IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 March 2023, accepted 3 May 2023, date of publication 8 May 2023, date of current version 12 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3274204

RESEARCH ARTICLE

An Evolutionary Navigation Algorithm for Multi-Robot With Priority Order

SHENG-KAI HUANG AND WEN-JUNE WANG, (Life Fellow, IEEE)
Department of Electrical Engineering, National Central University, Taoyuan 32001, Taiwan
Corresponding author: Wen-June Wang (wjwang@cc.ncu.edu.tw)

This work was supported by the Ministry of Science and Technology of Taiwan under Grant 111-2221-E-008-107.

ABSTRACT This study proposes a novel multi-robot navigation algorithm with priority order called, in short, PONA2.0. This algorithm is based on the generalized Voronoi diagram and contains an adjustable multipath switching mechanism and a collision prevention strategy, such that the arrival order of robots is in line with the priority order as much as possible, and the average trajectories length is as short as possible. The given average trajectories length of all robots (*ATLA*) and arrival order (*AO*) are used to be the two performance indices for the comparison between the proposed algorithm and recent existing algorithms NSPP (Huang et al., 2021), PONA (Huang et al., 2022), ROA, and SDA (Ali et al., 2016). The comparison shows that the PONA2.0 can reduce the average *AO* by more than 56% compared with NSPP and PONA and reduce the average *ATLA* between 5% and 17% compared with ROA and SDA.

INDEX TERMS Voronoi diagram, Yen's algorithm, multi-robot path planning, collision-free, path-priority order.

I. INTRODUCTION

In smart manufacturing, different types of robots are widely used in commercial and civilian fields. One of the most important problems in smart manufacturing is how to have very efficient transportation within warehousing. It is known that different goods in the warehouse may have different priorities for being transported. Most times, we need multiple robots to carry various goods moving in the warehouse or factory. Multiple robots moving in a space must avoid obstacles in the space and avoid collision with each other. Therefore, the navigation plan for multiple robots moving in space is worth studying the problem.

In general, the problem of multi-robot path planning in a space with obstacles has two cases, one is that each robot is without priority order and the other is with priority order. The case without priority has been studied in many papers, such as the following papers. A centroidal Voronoi tessellation (CVT) based path planning algorithm for self-assembly robots was proposed in [1] in which swarm robots move collaboratively from the initial area to the target area. A hierarchical model predictive control (HMPC) to solve the multi-robot navigation problem and provide theoretical results to demonstrate its stability and feasibility was proposed by [2]. A methodology for optimizing multi-robot paths using an improved gravitational search algorithm (IGSA) in dynamic environments was proposed in [3], which showed that IGSA has a better performance compared to the gravitational search algorithm (GSA) and particle swarm optimization (PSO). The authors in [4] proposed task allocation by using the genetic algorithm (GA) and path planning by using A* algorithm [6] for three robots in a common work area. The authors in [5] proposed an efficient artificial bee colony (EABC) algorithm for online multi-robot path planning such that each robot can reach the target position without collision.

Next, let us consider the case of the robots with priority order. Some studies dynamically adjusted the robot's priority order to achieve the task of multi-robot path planning. In [7], the randomized search with hill-climbing was used to find the optimal priority order for each robot to solve the coordination of multi-robot motions. In [8], the authors adjusted the priority order of robots by using a heuristic method to solve the dynamic conflict problem in a multi-robot system for cooperative path planning. If the priority order of each robot is

The associate editor coordinating the review of this manuscript and approving it for publication was P. Venkata Krishna.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

assigned based on its task importance, there have been several studies investigating the multiple robots' navigation problem. For instance, in paper [9], the navigation strategy with path priority (NSPP) was proposed such that robots with higher path priority have a shorter path than robots with lower path priority. A priority order navigation algorithm (PONA) was proposed by [10] to solve the problem of NSPP and give a shorter average travel length for all robots, but the target arrival order is not ensured to match the priority order. The paper [11] introduced an algorithm to implement cooperative path planning for multiple robots and dynamically adjusted the priority of robots by using the last path length constraint. In [12], they proposed a new algorithm to solve the problem, which is difficult to be handled by classical prioritized path planning algorithms. In [13], the authors proposed a modified Dijkstra A* algorithm to solve the multi-robot navigation problem in automated storage and retrieval systems (ASRS) so that the highest priority robot can have the shortest path and free collision with other robots. In [14], deterministic rescheduling and start-safe intervals techniques were used to solve the initial priority path planning failures problem.

In this paper, we will propose a new priority order navigation algorithm (PONA2.0) that improves the result of PONA in [10] and ensures each robot can reach its target in a two-dimensional flat space without any collision efficiently. We also suppose the speed of all robots is the same, and there is no-slip between the tire of the robot and the floor during motion. This paper is organized as follows: Section II reviews some preliminary background for the main algorithm. Section III explains the proposed main algorithm. The comparison between the proposed and other benchmark algorithms is described in Section IV. A conclusion is given in Section V.

II. PRELIMINARY BACKGROUND

In this section, we will review some necessary background before we design the main algorithm.

A. INPUT MAP AND PRIORITY ORDER

We consider that there are multiple robots moving in a flat space of a factory or a warehouse. This flat space is called the ''input map'' for the algorithm design. For instance, Fig. 1 shows three robots and five fixed obstacles. Suppose the input map with several known fixed objects or/and dynamic (such as other moving robots) obstacles. Each robot may be a type of differential wheeled robot. In this study, each robot has its priority order which depends on the order of the robot's task importance. The robot with higher priority moves and reaches its target point needs shorter paths and less time spent than the robot with lower priority. The lower priority robot regards the higher priority robot as a dynamic obstacle to be avoided, but not vice versa. It is noted that each robot has its map from its viewpoint. For instance, Fig. 2 is the map from the viewpoint of robot-2 (denoted as R2), where R2 regards Robot-1 (R1, the black square) as a dynamic obstacle. Fig. 3 is the viewpoint of robot-3 (R3), where R3 regards R1 and R2

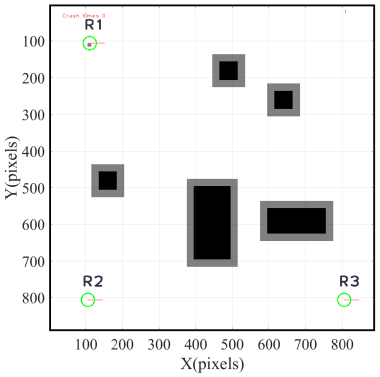

FIGURE 1. An example of the input map.

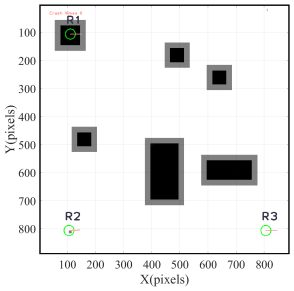

FIGURE 2. The map from the viewpoint of R2.

