IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 4, 2021, accepted March 26, 2021, date of publication March 31, 2021, date of current version April 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3070054

Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment

GANG TANG[1], CONGQIANG TANG[1], CHRISTOPHE CLARAMUNT[1,2], XIONG HU[1], AND PEIPEI ZHOU[3]

[1]School of Logistics Engineering, Shanghai Maritime University, Shanghai 201306, China
[2]Naval Academy Research Institute, 29240 Brest, France
[3]School of Mechatronic Engineering, Guangdong Polytechnic Normal University, Guangzhou 510665, China

Corresponding author: Peipei Zhou (zhoupeipei@gpnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 51905557.

ABSTRACT This research introduces a path planning method based on the geometric A-star algorithm. The whole approach is applied to an Automated Guided Vehicle (AGV) in order to avoid the problems of many nodes, long-distance and large turning angle, and these problems usually exist in the sawtooth and cross paths produced by the traditional A-star algorithm. First, a grid method models a port environment. Second, the nodes in the close-list are filtered by the functions $P(x, y)$ and $W(x, y)$ and the nodes that do not meet the requirements are removed to avoid the generation of irregular paths. Simultaneously, to enhance the stability of the AGV regarding turning paths, the polyline at the turning path is replaced by a cubic B-spline curve. The path planning experimental results applied to different scenarios and different specifications showed that compared with other seven different algorithms, the geometric A-star algorithm reduces the number of nodes by 10% $\sim$ 40%, while the number of turns is reduced by 25%, the turning angle is reduced by 33.3%, and the total distance is reduced by 25.5%. Overall, the simulation results of the path planning confirmed the effectiveness of the geometric A-star algorithm.

INDEX TERMS A-star algorithm, automated guided vehicle (AGV), grid method, path planning.

I. INTRODUCTION

Recent years, with the rapid development of warehouse logistics automation and the increase of labor cost, the transportation of goods in many areas is developing towards systematization and automatisation. Automated Guided Vehicle (AGV) are nowadays widely used in warehouses [1], ports [2] and factories [3]. In port environments, AGVs provide a kind of automatic handling machinery applied in automatic and semi-automatic container terminals. It can complete braking, turning, road selection, and other operations under unmanned control [4]. With the improvement of automatic port handling technology, working environments and tasks an AGV needs to adapt to are also changing. AGVs generally perform handling and transportation of goods but often need to quickly find a charging station to complete its charging tasks [5]. The above requirements mean that an AGV needs to plan a collision-free, safe, and feasible path to avoid obstacles and accurately reach the charging station in different port environments [6].

A global path planning of an AGV in a port should first obtain the static obstacle layout of the port. Then a planning search algorithm is used to find a suitable path from the start point to the goal point. The focus of a path planning is to build a reasonable obstacle distribution map based on the known environment [7], and then find a path with the least cost to avoid obstacles [8].

The preliminary layout tasks is to map the background environment towards a geographical representation that integrates the features and shapes. Various algorithms have been so far proposed to model the environment such as Voronoi diagrams [9], visibility graphs [10] and grid structures [11].

A Voronoi diagram consists of a series of continuous polygons made of vertical bisectors that connect two adjacent straight lines. Candeloro *et al.* [9] divided a scene containing obstacles based on a Voronoi diagram and then generate a path to avoid obstacles through a Fermat spiral.

The associate editor coordinating the review of this manuscript and approving it for publication was Laurence T. Yang.

The advantage of a Voronoi diagram is that the generated path is relatively safe. However, most algorithms require a large storage space and cannot guarantee an optimal path length.

Visibility graphs derive a geometric information that take into account the obstacles and replace them by polygon approximations. Majeed and Lee [12] used a space constraint and sparse visibility graph in the path finding process. The advantage of the a visibility graph is that it is intuitive and the shortest path can be easily determined. But once the positions of the start point and the goal point are changed, the whole map must be rebuilt, which offers poor flexibility.

Compared with the above two methods, the grid method uses conventional shapes to represent the environment. Each grid is constant; the data structure is simple; easy to manage, store and retrieve. Therefore, it has been often used to represent the environments. The grid method was first proposed by Hart *et al.* [13]. For instance, Xie *et al.* [14] transformed an offshore wind farm into a grid map. Fransen *et al.* [11] derived a parcel sorting layout, the bag handling layouts at an airport, and a semiconductor fabrication layout bay through the grid method. Tsatcha *et al.* [15] used a series of hexagonal grids to simulate sea lanes. Liu *et al.* [16] built a dynamic grid simulation environment based on the electronic chart and AIS system.

The path search algorithm is a method based on the map and searching for the path between the start point and goal point. According to different strategies, path search algorithms can be divided into bionic algorithms, geometric model search algorithms, and interpolation-based algorithms.

A bionic algorithm is a kind of random search algorithm that simulates natural biological evolution or group social behavior. A series of algorithms, such as particle swarm optimization (PSO) algorithm [17], genetic algorithm (GA) [18], and ant colony (AC) algorithm [19] have been applied to path search processes. Although they are generally easy to implement, they are generally computationally expensive and often fall into some local optimum, so they are not always appropriate for path planning in complex grid map environments.

