Multi-UAV Adaptive Path Planning in Complex Environment Based on Behavior Tree

Wendi Wu[1], Jinghua Li[2], Yunlong Wu[2,3(✉)], Xiaoguang Ren[2,3(✉)], and Yuhua Tang[1]

[1] State Key Laboratory of High Performance Computing (HPCL), College of Computer, National University of Defense Technology, Changsha 410073, Hunan, China
[2] Tianjin Artificial Intelligence Innovation Center (TAIIC), Tianjin 300457, China
[3] Artificial Intelligence Research Center (AIRC), National Innovation Institute of Defense Technology (NIIDT), Beijing 100071, China
ylwu1988@nudt.edu.cn, rxg_nudt@126.com

Abstract. In this paper, we consider a scenario where multiple tracking unmanned aerial vehicles (UAVs) pursue a target UAV in a complex environment. Consider the fast airspeed of the UAV, the path planning needs to be finished in a limited time. Moreover, the complex environment may involve diverse geographical areas, which raises the challenges for the path planning algorithms. For the first challenge, we will adopt the real-time algorithms to keep the efficiency of path planning. For the challenge of environment diversity, we involve the behavior tree (BT) model and propose a BT-organized path planning (BT-OPP) method aiming at achieving adaptive scheduling of different path planning algorithms in different geographical areas. Furthermore, in order to take the advantages of multiple tracking UAVs, we propose a virtual-target-based tracking (VTB-T) method which can make the tracking UAVs pursue the target UAV collaboratively. The effectiveness of the proposed BT-OPP method and the VTB-T method are verified by analysis and numerical results for different system configurations, showing that a substantial target tracking efficiency improvement may be achieved in comparison with the benchmark.

Keywords: Multi-UAV target tracking · Behavior tree · Real-time path planning

This work was supported in part by the National Key Research and Development Program of China under Grant No. 2017YFB1301104, and in part by the National Natural Science Foundation of China under Grant No. 61906212 and Grant No. 61802426.

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 350, pp. 494–505, 2021.
https://doi.org/10.1007/978-3-030-67540-0_32

1 Introduction

In recent years, the rapid development of unmanned aerial vehicles (UAVs) related technology has made it widely used in various fields such as industry, security, military, and scientific research [1]. Particularly, in the military field, UAVs have replaced manned aircraft in large numbers of important tasks such as intelligence, surveillance, and reconnaissance (ISR) [2]. As the decreasing manufacturing cost of the UAVs, it has led to their wide use in civil and commercial applications such as area exploration, surveillance, package delivery, etc. [3]. In UAV applications, target tracking as a typical task, is often applied to the national security and other aspects. The task requires the UAVs to face a complex environment for path planning in order to achieve a more efficient target tracking, and the path planning algorithm may face two main challenges. Firstly, the target moves fast and the UAV needs to respond in a limited time and plan a feasible path in real-time. Secondly, the environment is diverse, and the UAV may pass through many different geographical areas during the process of target tracking, which requires different area-oriented path planning algorithms can switch adaptively.

Traditional path planning algorithms are difficult to meet the real-time requirements, for example, the Dijkstra algorithm is a classical algorithm for single source path planning which adopts the breadth-first search to find the path [4]. The Dijkstra algorithm will eventually get a shortest path, but it is not a real-time algorithm. There are also more algorithms currently being researched for the real-time challenge. Based on the theory of Dijkstra algorithm, the A* algorithm introduces a heuristic search which can largely improve the efficiency of path planning [5,6]. The D* algorithm proposed in 1994 is an effective path planning in the unknown and dynamic environment, and only checks changes in the shortest path to the next or adjacent node when moving towards the target point in a reverse search mode. Based on the D* algorithm, Koenig proposed the D*Lite algorithm which can achieve fewer re-planning times and faster response to sudden obstacles. In 2004, Likhachev et al. implemented the ARA* algorithm which added the idea of anytime based on the A* algorithm [7]. When the time constraint is relaxed, the global optimal path can be obtained. On the contrary, if the time constraint is limited, a time-related sub-optimal path will be obtained. The anytime algorithm can allow us to obtain a planned path under any time constraint. In 2005, combined the ARA* algorithm with the D*Lite algorithm, the AD* algorithm is proposed to realize anytime path planning for unknown areas [8]. In 2012, Sun et al. designed the I-ARA* algorithm which runs on the basis of the ARA* algorithm [9]. In the I-ARA* algorithm, the planned results of the last iteration is efficiently reused, and the path planning for dynamic targets is finally realized through the idea of re-planning in motion.