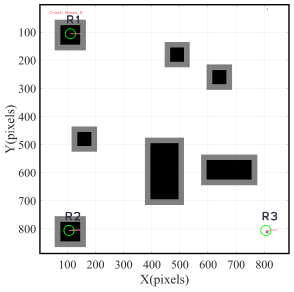

FIGURE 3. The map of R3.

as dynamic obstacles. The gray boundary of each black square shown in Fig. 2 and Fig. 3 is the dilated region which is the buffer area for avoiding being hit by other moving robots.

B. GENERALIZED VORONOI DIAGRAM GENERATION

In the map of the viewpoint of R2, we generate many red dots (called Voronoi corners) around the gray boundaries of all obstacles and around the map as shown in Fig. 4 (see [9]). Then, a generalized Voronoi diagram (called the GVD map) is generated by the method in [15] as shown in Fig. 5. Any intersection points or turning points of red lines is called the navigation point (NP). The robot will move along the line segments between any two neighbor NPs.

III. MAIN ALGORITHM FOR PATH PLANNING

A. ADJUSTABLE MULTIPATH SWITCHING MECHANISM

A path is composed of several line segments. The starting and target positions of each robot are initially set. There may

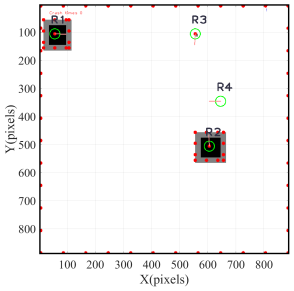

FIGURE 4. Red dots denote Voronoi corners.

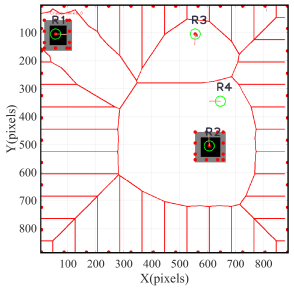

FIGURE 5. The generalized Voronoi diagram of robot-3.

be many possible paths from an initial position to reaching the target. We use Yen's algorithm [17] to find all feasible paths for each robot between its starting and target positions. As shown in Fig. 6, if the starting position and target of R2 are specified, there are several paths composed of blue navigation links (denoted as B1-B5) to reach the target (denoted as R2T in the figure). Yen's algorithm can sort all possible paths according to their lengths. The first shortest path (denoted as SP-1) is composed of B1+B4+B5 in blue color, the second shortest path (denoted as SP-2) is composed of B2+B4+B5 in pink color, and the third shortest path (denoted as SP-3) is composed of B3+B5 in cyan color shown in Fig. 7. Actually, a robot may have more than three shortest paths (e.g., SP-4, SP-5...) but only the three shortest paths are listed in the figure for clarity.

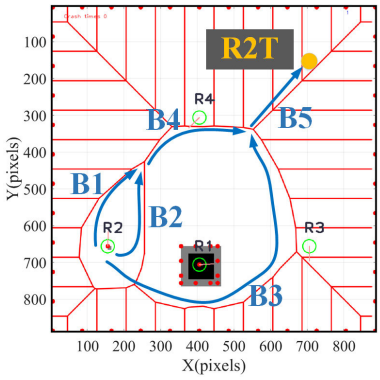

FIGURE 6. The three feasible paths for R2.

Frankly, neither SP-1, SP-2, nor SP-3 is the real shortest path if the paths must be along the navigation links. Using

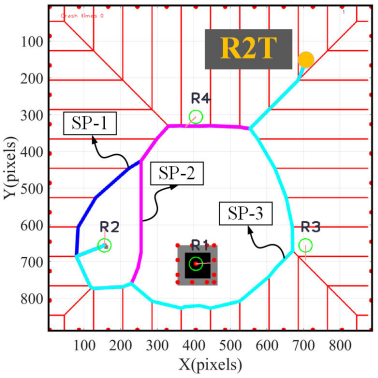

FIGURE 7. SP-1, SP-2, and SP-3 for R2 by Yen's algorithm.

the method of [10], for any robot, we can find a shorter path MSP $-\lambda$ which is shorter than SP $-\lambda$ corresponding to different λs as shown in Fig. 8, where the green paths are called the modified SP-1 (denoted as MSP-1), the modified SP-2 (denoted as MSP-2), and the modified SP-3 (denoted as MSP-3); besides, SP-1 and SP-2 are shortened to the same path MSP-1 or MSP-2 for R2. In Fig. 9, SP-1, SP-2, and SP-3 are shortened to the MSP-1, MSP-2, and MSP-3 for R4, respectively.

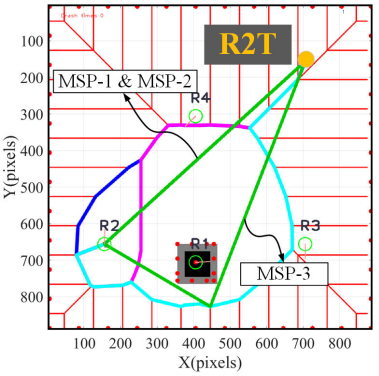

FIGURE 8. The MSP-1 (MSP-2) and MSP-3 for R2.

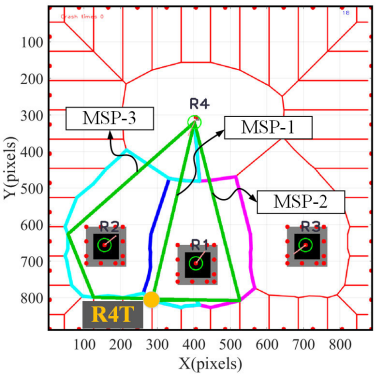

FIGURE 9. The MSP-1, MSP-2, and MSP-3 for R4.

In PONA [10], the authors only considered two MSPs, *i.e.*, MSP-1 and MSP-2. There will be a problem below.

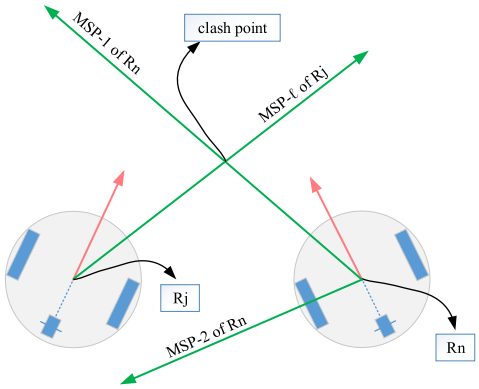

FIGURE 10. The MSPs of two robots [10].

When the MSP-1 of a low-priority robot and the MSP-1 of a high-priority robot have a clash point, it is possible that there will be a clash between the two robots. If two robots' MSPs have a clash point, two robots probably collide with each other. When the MSP switching mechanism [10] of a low-priority robot is satisfied, e.g., the low-priority robot is closer to the clash point than a high-priority robot, the low-priority robot will switch to MSP-2 to avoid colliding with the high-priority robot as possible. When both MSP-1 and MSP-2 have clash points with high-priority robots, [10] decided which robot should wait for a while. The simulation results show that fewer robots (less than five) can have a good performance (the arrival order is in line with the priority order of robots), but the performance is not good with eight robots. This is because even if the low-priority switch to MSP-2 still has a risk of blocking other higher-priority robots. If the "clash" may happen, a certain robot "waiting" becomes necessary. Therefore, in this paper, more MSPs are considered to avoid high-priority robots being blocked by low-priority robots or reduce the possibility of "waiting" happening.