Interpolation-based algorithms generally apply smooth curves, such as B-spline curve [20] and dubins curves [21] to fit the entire path. The generated paths are smooth and with continuous curvature, which are valuable properties for speed and acceleration requirements. But they do not apply very well to irregular paths, their application is also often reserved to smooth local or simple path problems.

Geometric search algorithms are commonly used path search algorithm, mainly including the Dijkstra algorithm [22] and the A-star algorithm [23]. The Dijkstra algorithm obtains the shortest path by traversing all nodes. The success rate of obtaining the path is high, but the computational efficiency is low. The A-star algorithm is a search algorithm based on the Dijkstra algorithm by adding a heuristic function. Compared with Dijkstra algorithm, the A-star algorithm is simpler and more direct in solving the shortest path problem, but this algorithm is also prone to fall into the local optimum.

The A-star algorithm was first proposed by Hart [24], which is a heuristic search algorithm [25]. The A-star algorithm not only runs fast in the path search process, but also has good real-time performance so it is widely used in path planning. However, when the A-star algorithm is used for path planning in complex environments, a cross path and sawtooth path will be generated [26]. If the map size is too large or the environment is too complicated, the A-star algorithm will seriously affect the efficiency of the AGV [27].

Thus, there is still a need to improve the A-star algorithm to avoid the above problems. Currently, two main methods have been applied to improve the A-star algorithm. The first one is based on a pre-processing step, while a second one applies post-processing. The pre-processing method is to optimize the path during the algorithm search. Fu *et al.* [28] proposed an improved A-star algorithm to shorten the path by judging whether there is an obstacle between the current way point and the target point. However, this method is computationally expensive and does not smooth the turning path. The post-processing method is to optimize the path after the algorithm search is over. Song *et al.* [29] used the three kinds of smoothers to reduce the number of turning points in order to delete redundant path nodes. However, this method is prone to the number of nodes and requires multiple iterations. Saeed *et al.* [30] introduced a path enhancement method to shorten the path length by judging whether the line between the adjacent path nodes passes through obstacles. But still, this generates larger turning angles. Han and Seo [31] applied a SPS algorithm to generate point sets around obstacles, and then the positions of nodes were continuously updated by a PSO algorithm. Huang *et al.* [32] proposed a k-degree smoothing method to generates the path satisfying the turning constraint of the AGV by changing the smoothness K. However, generating the initial path by the AC algorithm is still computationally expensive.

A few improved algorithms are based on the A-star algorithm such as the Bidirectional A-star algorithm [33] where the algorithm simultanesously searches for the start point and goal point. Last, the Breadth-First-Search (BFS) [34] and the Depth-First-Search (DFS) [35] algorithms are also used widely in the grid maps. The BFS algorithm is a search algorithm implemented using queues, and the DFS algorithm is a search algorithm implemented using recursion.

In addition, the paths generated by common algorithms are all polyline paths, which can't meet the requirements of continuous speed and acceleration when taking into account the turning process of the AGV. Therefore, an appropriate method to smooth the path is still required [36].

In order to plan a collision-free, smooth path with the least cost, this paper combines the advantages of A-star algorithm and interpolation algorithm, optimizes it through geometric rules, and proposes a geometric A-star algorithm. In the first stage, the proposed approach generates a grid map of the port environment so that the A-star algorithm can quickly search

for a collision-free path. In the second stage, this paper set the filter functions $P(x, y)$ and $W(x, y)$ to eliminate invalid nodes that cause irregular paths. Then in the third stage, the proposed opproach achieves the smoothness of the entire path through cubic B-spline interpolation. The contributions of the paper can be summarized as follows:

1) By setting the filter function, the path obtained by the A-star algorithm is processed to avoid the problem of excessive large turning angles and irregular paths.
2) Combined with the cubic B-spline interpolation function, the regularized path is smoothed to ensure the continuous speed and acceleration of the AGV during motion.

The rest of this paper is organized as follows. The second section introduces the modeling method of the port environment, the practical significance of the port AGV's path planning, and the objective function of the AGV'path planning. The third section explains the principle and the whole process of the geometric A-star algorithm. The fourth section verifies the effect of the geometric A-star algorithm on path planning in grid maps of different scales and different environments, then compares the path planning effects of seven different algorithms. The last section summarizes the optimization effect of the geometric A-star algorithm and the shortcomings of current research and a few directions for future research.

II. ENVIRONMENT MODELING AND OPTIMZATION GOAL

To accurately reflect the movement of the AGV in the port environment, a satellite map of port is used to establish an environment model. First, a port-related satellite map from Google Earth is extracted, and the satellite picture of the port is transformed into a grid map that can be easily computationally processed. The grid method provides a simplified view of the complex port environment which is acceptable, and improves the search efficiency during the path planning process.

A. RASTERIZATION OF THE PORT ENVIRONMENT

As shown in Fig. 1a, the satellite map 1 of the Yangshan Port is obtained from the Google Earth Maps provides the background scenario for studying the AGV path planning. The satellite map of port is made of some complex shapes formed by containers so the path search algorithm cannot be directly applied. Therefore, irregular graphics should be taken into account by the rasterization process and the satellite map is converted into an grid map with simplified geometric features. According to the information contained in the satellite map, the location information and size of containers can be obtained. Before rasterizing the satellite map, it is necessary to manually divide the boundary contour between the storage yard area and the access area for the AGV in the satellite map. The contour area is marked as black, and the area outside the contour is marked as white (Fig. 1b).