To address the second challenge of environment diversity, a single algorithm cannot be adapted to multiple situations, and require a mechanism for combining with different algorithms to achieve good results. Therefore, an appropriate robot control architecture needs to be considered to integrate multiple path planning algorithms. In traditional robot control architecture, finite state machine (FSM),

hierarchical finite state machine (HFSM), subsumption model, and decision tree can realize the robot controlling, but the above control architectures have certain drawbacks in the four aspects: implementability, maintainability, scalability, and reusability [10]. Behavior tree (BT), as a tree model consisting of hierarchical nodes, is used to construct robot behaviors for adapting to the switching among tasks [11]. BT is an effective method for creating modular and reactive complex systems, and can solve the challenge of environment diversity by constructing control nodes and execution nodes to decompose complex tasks into sub-tasks [12,13].

Based on the above discussion, in order to address the problem of target tracking in the complex environment, the two above challenges need to be simultaneously considered. Therefore, in this paper, we propose to implement the adaptive scheduling of different real-time path planning algorithms based on the BT model. The main contributions of this paper are as follows:

- A multi-UAV target tracking scenario is considered which involves a complex environment containing no obstacle area, known area and unknown area. When the tracking UAVs catch up with the target UAV, or the target UAV leaves a specified area, the tracking task is finished.
- In order to overcome the challenges of real-time and diversity brought by the above scenario, this paper proposes a BT-organized path planning (BT-OPP) method which can achieve the adaptive scheduling of different path planning algorithms for different areas.
- In order to fully take the advantages of multiple tracking UAVs, this paper proposes a virtual-target-based tracking (VTB-T) method which can significantly increase the success rate of target tracking.
- The effectiveness of the proposed method is verified through simulation experiments. From the experimental results, it shows that the BT-OPP method and the VTB-T method can largely improve the target tracking efficiency.

The rest of the paper is organized as follows. Firstly, the task scenario considered by this paper is presented in Sect. 2. Then, Sect. 3 and Sect. 4 propose the details of the BT-OPP method and the VPB-T method. Simulation results are provided and discussed in Sect. 5. Finally, we conclude this paper in Sect. 6.

2 Scenario Description

In this paper, we consider a multi-UAV tracking task in the complex environment, as shown in Fig. 1. In this scenario, multiple tracking UAVs catch up with a target UAV which flies through multiple diverse areas. We divide the environment into three areas which are no obstacle area, known area, and unknown area, respectively. In the no obstacle area, the path planning algorithm does not need to consider the influence of obstacles, and the tracking UAV can fly straight to the target UAV. In the known area, the positions of obstacles are known and can be considered into path planning. For the unknown area, the obstacles are not completely known or temporarily changes, the path planning algorithm needs

to fully considers the effects of unknown obstacles. We model the environment as a two-dimensional grid map and assume that the movement of the UAV is omnidirectional. For example, when the UAV has an unit airspeed, the moving distance of the UAV in each time step is fixed as a grid. In addition, this paper further assumes that UAVs are flying at a fixed altitude.

Fig. 1. A scenario where multiple UAVs track a target UAV which flies through a complex environment.

At the initial moment, the tracking UAVs are located at the starting point of the leftmost area, and the target UAV is located at a certain position in no obstacle area. When the mission is received, the tracking UAVs begin to plan the path with the current position of the target UAV, and starts to move after the calculation is completed. The target UAV flies from left to right from the current position, passing through the three areas in turn. The tracking UAVs update the position of the target UAV in each time step, and then plan the new path. In the no obstacle area, the tracking UAVs may fly straight to the position of the target UAV. When the target UAV flies into the known area, the tracking UAVs will switch to use a more adaptive path planning algorithm, such as the I-ARA* algorithm. Similarly, if the target UAV files from the known area to the unknown area, the tracking UAVs may use an algorithm which is suitable for the unknown-area path planning, such as the D*Lite algorithm. If the target UAV successfully passes the whole area, the tracking task is considered to fail.