In Fig. 10, suppose $j < n$, which means robot-j (Rj) has higher priority than robot-n (Rn). Suppose the MSP-l of Rj has a clash point with the MSP-1 of Rn, but has not one with the MSP-2 of Rn. Then in Fig. 10, Rn must choose MSP-2 to avoid clashing the MSP-l of Rj. In other words, the lower-priority robot should choose an MSP that does not clash with the high-priority robot as possible as it can. For a lower-priority robot, how to choose the suitable number k such that MSP-k can avoid blocking higher-priority robots will be studied below. Let us consider the matrix in (1), where CM_{Rn} denotes the clash matrix of Rn. Equation (1), as shown at the bottom of the page, where $n \in N$, N is the total number of robots, $j = n - 1$, and q denotes the number of MSPs for any robot. Here the value of q will be discussed later. Moreover, MSP_{Rj} indicates the chosen MSP number of Rj. $CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - q\right)$ denotes whether there is a clash point between MSP_{Rj} and MSP $- q$ of Rn. It is defined as

$$CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - q\right) = \begin{cases} n - j, & \text{clash point exists.} \\ 0, & \text{no clash point} \end{cases} \tag{2}$$

Let VCM_{Rn} denote the vector where each entry is the maximum value of the corresponding row of CM_{Rn}, *i.e.*, equation (3), as shown at the bottom of the page.

Finally, the minimum entry (such as k) of all entries of VCM_{Rn} denotes that Rn should choose MSP-k. Note that R1 is the highest priority robot, then MSP_{R1} is MSP-1. If all entries of VCM_{Rn} are zero, it means there are not any clash points between Rn's MSPs and other higher-priority robots' MSPs. If we find a minimum entry k, which is not zero, it means Rn should choose MSP-k and MSP-k is the most suitable MSP to avoid blocking higher-priority robots. Even MSP-k still cannot avoid blocking a certain higher-priority robot, the blocked robot will be the closest to Rn in priority robot such that the arrival order will be in line with the priority order of all robots as much as possible. In a word, finding MSP for each robot is to shorten its trajectory length, and choosing MSP-k from (3) is to keep the arrival order in line with the priority order of robots as much as possible. Therefore, the value of q may affect the chosen k, in other words, the value q may influence the trade-off between the arrival order and the average trajectories length of all robots. The higher the q value, the more chance to find the suitable k, and the influence of different values q can be seen in the comparison results of Section IV.

$$CM_{Rn} = \begin{bmatrix} CP\left(\mathrm{MSP}_{R1}, \mathrm{MSP} - 1\right) & CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - 1\right) & \cdots & CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - 1\right) \\ CP\left(\mathrm{MSP}_{R1}, MSP - 2\right) & CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - 2\right) & \cdots & CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - 2\right) \\ \vdots & \vdots & \ddots & \vdots \\ CP\left(\mathrm{MSP}_{R1}, \mathrm{MSP} - q\right) & CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - q\right) & \cdots & CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - q\right) \end{bmatrix}, \quad n \geq 2, q > 0, \tag{1}$$

$$VCM_{Rn} = \begin{bmatrix} Max\left\{CP\left(\mathrm{MSP}_{R1}, \mathrm{MSP} - 1\right), CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - 1\right), \cdots, CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - 1\right)\right\} \\ Max\left\{CP\left(\mathrm{MSP}_{R1}, \mathrm{MSP} - 2\right), CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - 2\right), \cdots, CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - 2\right)\right\} \\ \vdots \\ Max\left\{CP\left(\mathrm{MSP}_{R1}, \mathrm{MSP} - q\right), CP\left(\mathrm{MSP}_{R2}, \mathrm{MSP} - q\right), \cdots, CP\left(\mathrm{MSP}_{Rj}, \mathrm{MSP} - q\right)\right\} \end{bmatrix} \tag{3}$$

B. COLLISION PREVENTION [10]

If a lower-priority robot may block a higher-priority robot, the former should choose the suitable MSP-k to avoid blocking. However, the choice may be unavailable if the lower-priority robot does not have enough MSPs to avoid blocking or if all MSPs are the same as SP-1. This paper uses the same collision prevention strategy as in [10]. As shown in Fig. 11, we give a sector in front of each robot and the robot may hit other robots in the sector area as it moves forward, where M_j is the sector's radius and is set as multiples of the diameter of Rj such as $M_j = h \times L_j$, $h \geq 2$, and L_j is the diameter of Rj. Θ is half of the center angle of the sector of a robot and it must be larger than 30 degrees. H_{jn} is the distance between Rj and Rn. G_{jn} is the angle to which Rj rotates to face Rn. If $M_n > H_{jn}$ and $\Theta > G_{nj}$, it means that Rj is inside the sector of Rn. In Fig. 12, MSP_{Rj} and MSP_{Rn} are the final chosen MSPs for Rj and Rn, respectively, where γ_{jn} is the angle between the two MSPs of Rj and Rn, Ψ_{nj} is the vertical distance from the center position of Rn to the MSP_{Rj}. The flow charts in Fig. 13 and Fig. 14 outline robot collision prevention strategies in different situations. If a higher-priority robot (Rj) touches the sector of a lower-priority robot (Rn), Rn chooses its moving decision following the flow chart in Fig. 13. On the other hand, if a lower-priority robot (Rn) touches the sector of a higher-priority robot (Rj), Rj chooses its moving decision following the flow chart in Fig. 14. The above two flowcharts can make collision free between Rj and Rn. A more detailed description can be found in section IV in [10].

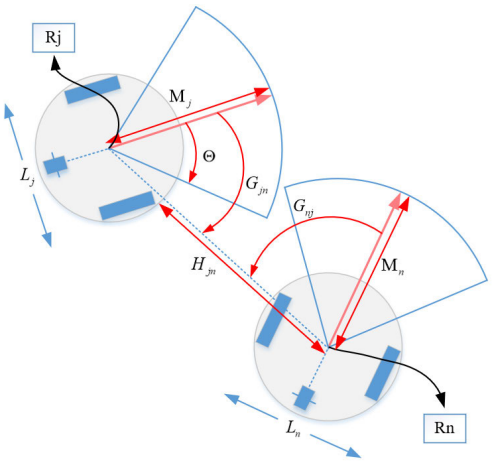

FIGURE 11. The sectors of two robots.

Finally, we summarize the above two subsections' content as a main algorithm as the flowchart shown in Fig. 15.

C. SIMULATION

After we finish the paths planned in the above subsection, let us give a simulation with four robots and eight robots to show the robots' moving performance. Some robot parameters are set in advance as follows. 1. The diameter of a robot is 40 pixels. 2. All robots' speeds are the same. 3. $q = 3$in (3). Let us consider the simulation result for four robots first.