In the grid map, the contour area that the AGV cannot pass is marked as the obstacle grid, which is represented by black; the area where the AGV can pass is marked as the free grid, which is represented by white. The grid size is smaller, the closer it is to reality, but the AGV in actual planning is likely to encounter a few obstacles. After considering the size of the actual scene and the real AGV size, the grid size of satellite map 1 is determined to be 50×50 as a good balance between performance and appropriateness. Fig. 1b is the grid map obtained by rasterizing satellite map 1.

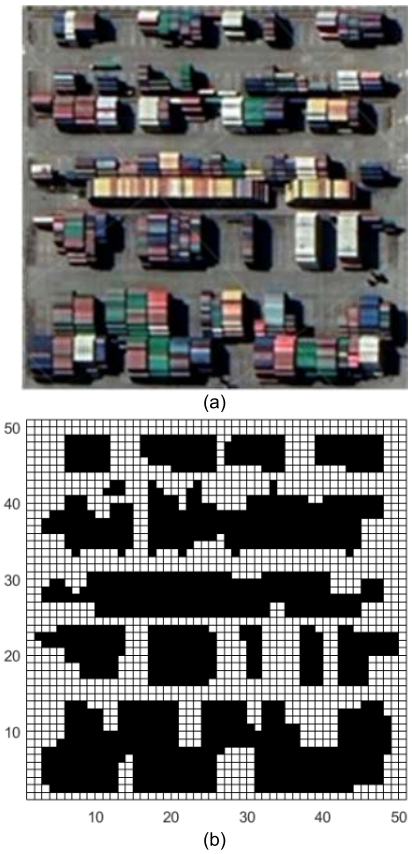

FIGURE 1. Simplified three-dimensional simulation model of a port: (a) Satellite map 1 of Yangshan Port; (b) 50 × 50 map 1.

B. GRID REPRESENTATION

After the satellite map of port is converted into a grid map, the path planning problem of the AGV is simplified to a path optimization problem problem from the start point to the goal point. In order to record the information of the grid, the serial number method is used to encode the grid. The specific steps are to establish a planar rectangular coordinate XOY with the bottom left of the map as the origin, and to encode the grid from bottom to top and from left to right. Any grid N_i corresponds to the grid center of the rectangular coordinate system with a coordinate value of (x_i, y_i). The conversion relationship between the grid coordinates and the grid number N_i is as follows:

$$\begin{cases} N_i = n[m - (x_i + 9.5)] + (y_i + 0.5) \\ x_i = \lceil N_i/n \rceil - 0.5 \\ y_i = \lfloor N_i/n \rfloor - 0.5 \end{cases}, \quad (1)$$

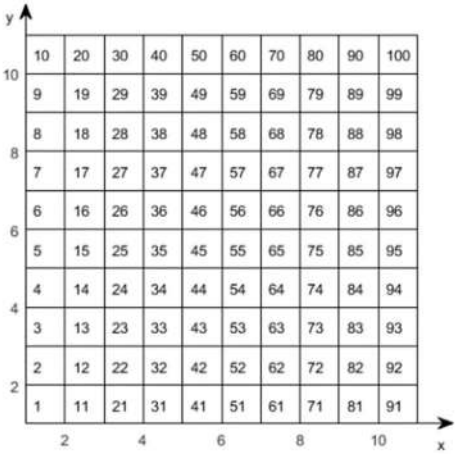

FIGURE 2. Grid sequence diagram.

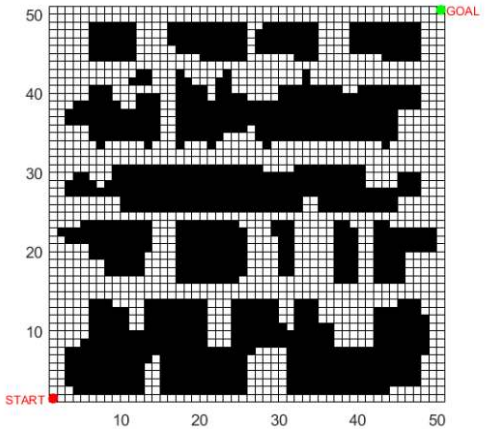

FIGURE 3. Schematic diagram of the start point and goal point.

where m and n respectively represent the number of rows and columns of the grid number N_i.

The grid number diagram is shown in Fig. 2, the grid number increases from bottom to top and from left to right.

The process of establishing an environmental model can be described as follows:

Step 1: Obtain port environmental information from satellite maps;

Step 2: Remove unnecessary obstacles, regularizing irregular shapes;

Step3: Mark the area that the AGV can pass through as a white grid, and mark the obstacle area that cannot pass as a black grid;

Step 4: Convert the entire port environment map into a grid map of equal size;

Step 5: Encode all the grids in the map.

