with high probability, with the help of terrain belt.

REFERENCES

[1] M. Raibert, K. Blankespoor, R. Playter, and G. Nelson, "BigDog, the rough-terrain quadruped robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, Juneau, AK, USA, Jul. 2008, pp. 4736–4741.

[2] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[3] M. Perdoch, D. M. Bradley, J. K. Chang, H. Herman, P. Rander, and A. Stentz, "Leader tracking for a walking logistics robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Hamburg, Germany, Sep./Oct. 2015, pp. 2994–3001.

[4] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *Proc. IEEE Int. Conf. Robot. Automat.*, Washington, DC, USA, May 2015, pp. 5148–5154.

[5] M. A. Arain, I. Havoutis, C. Semini, J. Buchli, and D. G. Caldwell, "A comparison of search-based planners for a legged robot," in *Proc. 9th Int. Workshop Robot Motion Control*, Kuslin, Poland, Jul. 2013, pp. 104–109.

[6] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 767–774.

[7] S. Koenig and M. D. Likhachev, "D* Lite," in *Proc. AAAI Conf. Artif. Intell. (IAAI)*, 2002, pp. 476–483.

[8] H. Chai, M. Jian, R. Xuewen, and L. Yibin, "Design and implementation of SCalf, an advanced hydraulic quadruped robot," *Robot*, vol. 36, no. 3, pp. 385–391, 2014.

[9] Z. Hui, R. Xuewen, L. Yibin, L. Bin, Z. Junwen, and Z. Qin, "Path planning based on sliding window and variant A* algorithm for quadruped robot," School Control Sci. Eng., Shandong Univ., Jinan, China, School Elect. Eng., Jinan Univ., Jinan, China, Oct. 2016, pp. 334–342.

[10] D. Wooden, "Graph-based path planning for mobile robots," Ph.D. dissertation, Georgia Inst. Technol., Atlanta, Georgia, 2006, pp. 102–150.

[11] D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A. A. Rizzi, and M. Raibert, "Autonomous navigation for BigDog," in *Proc. IEEE Int. Conf. Robot. Automat.*, Anchorage, AK, USA, May 2010, pp. 4736–4741.

[12] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime search in dynamic graphs," *Artif. Intell.*, vol. 172, no. 14, pp. 1613–1643, May 2008.

[13] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Proc. Adv. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, 2003, pp. 52–89.

[14] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 1994, pp. 3310–3317.

[15] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 1995, pp. 1652–1659.

[16] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 354–363, Jun. 2005.

[17] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A*," *Artif. Intell.*, vol. 155, nos. 1–2, pp. 93–146, May 2004.

[18] C. de Boor, *A Practical Guide to Splines* (Applied Mathematical Sciences). Viterbo, Italy: Univ. Tuscia, 1978.

[19] P. Dierckx, *Curve and Surface Fitting with Splines*. New York, NY, USA: Clarendon, 1995, pp. 63–80.

[20] A. Piazzi and A. Visioli, "Global minimum-jerk trajectory planning of robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 47, no. 1, pp. 140–149, Feb. 2000.

[21] G. Song and N. M. Amato, "Randomized motion planning for car-like robots with C-PRM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Maui, HI, USA, Oct./Nov. 2001, pp. 37–42.

[22] R. Buchanan, T. Bandyopadhyay, M. Bjelonic, L. Wellhausen, M. Hutter, and N. Kottege, "Walking posture adaptation for legged robot navigation in confined spaces," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 2148–2155, Apr. 2019.

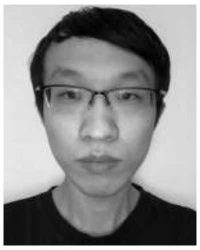
[23] A. Elfes, R. Steindl, F. Talbot, F. Kendoul, P. Sikka, T. Lowe, N. Kottege, M. Bjelonic, R. Dungavell, T. Bandyopadhyay, M. Hoerger, B. Tam, and D. Rytz, "The multilegged autonomous explorer (MAX)," in *Proc. IEEE Int. Conf. Robot. Automat.*, May/Jun. 2017, pp. 1050–1057.

[24] C. Semini, V. Barasuol, J. Goldsmith, M. Frigerio, M. Focchi, Y. Gao, and D. G. Caldwell, "Design of the hydraulically actuated, torque-controlled quadruped robot HyQ2Max," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 2, pp. 635–646, Apr. 2017.

[25] A. Short and T. Bandyopadhyay, "Legged motion planning in complex three-dimensional environments," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 29–36, Jan. 2018.

[26] M. Bjelonic, T. Homberger, N. Kottege, P. Borges, M. Chli, and P. Beckerle, "Autonomous navigation of hexapod robots with vision-based controller adaptation," in *Proc. IEEE Int. Conf. Robot. Automat.*, May/Jun. 2017, pp. 5561–5568.

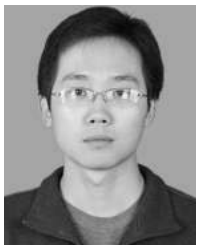
[27] P. Fankhauser, M. Bjelonic, C. D. Bellicoso, T. Miki, and M. Hutter, "Robust rough-terrain locomotion with a quadrupedal robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2018, pp. 1–8.



LI ZHE received the bachelor's and master's degrees from the Shandong University of Science and Technology, China, in 2008 and 2014, respectively. He is currently pursuing the Ph.D. degree with the School of Control Science and Engineering, Shandong University, China. His research interest includes the environmental perception and path planning for quadruped robot.

LI YIBIN received the bachelor's degree from Tianjin University, China, in 1982, the master's degree from the Shandong University of Science and Technology, China, in 1988, and the Ph.D. degree from Tianjin University, in 2006. He is currently a Professor and the Associate Dean with the School of Control Science and Engineering, Shandong University, China. His research interests include robotics, mechatronics, intelligent control, and intelligent vehicles.

RONG XUEWEN received the bachelor's and master's degrees from the Shandong University of Science and Technology, China, in 1996 and 1999, respectively. He is currently pursuing the Ph.D. degree with the School of Control Science and Engineering, Shandong University, China, where he is also a Senior Engineer. His research interests include robotics, mechatronics, and hydraulic servo driving technology.



ZHANG HUI received the bachelor's and master's degrees from Shandong Agricultural University, China, in 2009 and 2012, respectively, and the Ph.D. degrees from Shandong University, China, in 2016. He is currently a Lecturer with the School of Electrical Engineering and Automation, Qilu University of Technology (Shandong Academy of Sciences), China. His research interest includes the environmental perception and path planning for quadruped robot.