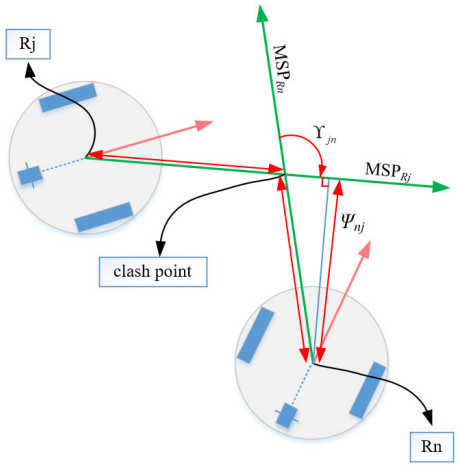

FIGURE 12. The schematic diagram of γ_{jn} and Ψ_{nj}. [10].

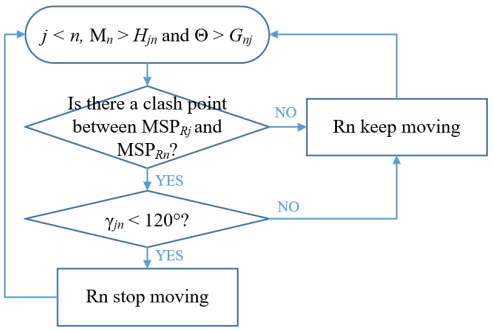

FIGURE 13. The flowchart for the case when the higher-priority robot appears in the sector of the lower-priority robot.

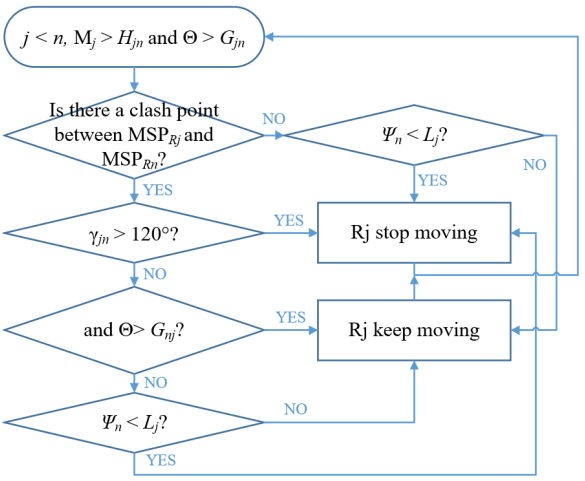

FIGURE 14. The flowchart for the case where the lower-priority robot appears in the sector of the higher-priority robot.

Figure 16 shows the starting positions and orientations of the four robots at the initial frame. In Fig. 17, all robots reach their targets without any collision at frame 269 (frames denote the timing for the robots' moving show). In Fig. 18, R4 has three MSPs and we have $VCM_{R4} = \begin{bmatrix} 3 & 3 & 2 \end{bmatrix}^T$ from (3),

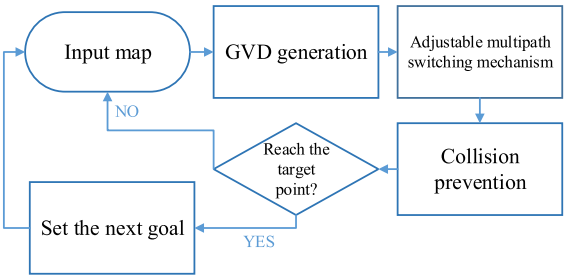

FIGURE 15. The flow chart of PONA2.0.

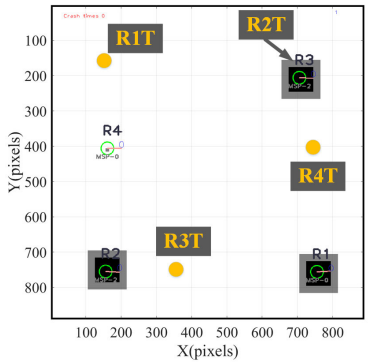

FIGURE 16. At frame 1, the starting positions of four robots.

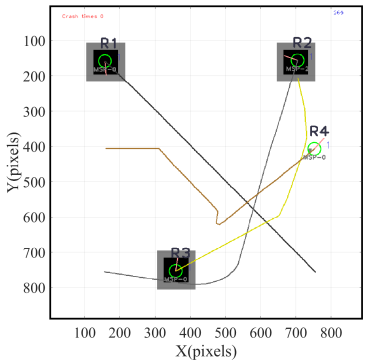

FIGURE 17. At frame 269, all robots arrived at their targets individually.

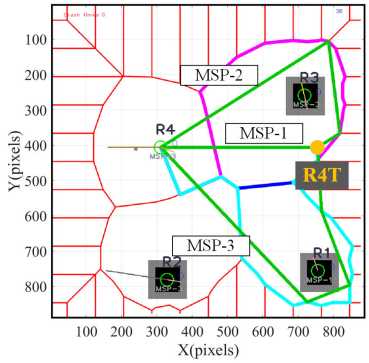

FIGURE 18. At frame 38, R4 chooses MSP-3 to avoid blocking R1.

in which the third entry 2 is minimum. Robot 4 chooses MSP-3 at frame 38. At frame 57 shown in Fig. 19, R4 has

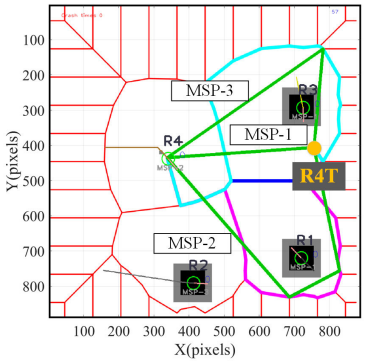

FIGURE 19. At frame 57, R4 chooses MSP-2 to avoid blocking R1.

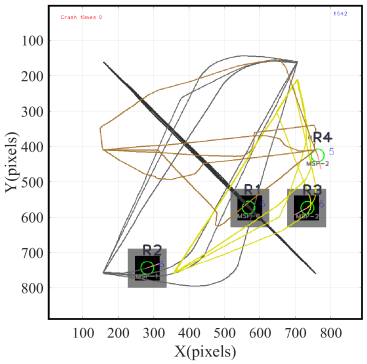

FIGURE 20. All robots achieved more than 2 times round trips.

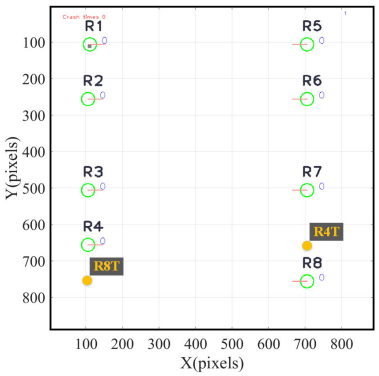

FIGURE 21. The starting positions of eight robots.

$VCM_{R4} = \begin{bmatrix} 3 & 2 & 3 \end{bmatrix}^T$ from(3), so R4 chooses MSP-2. It is noted that the selected MSP-k is calculated in each frame to adjust the path in real time. In Fig. 20, each robot has moved more than 2 round trips. From Fig. 16 to Fig. 20, it is seen that the proposed PONA2.0 achieves collision-free travel for these four robots. A video of this simulation result can be seen at https://youtu.be/6JOGdIRx988 (Accessed on: 21 December 2022).