C. OBJECTION FUNCTION OF THE PATH PLANNING

There is an additional important constraint as the AGV needs to automatically navigate to the nearest charging station to charge itself when the battery is insufficient. Therefore, the AGV needs to avoid containers and find a path that is closest to the charging station with the least energy consumption. The starting position of the AGV to find the charging station is defined as the start point, and the location of the charging station is defined as the goal point.

Consider the most complicated situation, that is, the distance between the start point and the goal point is the farthest, as shown in Fig. 3. The start point of red indicates the location before the AGV departure, and the goal point of green indicates the location of the charging station.

In the port map, the movement of the AGV will be restricted by obstacles, as shown in Fig. 3. The distance d_i between the AGV and the obstacle i can be defined as:

$$d_i(x, y) > r_A + r_O, \tag{2}$$

where r_A is the circumscribed radius of the AGV shape; the r_O is the circumscribed radius of the obstacle shape.

While avoiding obstacles, the AGV also needs to reduce the distance of travel. The total distance traveled by the AGV is L_{path}, the calculation formula of L_{path} is

$$L_{path} = \sum_{j=1}^{n-1} \sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2}, \tag{3}$$

where $(x_j,\ y_j)$ represents the j-th grid that the AGV has passed.

To ensure the stability of the AGV when turning, the turning angle θ must be less than the maximum turning angle

$$\theta(n_{i-1}, n_i, n_{i+1}) \leq \theta_{\max}, \tag{4}$$

where n_{i-1}, n_i, and n_{i+1} represent three adjacent nodes; $\theta_{\max}$ is the maximum turning angle, and its value needs to be determined as required.

III. GEOMETRIC A-STAR ALGORITHM FOR PATH PLANNING

The traditional A-star algorithm is a heuristic search algorithm in global path planning. It continuously expands the adjacent nodes from the start point until it reaches the goal point, thus searching for a path around the obstacle from the start point to the goal point. Based on the traditional A-star algorithm, the geometric A-star algorithm shortens the global search path through the functions $P(x, y)$ and $W(x, y)$, thereby eliminating the redundant nodes in the close-list.

A. GEOMETRIC A-STAR ALGORITHM

The geometric A-star algorithm has the following operating steps, and the flow of algorithm is shown in Table 1.

Step 1: Put the 8 nodes adjacent to the start point into the open-list. If the node is an obstacle grid, it is removed from the open-list;

Step 2: Calculate the cost function $f(n)$ of the adjacent nodes, the node with the smallest substitute value is selected as the grid for the next pass, and the start point is placed in

TABLE 1. The flowchart of geometric A-star algorithm.

Algorithm: Geometric A-start

```
algorithm Geomtric A-start(start, n, goal)
if reachAroundGoal(start) ≠ goal then return makePath(start)
open ← closestPoint(start)
closed ← 0
final ← 0
while open ≠ 0 do
  sort(open)
  n ← open.pop( )
  if reachAroundGoal(n) ≠ goal then return makePath(n)
    neighbors ← expendFlexibleUnits(n)
  end if
  for all the neighbors do
    if neighbor ∉ Obstracle
      neighbor.f ← (n.g + g) + (n.p + p) + h
      if neighbor ∩ closed=∅ then open ← neighbor
        else closed ← neighbor
      end if
      closed ← n
      if crossPath(n) and sawtoothPath(n) then
        final ← closed
      end if
    end if
  end for
end
return 0
```

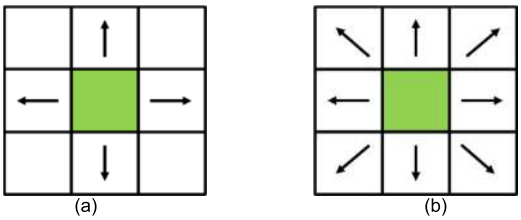

FIGURE 4. Search direction of two distance formulas: (a) Search direction of manhattan distance; (b) Search direction of euler distance.

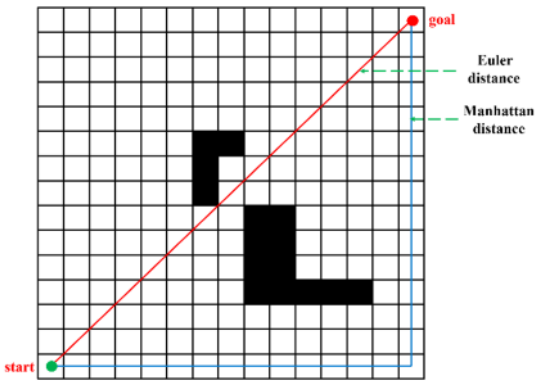

FIGURE 5. Paths generated by the Euler distance and manhattan distance.

the close-list. The calculation formula of the $f(n)$ is

$$f(n) = g(n) + h(n), \tag{5}$$

where $g(n)$ represents the cost from the start point to the current node; the $h(n)$ represents the estimated cost from the current node to the goal point;

Step 3: Put the node adjacent to the current node n and is not placed in the close-list into the open-list. Then use the current node as the parent node of the newly added node. If the adjacent node already exists in the open-list, and judge whether the $g(n)$ from the start point to the current node is smaller; if it is smaller, then use the node as the parent node of the newly added node, and recalculate the $f(n)$ of the node value;

Step 4: Repeat **Step 3** until the goal point is added to the close-list. Then, start the searching from the goal point and move along the parent node to gradually find the start point to form a path from the start point to the goal point;

Step 5: After the goal point is put into the close-list, all nodes in the close-list are judged by the functions $P(x, y)$ and $W(x, y)$, and the nodes that do not meet the requirements in the close-list will be deleted;

Step 6: Determine whether the number of nodes in the close-list decreases after Step 5; if so, continue to **Step 5**; otherwise, exit the loop, save the remaining nodes to the final-list, and generate a complete path;

Step 7: Connect the remaining nodes to form a new path, and smooth the turning path at the same time.