3 BT-Organized Path Planning (BT-OPP) Method

3.1 BT Design

In order to realize the adaptive path planning in complex environment, this paper adopts the BT model to organize the path planning algorithms which meets different requirements of the environment. The design of the BT structure is shown in Fig. 2. The ellipse denotes the condition node and the blue rectangle represents the action node. Starting from the root node, the BT firstly ticks the condition node that judge whether the target UAV has been successfully caught. If the target UAV fails to be caught (return failure), it will traverse to the child node to determine whether the target UAV is in the area under consideration. If it is not in the effective area, it returns failure. If the target UAV is in the effective area, its position will be obtained and transmitted to the

tracking UAVs. When in the no obstacle area, tracking UAVs can fly straight to the target UAV. Otherwise, the I-ARA* algorithm or D*Lite algorithm will be adopted adaptively. The BT iterates through all nodes every 1 s until the end of the target tracking task.

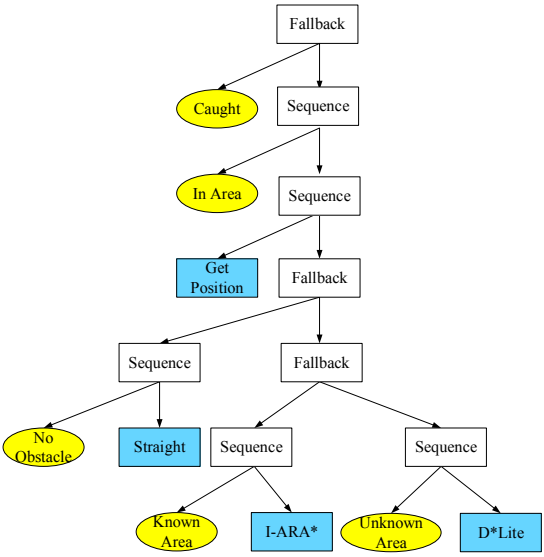


Fig. 2. BT design that organizes three different path planning algorithms. (Color figure online)

3.2 Action Node Design

As shown in Fig. 2, the action nodes carry out path planning algorithms according to specific geographical areas. The *Straight* action node controls the tracking UAV to fly straight to the target UAV. The calculation cost of path planning is almost negligible, and the planned path is highly efficient. For the *I-ARA** action node, it will load the I-ARA* algorithm and provide a feasible path under anytime limit. The longer the time, the better the path, so that the real-time performance of the algorithm can be guaranteed. It repeatedly uses the planned results and reduces the computational cost of path planning as much as possible. Each time the BT is scheduled to the I-ARA* action node, a new starting position and target position will be assigned to it. The I-ARA* algorithm may use the information after the last calculation to re-plan a new path, realizing target tracking in the known and dynamic environment.

In the unknown area, the environment is not completely known by the tracking UAVs, which may leads to the appearance of unexpected obstacles. At this time, the *D*Lite* action node is selected. The D*Lite algorithm is used to deal with the path planning in unknown environment. The core of the D*Lite algorithm is assuming that the unknown area is all free space. On this basis, the path

planning is implemented in an incremental way, and the shortest distance to the target position is found in a limited time. The traditional D*Lite algorithm is a path planning algorithm for static targets, while the improved D*Lite algorithm can re-plan a new path to a dynamic target every time BT is traversed.

4 Virtual-Target-Based Tracking (VTB-T) Method

In order to take the advantages of multiple UAVs, we propose a virtual-target-based tracking (VTB-T) method. This method is based on the idea of multiple hunters "rounding up" a prey. One hunter is selected as the main hunter, and the remaining hunters plan their paths according to the possible positions of the prey at the next moment. The possible positions are called virtual targets by our method and its distribution diagram is shown in Fig. 3.

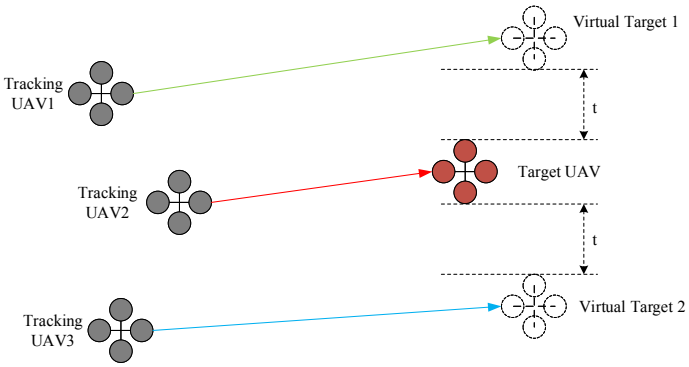


Fig. 3. Virtual targets selection in multi-UAV target tracking task.