The following simulation will consider the moving of eight robots. Fig. 21 shows the starting positions and orientations of the eight robots, respectively. After moving, Fig. 22 shows that each robot finally arrives at its target which is opposite its

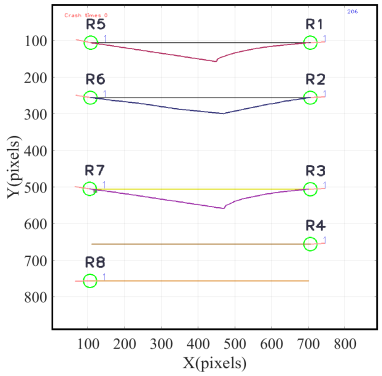

FIGURE 22. Finally, the robots in Fig. 21 arrive at their target positions, respectively.

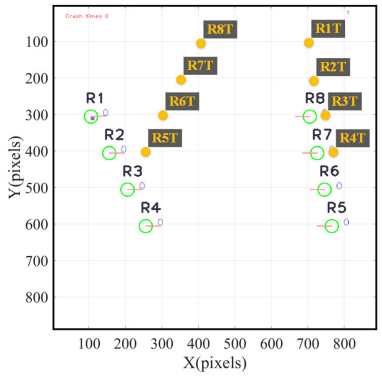

FIGURE 23. Another case was for eight robots with their starting positions and target, respectively.

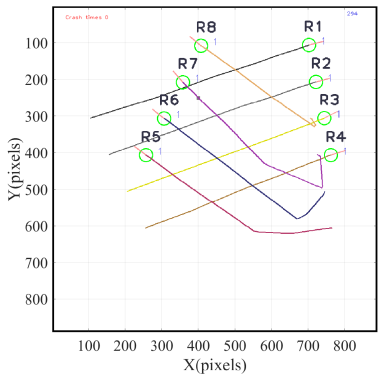

FIGURE 24. Finally, the robots in Fig. 23 arrive at their target positions, respectively.

starting position, such as R1 reaching the starting position of R5 and R2 reaching the starting position R6, etc. Figure 23 shows another case for the starting positions and orientations of the eight robots, respectively. In this case, Fig. 24 shows R1∼R4 finally reaching the targets, respectively, in the upper right and R5∼R8 finally reaching the targets, respectively, in the upper left. These results show that PONA2.0 can achieve collision-free missions under different starting positions and targets.

IV. COMPARISON RESULTS

A. EXPERIMENT ENVIRONMENT

This section will compare the results between the proposed PONA2.0 and other existing algorithms, which are ROA and SDA in [16], NSPP in [9], and PONA in [10], for the four and eight robots' navigation, respectively. The experiment environment is shown in Fig. 25, where all robots are evenly distributed on a circle with a diameter of 520 pixels. The starting positions of the robots are initialized on a circle from P1 to P8 with arbitrary order, and the targets are on the opposite side of the corresponding starting positions, respectively. Each robot is at its starting position and must move toward its target. In the four robots' experiment, the robots are evenly distributed on the positions P1, P3, P5, and P7. Suppose R2, R3, R1, and R4 are at the starting positions P1, P3, P5, and P7, respectively, and their targets are at P5, P7, P1, and P3, respectively. There are six configurations for four robots and 5040 ($7 \times 6 \times 5 \times 4 \times 3 \times 2$) configurations for eight robots. The eight robots are placed from P1 to P8 initially on a circle as in Fig. 25, and it is one configuration. However, those robots can change their initial positions each other randomly. Since their positions are on a circle, so there are $7 \times 6 \times 5 \times 4 \times 3 \times 2 = 5040$ possible initial placements totally, that is, there are 5040 configurations.

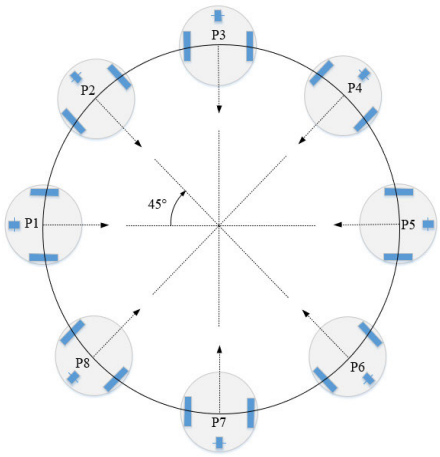

FIGURE 25. The experiment environment for the comparison. [9].

B. COMPARISON RESULTS IN ARRIVAL ORDER

We selected all configurations of four robots and six configurations of eight robots to compare their performance with other existing algorithms. The performance includes arrival order and average trajectory length. The comparison of the arrival order of robots between NSPP, PONA, and PONA2.0 is shown in Table 1 and Table 2, respectively. Let PONA2.0(2314) denote that four robots are arranged in order R2, R3, R1, and R4, initially, and the PONA2.0 navigation algorithm is used. In Table 1, R2 in the "1st" column represents R2 arriving at its target first, and R1 in the "4th" column represents R1 arriving at its target last. Both PONA and PONA2.0 performed better than NSPP in all configu-

TABLE 1. The arrival order for four robots.

Four robots in circular test					
Arrival order		1st	2nd	3rd	4th
NSPP(1234)		R2	R3	R4	R1
NSPP(1243)		R2	R4	R3	R1
NSPP(1324)		R3	R4	R1	R2
NSPP(1342)		R4	R2	R3	R1
NSPP(1423)		R3	R4	R1	R2
NSPP(1432)		R3	R2	R4	R1
PONA(1234)		R1	R2	R3	R4
PONA(1243)		R1	R2	R4	R3
PONA(1324)		R1	R2	R3	R4
PONA(1342)		R1	R2	R3	R4
PONA(1423)		R1	R2	R3	R4
PONA(1432)		R1	R2	R3	R4
$q = 2$	PONA2.0(1234)	R1	R2	R3	R4
	PONA2.0(1243)	R1	R2	R3	R4
	PONA2.0(1324)	R1	R2	R3	R4
	PONA2.0(1342)	R1	R2	R3	R4
	PONA2.0(1423)	R1	R2	R3	R4
	PONA2.0(1432)	R1	R2	R3	R4

rations, which means the robot arrival order of PONA and PONA2.0 is more in line with the robot priority than that of NSPP. Furthermore, it is seen from Table 1 that R4 reaches the target earlier than R3 in PONA(1243), but the arrival order is in line with their priority orders in PONA2.0(1243) when $q = 2$. It is seen from Table 1 that all robots arrive at their targets in line with the robot's priority orders no matter what initial order arrangement for four robots. We found that there is the same result for $q = 2$.