This section mainly introduces the whole process of the geometric A-star algorithm. The improved process mainly includes geometric path optimization and path smoothing. Table 1 shows the steps of the geometric A-star algorithm.

B. DETERMINE THE CALCULATION FORMULA

There are two methods for calculating $h(n)$ in the A-star algorithm: Euler distance and Manhattan distance. Euler distance and Manhattan distance are the two different ways to calculate distance in space. In the A-star algorithm, Euler distance $h_E(n)$ and Manhattan distance $h_M(n)$ can be used to calculate the cost function $f(n)$.

$$\begin{aligned} h_E(n) &= \sqrt{(x_n - x_g)^2 + (y_n - y_g)^2} \\ h_M(n) &= |x_n - x_g| + |y_n - y_g|, \end{aligned} \tag{6}$$

where (x_n, y_n) is the grid coordinate of the current node n; (x_g, y_g) is the grid coordinate of the goal point g.

The A-star algorithms with Manhattan distance can only search in four directions: up, down, left and right; and the A-star algorithm based on Euler distance can use all the eight adjacent nodes as the next search direction. Therefore, Euler distance has more possibilities on the path search than Manhattan distance, as shown in Fig. 4.

As shown in Fig. 5, the coordinates of the start point in the map are set to (0,0), and the coordinates of the goal point are set to (15,15). The red path is a path planned by the A-star algorithm of the Manhattan distance, and the blue path is a path planned by the A-star algorithm of the Euler distance.

By comparing the results of path planning in the same map, it can be found that the A-star algorithm that uses Manhattan distance to calculate $h(n)$ has fewer nodes and turns, and the total distance is also shorter, as shown in Table 2. Therefore,

TABLE 2. Comparison of experimental data from two distance formulas.

Distance formula	Number of nodes	Number of turns	Total distance/m
Manhattan	29	1	19.78
Euler	15	0	28

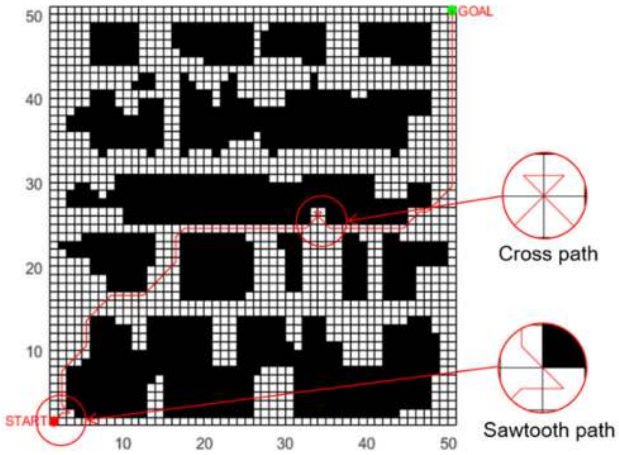

FIGURE 6. Irregular geometric path.

when there are more free grids around the diagonal formed from the start point to the goal point, Euler distance is more suitable than Manhattan distance to calculate $h(n)$. All the follow-up studies in this paper use the evaluation function with Euler distance.

C. IRREGULAR GEOMETRIC PATH OPTIMIZATION

When the AGV uses the traditional A-star algorithm to plan the path in the port map and falls into the local optimum position, it will break away from the local optimum position by generating irregular geometric paths (cross path and swatooth path), as shown in Fig. 6. When there are irregular paths in the generated path, the unnecessary working distance will increase, the number of turns and the turning angle will also increase. Thus, it is necessary to deal with the irregular path to increase the smoothness of the path.

By constructing the filter functions $P(x, y)$ and $W(x, y)$ to judge the nodes in the close-list, and deleting the nodes do not meet the requirements. So as to shorten the path generated by the A-star algorithm.

1) OPTIMIZATION OF CROSS PATHS

The path generated by the A-star algorithm in the grid map is composed of a series of line segments. When multiple paths have the same node, cross paths will be generated. The characteristics and discriminant of the cross path are given for the above situation, as shown in Fig. 7.

The path points passed by the AGV in the sequence is $ABCD$. The coordinates of A and B are (x_A, y_A) and (x_B, y_B) respectively, and the coordinates of C and D are (x_C, y_C) and (x_D, y_D). The generation of the cross path is because the algorithm is stuck in the local optimal position when searching.