As shown in Fig. 3, UAV1, UAV2 and UAV3 move towards the target UAV. At the beginning, UAV2 is closest to the target UAV and acts as the main tracking UAV. UAV1 and UAV3 will plan the paths for two virtual targets, respectively. In each time step, the distance between the tracking UAVs and the target UAV is calculated. If the shortest distance between the tracking UAVs and the target UAV changes, the corresponding main tracking UAV also needs to change, and the remaining tracking UAVs carries out path planning for the virtual targets. Otherwise, the tracking state remains unchanged.

In our considered scenario, each tracking UAV will be allocated a serial number i, $i \in [1, N]$, where N represents the number of tracking UAVs. The multi-UAV target tracking mainly focuses on selecting the best virtual target allocation for each tracking UAV. The main steps are shown in Algorithm 1. The inputs of the algorithm are the current positions of the tracking UAVs $p = \{p_1, p_2, \ldots, p_N\}$, where $p_i = (x_i, y_i), x_i, y_i \in \mathbb{R}, i \in [1, N]$ denotes the position of the tracking UAV i, and x_i, y_i represent its positions on the x-axis and y-axis. $p_{target} = (x_{target}, y_{target})$ denotes the position of the target UAV, and w,

h are the width and height of the grid map G, respectively. The outputs of the algorithm are $q = \{q_1, q_2, \ldots, q_n\}$, the planned positions of the virtual targets, where $q_i = (x_i, y_i), x_i, y_i \in \mathbb{R}, i \in [1, N]$ represents the virtual target of the tracking UAV i.

Algorithm 1. Virtual-Target-Based Tracking (VTB-T) Method

Input: $p = \{p_1, p_2, ..., p_N\}$: position vector of the tracking UAVs, $p_i = (x_i, y_i)$; N: number of tracking UAVs; $p_{target} = (x_{target}, y_{target})$: the target UAV position; G: grid map; h: height of map G; w: width of map G; t: interval.
Output: $q = \{q_1, q_2, ..., q_N\}$: the positions of the virtual targets.

1: $k_{min} = \text{CALCULATENEARESTUAV}(p, p_{target})$
2: $q_{k_{min}} = p_{target}$
3: $N_{above} = \text{CALCULATETARGETNUMBER}(N, p_{target}, h, w)$
4: **for** $i \in [1, N]$ **and** $i \neq k_{min}$ **do**
5: $\quad x = x_{target} + |N_{above} - i| - 1$
6: $\quad y = y_{target} + t(i - N_{above})$
7: $\quad$ **while** $G[x][y]$ is obstacle and $x < w$ **do**
8: $\quad\quad x = x + 1$
9: $\quad$ **end while**
10: $\quad q_i = (x, y)$
11: **end for**

In the beginning of Algorithm 1, we need to find the nearest tracking UAV to the target UAV which is calculated by function *CalculateNearestUAV()*, and $k_{\min}$ represents the index of the nearest tracking UAV. The details of *CalculateNearestUAV()* is shown in Algorithm 2. On line 2 of Algorithm 1, $q_{k_{\min}}$ represents the position of the target UAV. In addition to the tracking UAV $k_{\min}$, there are still $N - 1$ remaining UAVs need to be assigned virtual targets, which will be processed by function $CalculateTargetNumber()$ implemented by Algorithm 3.

Algorithm 2. Calculate the nearest tracking UAV

Input: $p = \{p_1, p_2, ..., p_N\}$: position vector of the tracking UAVs, $p_i = (x_i, y_i)$; N: number of tracking UAVs; $p_{target} = (x_{target}, y_{target})$: the target UAV position.
Output: k_{min}: the index of the nearest tracking UAV.

1: **function** $\text{CALCULATENEARESTUAV}(p, p_{target})$
2: $\quad d_{min} = \infty$, $k_{min} = -1$
3: $\quad$ **for** each $p_i \in p$ **do**
4: $\quad\quad$ Calculate the distance d_i between the p_i and p_{target}
5: $\quad\quad$ **if** $d_i < k_{min}$ **then**
6: $\quad\quad\quad k_{min} = i$
7: $\quad\quad$ **end if**
8: $\quad$ **end for**
9: $\quad$ **return** k_{min}
10: **end function**

In order to achieve the rounding effect of multiple tracking UAVs, the virtual targets need to be placed above and below the target UAV. On lines 2 and 3 of Algorithm 3, half of the tracking UAVs are allocated above the target UAV, and the other half are allocated below. On line 3, the tracking UAV that is nearest to the target UAV is excluded. On lines 4 to 10 of Algorithm 3, if the upper or lower boundaries of the map are encountered, the upper and lower UAVs between target UAV increase or decrease by the same amount.