In the eight robots circular test, the path clash between multiple robots will be more frequent than in the four robots circular test. We choose q value from $q = 2$ to $q = 6$ in the test. In Table 2, both PONA and PONA2.0 also performed better than NSPP in six configurations, R1 and R2 can usually be the first or second to reach the target when $q = 3$ in PONA2.0. When $q = 4$ in PONA2.0, the order of robot arrival is mostly in line with the robot priority except in PONA2.0(15348726). When $q = 5$ and $q = 6$, lower-priority robots rarely reach the target before higher-priority robots. However, PONA2.0 performs inferiorly to PONA when $q = 2$. In more robots, PONA2.0 is not guaranteed to perform well in the case of insufficient MSPs. Here we define the performance index AO of arrival order.

$$AO = \sum_{k=1}^{n} |k - \tau_k|, \tag{4}$$

where τ_k denotes the robot number which is the k^{th} robot to reach the target. The larger AO means the more the number of lower-priority robots arriving at the target earlier than higher-priority robots. When $AO = 0$, it means all robots arrive in the correct order (in line with robot priority). In Table 3, PONA and PONA2.0 perform well in the four robot circular tests. PONA2.0 with $q = 5$ and $q = 6$ can always provide lower AO than PONA in the eight robots' circular test, and it gives higher-priority robots to reach the target earlier than PONA.

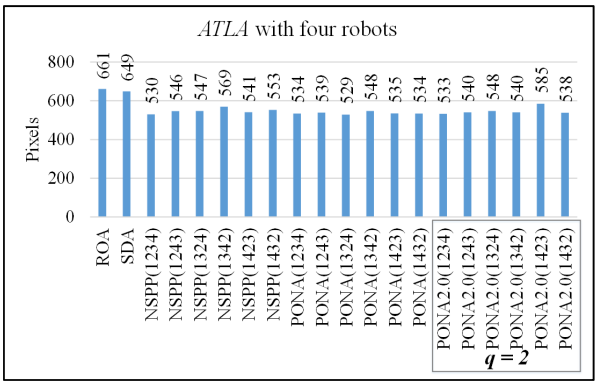

FIGURE 26. *ATLA* comparison with different algorithms for four robots.

The AO in PONA2.0 with $q = 3$ and $q = 4$ are 30 and 22, respectively, both better than PONA with AO= 50. We do not recommend $q = 2$ in eight robots test. In general, the larger q will give a smaller AO in PONA2.0.

C. COMPARISON RESULTS IN AVERAGE LENGTH

Next, let us compare the other performance index, which is the average length of moving trajectories of all robots. Define the average trajectories length of all robots (*ATLA*) below.

$$ATLA = \frac{STLA}{N}, \tag{5}$$

and STLA denotes the sum of the trajectory lengths of all robots. The comparison of *ATLA* is shown in Fig. 26 and Fig. 27. In Fig. 26, NSPP, PONA, and PONA2.0 almost give shorter *ATLA* than ROA and SDA. Compared to NSPP and PONA, PONA2.0 can improve the arrival order while maintaining a similar *ATLA* except for PONA2.0(1423). Compared to PONA2.0 with $q = 2$, PONA has a stricter switching mechanism that sometimes results in shorter *ATLA*. The shorter *ATLA* will be called the better performance in *ATLA*. But PONA2.0(1423) can use a higher q to get better performance in *ATLA* than PONA in the four robots test. The *ATLA* of PONA2.0(1423) with $q = 3$ is 576 pixels and with $q = 4$ is 544 pixels, which are close to the *ATLA* of PONA(1423), and all of them have $AO = 0$. In Fig. 27, PONA2.0 with $q = 5$ can give a shorter *ATLA* than ROA and SDA and a similar *ATLA* as NSPP and PONA. However, we can see in some configurations that the *ATLA* of PONA2.0 is larger than other algorithms because lower-priority robots sometimes took farther MSP to avoid blocking higher-priority robots. Especially in the configurations of PONA2.0(15462738) with $q = 3$ and PONA2.0(18526743) with $q = 6$, the *ATLA* is even larger than SDA. But in PONA2.0, the average of all *ATLA* is 594 with $q = 2$; is 620 with $q = 3$; is 598 with $q = 4$; is 607 with $q = 5$; and is 619 with $q = 6$. All of them are shorter than the *ATLA* of ROA and SDA. The average *ATLA* of PONA 2.0 is larger than the average *ATLA* = 575 of NSPP and 586 of PONA. A low-priority robot always chooses farther

TABLE 2. The order of the eight robots' arrival.

Eight robots in circular test																			
Arrival order		1st	2nd	3rd	4th	5th	6th	7th	8th	Arrival order		1st	2nd	3rd	4th	5th	6th	7th	8th
NSPP(15462738)		R6	R7	R8	R3	R5	R4	R1	R2	$q=4$	PONA2.0(15462738)	R1	R2	R3	R4	R5	R6	R7	R8
NSPP(12345678)		R3	R7	R4	R1	R5	R6	R8	R2		PONA2.0(12345678)	R1	R2	R3	R4	R5	R6	R8	R7
NSPP(18526743)		R4	R7	R1	R8	R5	R6	R2	R3		PONA2.0(18526743)	R1	R2	R3	R4	R6	R5	R7	R8
NSPP(14632875)		R7	R8	R3	R5	R6	R4	R1	R2		PONA2.0(14632875)	R1	R2	R3	R4	R5	R6	R8	R7
NSPP(18247635)		R7	R8	R4	R6	R5	R2	R1	R3		PONA2.0(18247635)	R1	R2	R3	R4	R5	R7	R6	R8
NSPP(15348726)		R6	R4	R8	R7	R5	R2	R3	R1		PONA2.0(15348726)	R1	R2	R7	R3	R8	R4	R5	R6
PONA(15462738)		R1	R2	R5	R4	R3	R6	R7	R8	$q=5$	PONA2.0(15462738)	R1	R2	R3	R4	R5	R6	R7	R8
PONA(12345678)		R1	R4	R2	R7	R5	R3	R8	R6		PONA2.0(12345678)	R1	R2	R3	R4	R7	R5	R6	R8
PONA(18526743)		R2	R3	R1	R4	R6	R5	R7	R8		PONA2.0(18526743)	R1	R2	R3	R4	R5	R6	R7	R8
PONA(14632875)		R1	R7	R2	R3	R5	R6	R4	R8		PONA2.0(14632875)	R1	R2	R3	R5	R4	R7	R6	R8
PONA(18247635)		R1	R2	R3	R8	R4	R6	R5	R7		PONA2.0(18247635)	R1	R2	R3	R4	R5	R6	R7	R8
PONA(15348726)		R1	R2	R3	R8	R4	R7	R5	R6		PONA2.0(15348726)	R1	R2	R3	R4	R6	R7	R5	R8
$q=2$	PONA2.0(15462738)	R1	R2	R5	R3	R6	R7	R4	R8	$q=6$	PONA2.0(15462738)	R1	R2	R3	R4	R5	R6	R8	R7
	PONA2.0(12345678)	R1	R2	R3	R6	R5	R4	R7	R8		PONA2.0(12345678)	R1	R2	R3	R4	R5	R6	R7	R8
	PONA2.0(18526743)	R1	R2	R6	R3	R8	R4	R5	R7		PONA2.0(18526743)	R1	R2	R3	R4	R5	R6	R7	R8
	PONA2.0(14632875)	R1	R4	R5	R2	R6	R3	R7	R8		PONA2.0(14632875)	R1	R2	R3	R4	R5	R6	R7	R8
	PONA2.0(18247635)	R1	R2	R3	R6	R8	R4	R5	R7		PONA2.0(18247635)	R1	R2	R3	R4	R5	R6	R7	R8
	PONA2.0(15348726)	R1	R5	R3	R6	R8	R4	R2	R7		PONA2.0(15348726)	R1	R2	R3	R4	R5	R8	R6	R7
$q=3$	PONA2.0(15462738)	R1	R2	R4	R3	R6	R7	R5	R8										
	PONA2.0(12345678)	R1	R2	R7	R3	R4	R5	R6	R8										
	PONA2.0(18526743)	R1	R2	R3	R4	R6	R5	R8	R7										
	PONA2.0(14632875)	R1	R2	R4	R3	R5	R7	R8	R6										
	PONA2.0(18247635)	R2	R1	R3	R4	R5	R6	R7	R8										
	PONA2.0(15348726)	R1	R2	R3	R6	R4	R5	R7	R8										