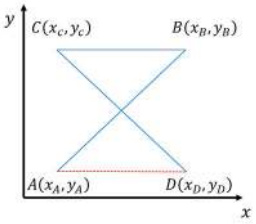

FIGURE 7. Analysis of the cross path.

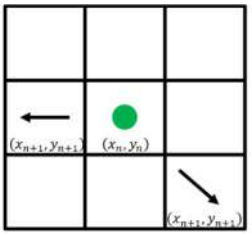

FIGURE 8. Coordinates of the cross path.

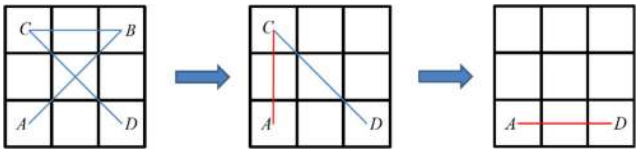

FIGURE 9. Optimization of cross path.

In order to escape the local optimal position, the algorithm can only search in the left or lower right direction. As shown in Fig. 8, the green point n is the location of the current node; the grid pointed by the arrow is the location of the node $n+1$ in the next step of the algorithm selection.

As shown in Fig. 8 above, the current node n and the next node $n+1$ have the following relationship in coordinates:

$$\begin{cases} x_n > x_{n+1} \\ \quad or \\ y_n > y_{n+1} \end{cases}. \tag{7}$$

Therefore, it is only necessary to construct the function $P(x, y)$ to determine whether the two consecutive nodes satisfies the relationship in formula (7). Then decide whether to delete the current node n, and the principle of function $P(x, y)$ can be expressed by formula (8):

$$P(x_n, y_n) = \begin{cases} \emptyset, & x_n > x_{n+1} \cup y_n > y_{n+1} \\ (x_n, y_n), & otherwise \end{cases}, \tag{8}$$

where $\emptyset$ means to assign the coordinates of the point n to null; $\cup$ means that only one of the pre and post conditions needs to be satisfied.

Taking the Fig. 9 as an example, the four nodes in the order of A, B, C, and D constitute the cross path; according to the function $P(x, y)$, there is $x_B > x_c$, node B is deleted, connecting nodes A and C; then the $y_C > y_D$, node C is deleted, connecting nodes A and D, so the path $ABCD$ can be optimized to AD

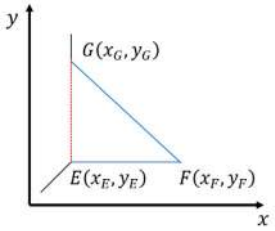

FIGURE 10. Analysis of sawtooth path.

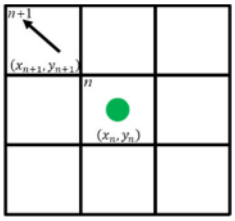

FIGURE 11. Coordinates of sawtooth path.

2) OPTIMIZATION OF SAWTOOTH PATHS

In the grid map, the formation of the sawtooth path is due to the continuous change of direction of the AGV for oblique movement. The characteristics of the sawtooth path are shown in Fig. 10.

The path EFG shown in Fig. 10 is formed in the grid map. The node F can reach the node G, indicating that there are no obstacles between the points E and G. In this case, the redundant node F can be deleted, connect the nodes E and G to form a new path. The above operations can shorten the total length of the global path. The optimization in the grid map requires additional analysis of the coordinates of each point on the irregular path.

As shown in Fig. 11, a function $W(x, y)$ needs to be constructed to determine whether the slope $K_{n,n+1}$ of two adjacent nodes n and $n+1$ is less than zero. When $K_{n,n+1} < 0$, delete the node with the lower serial number from the close-list. The calculation formula of $K_{n,n+1}$ is shown in formula (9):

$$K_{n,n+1} = \frac{y_{n+1} - y_n}{x_{n+1} - x_n}. \tag{9}$$

And the principle of function $W(x, y)$ can be expressed by formula (10):

$$W(x_n, y_n) = \begin{cases} \emptyset, & K_{n,n+1} < 0 \\ (x_n, y_n), & otherwise \end{cases}, \tag{10}$$

where $\emptyset$ means to assign the coordinates of the point n to null.

Taking the Fig. 12 as an example, the three nodes in the order of A, B, and C constitute the sawtooth path. According to the function $W(x, y)$, there is $K_{B,C} < 0$, node B can be deleted, connect the nodes A and C, so the path ABC can be optimized to AC.

The running time Δt represents the time consumed by the AGV from the current position to the charging station

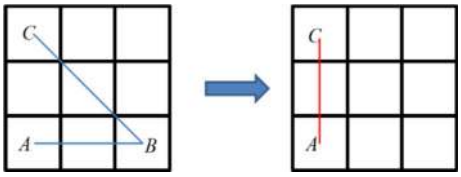

FIGURE 12. Coordinates of sawtooth path.