Algorithm 3. Calculate the number of virtual targets above and below the target UAV

Input: N: number of tracking UAVs; $p_{target} = (x_{target}, y_{target})$: the target UAV position; $height$: height of map; $width$: width of map.
Output: N_{above}: number of virtual targets above.

1: **function** CALCULATETARGETNUMBER($N, p_{target}, height, width$)
2: $N_{above} = \lfloor \frac{N}{2} \rfloor$
3: $N_{below} = N - N_{above} - 1$
4: **if** $N_{above} > y_{target}$ **then**
5: $N_{below} = N_{below} + N_{above} - y_{target}$
6: $N_{above} = y_{target}$
7: **end if**
8: **if** $N_{below} > height - y_{target} - 1$ **then**
9: $N_{above} = N_{above} + N_{below} - height + y_{target} + 1$
10: $N_{below} = height - y_{target} - 1$
11: **end if**
12: **return** N_{above}
13: **end function**

Back to the step of Algorithm 1, after arranging the number of virtual targets above and below the target UAV, we need to allocate the positions of the virtual targets. On lines 4–9 of Algorithm 1, the tracking UAV $k_{\min}$ that has been allocated is excluded, leaving $N - 1$ tracking UAVs to be allocated. Lines 5 and 6 calculate the virtual target i between top and bottom of the target UAV. Lines 7 and 8 indicate that when the allocated virtual target is an obstacle, the position of the current virtual target is shifted to the right by one unit distance until it is free. If the allocated virtual target reaches the right boundary of the map, it will be fixed at the right boundary. Finally, line 10 sets the corresponding virtual targets to each tracking UAV.

5 Simulation Results

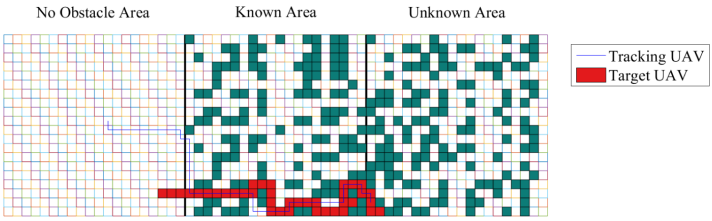
In this section, we may verify the effectiveness of our proposed methods. Firstly, in Subsect. 5.1, we may give the comparison between the BT-organized path planning (BT-OPP) method and the original path planning (OPP) method.

In order to take advantages of the multi-UAV system, Subsect. 5.2 will test the validation of our virtual-target-based tracking (VTB-T) method.

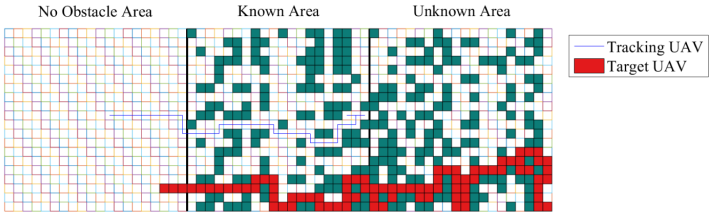
5.1 The Performance Comparison Between the BT-OPP and OPP Methods

In order to verify the effectiveness of the BT-OPP method, we consider a complex scenario where the target UAV flies through a complex environment, as shown in Fig. 1. The first area is a no obstacle area which can be seen as an open field. The second area is a known and obstacle area. In the third area, it has obstacles but unknown. In this scenario, the tracking UAV will pursue the target UAV until it gets to the end position of the map.

In order to verify the tracking performance of the BT-OPP method, we may select the D*Lite algorithm as the OPP method and the benchmark. For the BT-OPP method, straight, I-ARA*, and D*Lite algorithms are used to plan the paths for different areas, respectively. A trajectory comparison between the BT-OPP and OPP methods are shown in Fig. 4(a) and Fig. 4(b), respectively. In this testing, the airspeed of the tracking UAV is the same as that of the target UAV. From the result, we may see that the length of the planned path by the BT-OPP method is apparently shorter than the OPP method. Furthermore, the tracking UAV with the BT-OPP method finally catches up with the target UAV successfully, but the OPP method may not.



(a) The BT-OPP method.