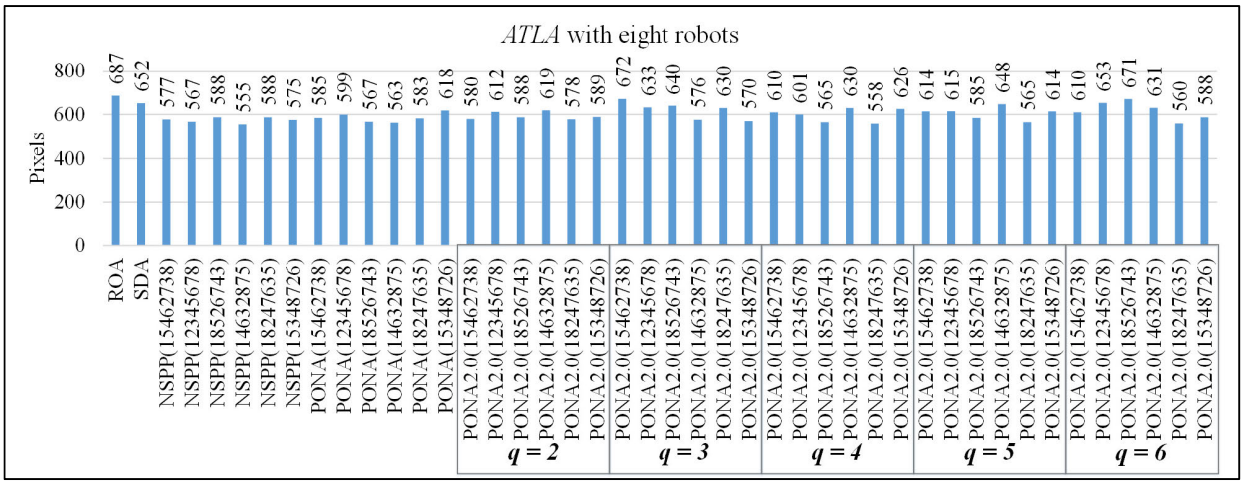

FIGURE 27. *ATLA* comparison with different algorithms for eight robots.

MSP, otherwise, many high-priority robots may clash with it, especially in the circular test. Unfortunately, it may have some cases where a low-priority robot has an exceptionally long trajectory such that its *ATLA* becomes very large, which is seen from PONA2.0(18526743) with $q = 6$ in Fig. 28, where R8 in Fig. 28 is that robot with exceptionally long trajectory. Therefore, the larger q in PONA2.0 can make *AO* smaller, and most PONA2.0 with adjustable value q may have a shorter *ATLA* than ROA and SDA. But we have to admit that we still cannot guarantee that the *ATLA* of any PONA 2.0 in the same robots' configuration must be smaller than other algorithms.

In our simulation experience, we have learned some knowledge as follows. In an experiment environment, there are many robots in the space with a certain configuration, such as Fig. 25. Let each robot's starting position and target position be connected by a straight line. If the connection lines of all robots have most intersections (such as ω intersections) in a very small area. If the intersection number ω is less than five, then $q = 2$ can get very low *AO*, and $q = 3$ not only can have low *AO*, but also reduce its *ATLA* further. Then, if the intersection number ω increases one more, we increase the value q one more to get a low *PI* and small *ATLA*. However, when the number ω increases up to 8, the increase of q may

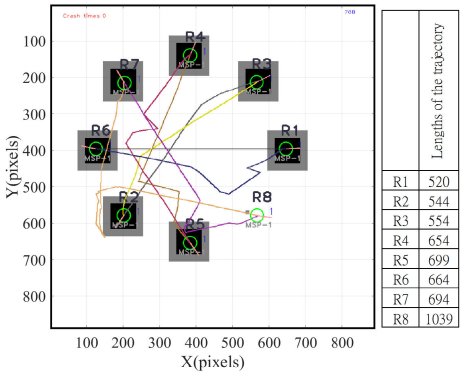

FIGURE 28. All robot's trajectories and lengths in PONA2.0(18526743) with $q = 6$.

TABLE 3. The performance index of arrival order.

Four robots in circular test			Eight robots in circular test		
Score (lower is better)		*AO*	Score (lower is better)		*AO*
NSPP(1234)		6	NSPP(15462738)		30
NSPP(1243)		8	NSPP(12345678)		18
NSPP(1324)		6	NSPP(18526743)		24
NSPP(1342)		6	NSPP(14632875)		28
NSPP(1423)		6	NSPP(18247635)		30
NSPP(1432)		6	NSPP(15348726)		30
PONA(1234)		0	PONA(15462738)		4
PONA(1243)		2	PONA(12345678)		12
PONA(1324)		0	PONA(18526743)		6
PONA(1342)		0	PONA(14632875)		10
PONA(1423)		0	PONA(18247635)		8
PONA(1432)		0	PONA(15348726)		10
$q = 2$	PONA2.0(1234)	0	$q = 2$	PONA2.0(15462738)	8
	PONA2.0(1243)	0		PONA2.0(12345678)	4
	PONA2.0(1324)	0		PONA2.0(18526743)	12
	PONA2.0(1342)	0		PONA2.0(14632875)	10
	PONA2.0(1423)	0		PONA2.0(18247635)	10
	PONA2.0(1432)	0		PONA2.0(15348726)	16
			$q = 3$	PONA2.0(15462738)	6
				PONA2.0(12345678)	8
				PONA2.0(18526743)	4
				PONA2.0(14632875)	6
				PONA2.0(18247635)	2
				PONA2.0(15348726)	4
			$q = 4$	PONA2.0(15462738)	0
				PONA2.0(12345678)	2
				PONA2.0(18526743)	2
				PONA2.0(14632875)	2
				PONA2.0(18247635)	2
				PONA2.0(15348726)	14
			$q = 5$	PONA2.0(15462738)	0
				PONA2.0(12345678)	4
				PONA2.0(18526743)	0
				PONA2.0(14632875)	4
				PONA2.0(18247635)	0
				PONA2.0(15348726)	4
			$q = 6$	PONA2.0(15462738)	2
				PONA2.0(12345678)	0
				PONA2.0(18526743)	0
				PONA2.0(14632875)	0
				PONA2.0(18247635)	0
				PONA2.0(15348726)	4

slightly increase the *ATLA*. In conclusion, if we only concern with lower *AO* (better arrival order), a larger q is useful.