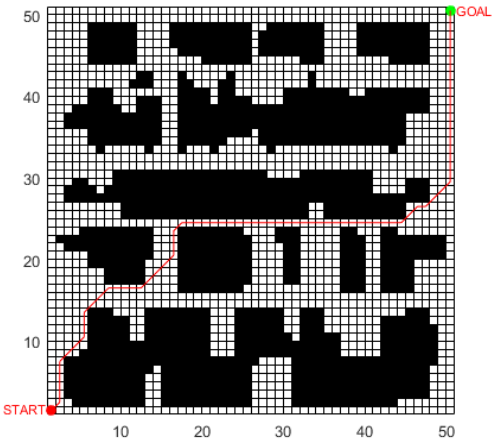

FIGURE 13. Optimized path map of the irregular geometric path.

TABLE 3. Comparison of experimental data between two algorithms.

Path parameters	Traditional A-star	Geometric A-star	Reduced proportion
Running time/s	186.224	176.082	5.4%
Number of nodes	85	80	5.8%
Total distance/m	93.112	88.081	5.4%

position, which can be calculated by the following formula (11):

$$\Delta t = t_f - t_0 = \frac{L}{V}, \tag{11}$$

where L is the total distance traveled by the AGV; V is the average speed of the AGV, the speed of the AGV is generally set to 0.5m/s.

The optimized path can effectively avoid local cross paths and irregular sawtooth paths. The optimized local path is shown in Fig. 13.

Table 3 shows the statistics of the number of nodes and the total distance in the path generated by the algorithms before and after the optimization. The number of nodes before the optimization is 85 and the total distance is 93.112; the number of nodes after the optimization is 80, and the total distance is 88.041. The number of nodes is reduced by 5.8%, the total distance is reduced by 5.4%, and the reduction in running time is 5.4%.

D. SAWTOOTH TURNING PATH

In the path planning, the AGV will inevitably turn when the direction of motion changes, resulting in a turning angle. During the turn, due to a sudden change in the direction

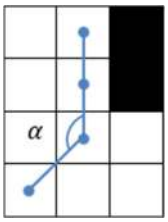

FIGURE 14. Turning angle α.

of motion, the direction of speed and acceleration suddenly changes, which is likely to cause the instability of the AGV's turning motion. To this end, it is proposed to use the cubic B-spline curve to fit the polyline path of turning. To ensure that the AGV can complete the steering stably with a smooth trajectory.

The turning angle during the turn of the AGV is recorded as $\alpha \in [0, 2\pi)$, as shown in Fig. 14.

Compared with the Bezier curve, the B-spline curve has the characteristics of local adjustment and shape closer to the polygon. Therefore, the cubic B-spline is choosed to smooth the planned path. The general formula of B-spline curve is formula (12):

$$Q(t) = \sum_{i=0}^{n} P_i N_{i,r}(t), \tag{12}$$

where P_i is the control point; $N_{i,3}(t)$ is the basic function of the 3rd degree B-spline curve, as shown in formula (13):

$$N_{i,3}(t) = \frac{t - t_i}{t_{i+3} - t_i} N_{i,2}(t) + \frac{t_{i+4} - t}{t_{i+4} - t_{i+1}} N_{i+1,1}(t). \tag{13}$$

When $n = 3$, the basic function of the cubic B-spline function is:

$$N_{i,3}(t) = \begin{cases} \frac{1}{6}(1-t)^3 \\ \frac{1}{6}\left(3t^3 - 6t^2 + 4\right) \\ \frac{1}{6}\left(-3t^3 + 3t^2 + 3t + 1\right) \\ \frac{1}{6}t^3 \end{cases} \quad t \in [0, 1] \tag{14}$$

Convert the cubic B-spline curve formulation into a matrix expression:

$$Q(t) = \frac{1}{6}\begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix} \tag{15}$$

The formulas (12)∼(15) can be found in Ref. 22, and the specific operation flow for smoothing the corner path is as follows:

Step 1: Access the final-list;

Step2: Due to the adjacent nodes in the final-list may be far away from in map after path optimization. The path smoothing will collide with obstacles. Traverse the final-list, and calculate the turning angle α formed by the current node n ($n > 1$), the previous node $n-1$, and the latter node in the final-list is $n+1$ according to formula (16). If $\alpha \neq \pi$, the coordinates of the current node n will be added to the final-list, and the coordinates of other nodes will be represented by spaces. If $\alpha = \pi$, continue to filter the remaining nodes;

Calculation formula of turning angle α is (16), as shown at the bottom of the page.

Step 3: According to the coordinate relationship between the current node n and the previous node $n-1$ in the close-list; the coordinates of the $n-1$ and $n+1$ nodes in the final-list are updated by formulations (17) and (18),

$$if\ x_{n-1} \neq x_n, therefore \begin{cases} x_{n-1} = x_n - 1 \\ y_{n-1} = y_n - 1 \\ x_{n+1} = x_n + 1 \\ y_{n+1} = y_n \end{cases}, \tag{17}$$

$$if\ x_{n-1} = x_n, therefore \begin{cases} x_{n-1} = x_n \\ y_{n-1} = y_n - 1 \\ x_{n+1} = x_n + 1 \\ y_{n+1} = y_n + 1 \end{cases}; \tag{18}$$

Step 4: The cubic B-spline interpolation function in formulation (15) is used to fit the nodes in the final-list;

After fitting the polyline path with a cubic B-spline curve, the smoothed turning path shown in Fig. 16 are obtained.

The entire process of the AGV path planning can be illustrated by the flowchart in Fig. 15.