(b) The OPP method.

Fig. 4. Trajectories planned by the BT-OPP method and the OPP method.

We may further analyze the efficiency of the BT-OPP and OPP methods. Let d and a denote the initial distance and airspeed ratio between the tracking

UAV and the target UAV, respectively. We may change d and a, then compare the final distance between the tracking UAV and the target UAV, as shown in Fig. 5. If the final distance is greater than 0, it means the target UAV can not be caught. From the figure, we may see that the final distance increases with the increase of the initial distance, but the tracking efficiency of the BT-OPP method is obviously higher than that of the OPP method. When $a \geq 1.5$, the final distance of the BT-OPP method was always 0, which means the target UAV can be always caught.

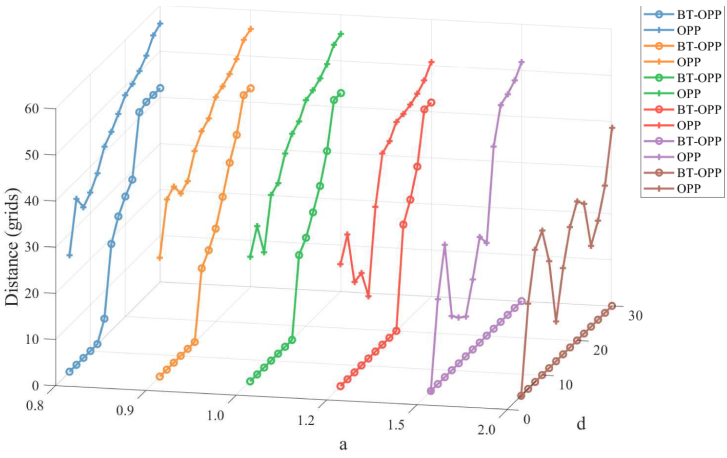
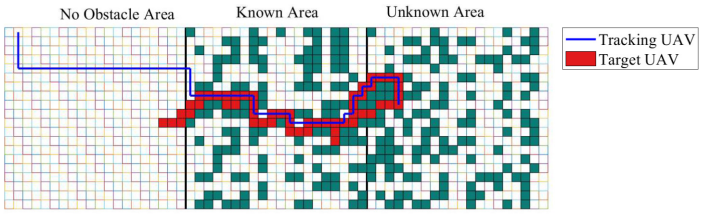


Fig. 5. Final distance comparison between the BT-OPP and the OPP methods in different settings.

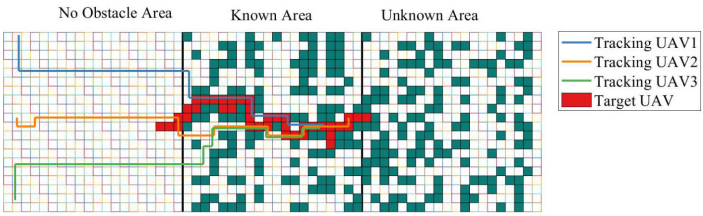
5.2 Verify the Optimization Effects of the VTB-T Method

In this paper, a multi-UAV target tracking algorithm called VTB-T method is proposed, which aims at increasing the chance to catch up with the target UAV. We compare the flight trajectories under the BT-OPP methods with a single tracking UAV (Fig. 6(a)) and multiple tracking UAVs (Fig. 6(b)). In the multi-UAV case, the virtual target distribution is calculated by the VTB-T method.

In order to verify the optimization effect of the multiple tracking UAV case and the single tracking UAV case, this experiment designs the path planning efficiency comparison of the two cases. As shown in Fig. 7, the x-axis is the airspeed ratio of the tracking UAV to the target UAV, the y-axis is the time step index, and the z-axis is the time steps that the target UAV being caught. According to the results, it can be seen that under the same number of tracking UAVs, the tracking efficiency increases with the increase of a. Furthermore, with the same a, the more tracking UAVs there are, the faster the target will be caught.



(a) The single tracking UAV case.



(b) The multiple tracking UAV case.

Fig. 6. Trajectory planned by the BT-OPP method with a single tracking UAV and multiple tracking UAVs.