TABLE 4. The average *ATLA* difference between pona2.0 and other methods.

compare with \ number of robots	4 ($q = 2$)	8 ($q = 4$)	8 ($q = 5$)	8 ($q = 6$)
ROA	-17%	-13%	-11%	-10%
SDA	-15%	-8%	-7%	-5%
NSPP	0%	+3.3%	+5%	+7%
PONA	+2%	+2.2%	+3.5%	+5.6%

TABLE 5. The average *AO* difference between pona2.0 and other methods.

compare with \ number of robots	4 ($q = 2$)	8 ($q = 4$)	8 ($q = 5$)	8 ($q = 6$)
NSPP	-100%	-86%	-92%	-96%
PONA	-100%	-56%	-76%	-88%

However, if we care about both *AO* and *ATLA*, q cannot be too large.

Table 4 compares PONA2.0 with ROA, SDA, NSPP, and PONA on the differences in average *ATLA*. If the difference is negative, the average *ATLA* of PONA 2.0 is less than the others. Otherwise, it is larger than the others. The so-called average *ATLA* is the average of the six different configurations of *ATLA* we have chosen in subsection B of Section IV. For example, the NSPP's average *ATLA* is the average of *ATLA* values of six configurations which include the *ATLAs* of NSPP(1234), NSPP(1243), NSPP(1324), NSPP(1342), NSPP(1424), and NSPP(1432). Also, the average of *AO* is the average of the six configurations of *AO*. Compared with ROA and SDA, PONA2.0 has the shorter average *ATLA* in four and eight robots' configurations. Also, compared with NSPP and PONA, PONA2.0 has a little larger average *ATLA* than those of NSPP and PONA, but PONA2.0 improves the waiting problem of the higher priority robot significantly (as shown in Table 5).

V. CONCLUSION AND FUTURE WORK

This study has proposed an evolutionary multi-robot navigation algorithm (PONA2.0) where all robots are in priority order. It is known that PONA [10] considered two MSPs, *i.e.*, MSP-1 and MSP-2, to shorten the trajectory for arriving at the targets. However, it still has the possibility that one of the robots must wait for a while to avoid colliding with another robot. Therefore, the proposed PONA2.0 with choosing a suitable MSP could solve this problem. This paper has proposed an adjustable multipath switching mechanism to find a suitable MSP such that the multiple robots can arrive at their targets being in line with the robot's priority order. We defined *ATLA* and *AO* to be the performance indices of average trajectories length and arrival order, respectively. Based on the experiment on the four and eight robots' configurations, PONA2.0 actually has a shorter average *ATLA* than ROA and SDA in [16], and a lower average *AO* than NSPP in [9] and PONA in [10]. However, we have to admit that how to find a suitable MSP more systematically and how to guarantee the performance of PONA2.0 in more robots (more than 8) are still open problems to be worthen studying in the future.

REFERENCES

[1] H.-X. Wei, Q. Mao, Y. Guan, and Y.-D. Li, "A centroidal Voronoi tessellation based intelligent control algorithm for the self-assembly path planning of swarm robots," *Expert Syst. Appl.*, vol. 85, pp. 261–269, Nov. 2017.
[2] C. Huang, X. Chen, Y. Zhang, S. Qin, Y. Zeng, and X. Li, "Switched linear multi-robot navigation using hierarchical model predictive control," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 4331–4337.
[3] P. K. Das, H. S. Behera, P. K. Jena, and B. K. Panigrahi, "Multi-robot path planning in a dynamic environment using improved gravitational search algorithm," *J. Electr. Syst. Inf. Technol.*, vol. 3, no. 2, pp. 295–313, Sep. 2016.
[4] K. Jose and D. K. Pratihar, "Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods," *Robot. Auton. Syst.*, vol. 80, pp. 34–42, Jun. 2016.
[5] 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.
[6] J.-H. Liang and C.-H. Lee, "Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm," *Adv. Eng. Softw.*, vol. 79, pp. 47–56, Jan. 2015.
[7] M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing schedules for prioritized path planning of multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2001, pp. 271–276.
[8] R. Regele and P. Levi, "Cooperative multi-robot path planning by heuristic priority adjustment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 5954–5959.
[9] S.-K. Huang, W.-J. Wang, and C.-H. Sun, "A path planning strategy for multi-robot moving with path-priority order based on a generalized Voronoi diagram," *Appl. Sci.*, vol. 11, no. 20, p. 9650, Oct. 2021.
[10] S.-K. Huang, W.-J. Wang, and C.-H. Sun, "A new multirobot path planning with priority order based on the generalized Voronoi diagram," *IEEE Access*, vol. 10, pp. 56564–56577, 2022.
[11] W. Yu, J. Peng, and X. Zhang, "A prioritized path planning algorithm for MMRS," in *Proc. 33rd Chin. Control Conf.*, Jul. 2014, pp. 966–971.
[12] M. Cap, P. Novak, A. Kleiner, and M. Selecky, "Prioritized planning algorithms for trajectory coordination of multiple mobile robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 835–849, Jul. 2015.
[13] C. Riman and P. E. Abi-Char, "A priority-based modified A* path planning algorithm for multi-mobile robot navigation," in *Proc. 19th Int. Conf. Electr. Eng., Comput. Sci. Autom. Control (CCE)*, Nov. 2022, pp. 1–6.
[14] A. Andreychuk and K. Yakovlev, "Two techniques that enhance the performance of multi-robot prioritized path planning," in *Proc. Int. Joint Conf. Auto. Agents Multi-Agent Syst.*, 2018, pp. 2177–2179.
[15] D. T. Lee, "Generalized Voronoi diagrams in the plain," *SIAM J. Comput.*, vol. 10, no. 1, pp. 73–87, 1981.
[16] A. A. Ali, A. T. Rashid, M. Frasca, and L. Fortuna, "An algorithm for multi-robot collision-free navigation based on shortest distance," *Robot. Auto. Syst.*, vol. 75, pp. 119–128, Jan. 2016.
[17] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quart. Appl. Math.*, vol. 27, no. 4, pp. 526–530, 1970.

SHENG-KAI HUANG is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, National Central University. He has been a Researcher with the Intelligent Control and Image Processing Laboratory, National Central University, since 2016. His research interests include multi-robot path planning, visual odometry, and image processing.

WEN-JUNE WANG (Life Fellow, IEEE) received the B.S. degree in control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1980, the M.S. degree in electrical engineering from Tatung University, Taipei, Taiwan, in 1984, and the Ph.D. degree in electronics from National Chiao Tung University, in 1987. He has authored or coauthored more than 160 refereed journal articles and 160 conference papers in the areas of fuzzy systems and theorems, robust and nonlinear control in large-scale systems, and neural networks. His most significant contributions are the design of fuzzy systems and the development of robotics. His current research interests include robot control, neural networks, and pattern recognition. He was an IFSA Fellow, in 2017.