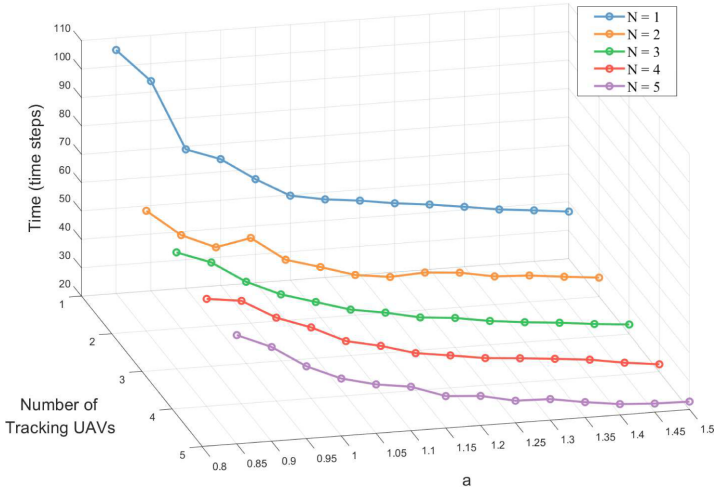


Fig. 7. Tracking efficiency comparison in different system settings.

6 Summary

This paper considered a multi-UAV target tracking scenario in a multi-geographical area, which involved the challenges of real-time path planning and environment diversity. We proposed to use BT to solve the two challenges, and introduced the BT-OPP method which could automatically select the appropri-

ate path planning algorithms according to the environment conditions. Moreover, in order to increase the tracking performance, we proposed the VTB-T method that fully utilized the advantages of multiple tracking UAVs. The simulation results showed that the BT-OPP method could largely improve the target tracking performance more than the classical OPP method. When involved with the VTB-T method, the BT-OPP could further decrease the tracking time.

References

1. Wallace, L., Lucieer, A., Watson, C.S., Turner, D.: Development of a UAV-LiDAR system with application to forest inventory. Remote Sens. **4**(6), 1519–1543 (2012)
2. Spedicato, S., Notarstefano, G., Bülthoff, H.H., Franchi, A.: Aggressive maneuver regulation of a quadrotor UAV. In: Inaba, M., Corke, P. (eds.) Robotics Research. STAR, vol. 114, pp. 95–112. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-28872-7_6
3. Wu, Y., Zhang, B., Yang, S., Yi, X., Yang, X.: Energy-efficient joint communication-motion planning for relay-assisted wireless robot surveillance. In: Proceedings of IEEE Conference on Computer Communications, pp. 1–9. IEEE (2017)
4. Dijkstra, E.W., Dijkstra, E.W., Dijkstra, E.W., Informaticien, E.U., Dijkstra, E.W.: A Discipline of Programming. Prentice-Hall, Englewood Cliffs (1976)
5. Zhang, H., Cheng, M.: Path finding using A* algorithm. Microcomput. Inf. **23**(6), 238–241 (2007)
6. Likhachev, M., Ferguson, D., Gordon, G., Stentz, A., Thrun, S.: Anytime dynamic A*: an anytime, replanning algorithm. In: Proceedings of International Conference on Automated Planning and Scheduling, pp. 262–271. AAAI Press (2005)
7. Likhachev, M., Gordon, G., Thrun, S.: ARA*: anytime A* with provable bounds on sub-optimality. In: Advances in Neural Information Processing Systems 16, pp. 767–774. MIT Press (2004)
8. Aine, S., Likhachev, M.: Anytime truncated D*: anytime replanning with truncation. In: Proceedings of Annual Symposium on Combinatorial Search, pp. 2–10. AAAI Press (2013)
9. Sun, X., Yeoh, W., Uras, T., Koenig, S.: Incremental ARA*: an incremental anytime search algorithm for moving-target search. In: Proceedings of International Conference on Automated Planning and Scheduling, pp. 243–251. AAAI Press (2012)
10. Hu, D., Gong, Y., Hannaford, B., Seibel, E.J.: Semi-autonomous simulated brain tumor ablation with RAVENII surgical robot using behavior tree. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 3868–3875. IEEE (2015)
11. Paxton, C., Hundt, A., Jonathan, F., Guerin, K., Hager, G.D.: CoSTAR: instructing collaborative robots with behavior trees and vision. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 564–571. IEEE (2017)
12. Gershenson, J., Prasad, G., Zhang, Y.: Product modularity: definitions and benefits. J. Eng. Des. **14**(3), 295–313 (2003)
13. Wu, Y., Ren, X., Zhou, H., Wang, Y., Yi, X.: A survey on multi-robot coordination in electromagnetic adversarial environment: challenges and techniques. IEEE Access **8**, 53484–53497 (2020)