IV. SIMULATION EXPERIMENT AND RESULT ANALYSIS

A. SIMULATION COMPARISON UNDER DIFFERENT GRID SPECIFICATIONS

Due to the complexity of the latest path planning algorithm, computation time should be derived to evaluate the effectiveness of the geometric A-star algorithm for path planning, and this for grid maps with different specifications. This implied to simulate the path planning of the common A-star algorithm and the geometric A-star algorithm on different grid sizes.

All simulation experiments are carried out under matlab2016b. In order to ensure the accuracy of the results, the same experiment was repeated many times.

$$\alpha = \arccos\left[\frac{(x_{n-1} - x_n)(x_{n+1} - x_n) + (x_{n-1} - x_n)(x_{n+1} - x_n)}{\sqrt{(x_{n-1} - x_n)^2 + (y_{n-1} - y_n)^2}\sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2}}\right]. \tag{16}$$

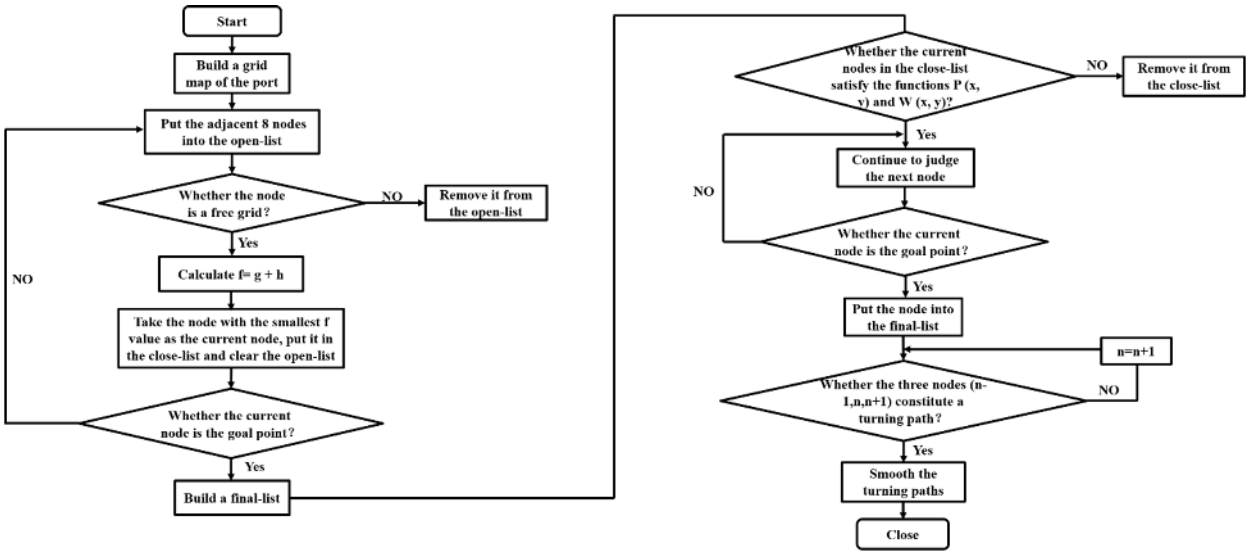

FIGURE 15. Flowchart of the AGV path planning in the port.

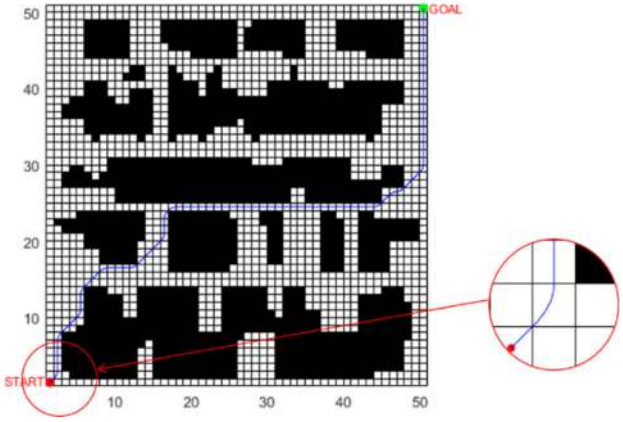

FIGURE 16. Optimized route map of the turning path.

1) THREE KINDS OF GRID MAPS OF THE PORT ENVIRONMENT

The experiments are process as follows. First convert satellite map 2 in Fig. 17a to the grid map 2 with three specifications of 20×20, 50×50, and 100×100. The converted grid maps are shown in Fig. 17b-d.

2) SIMULATION USING THREE SPECIFICATION MAPS

To determine the potential superiority of the improved algorithm for path planning in a given map, the start point and goal point are set at the same position in the three maps. As shown in TABLE 4, grid map number 2.1, 2.2 and 2.3 respectively represent the conversion of the satellite map 2 into three raster images of different specifications.

At the same time, in order to ensure that the start point and the goal point have the same spatial location in the grid map, setting their positions according to the TABLE 4, so the result of path planning will not be affected.

TABLE 4. Coordinates of the start point and end point in the grid map with different of specifications.

Grid map number	Start point	End point	Map specifications
2.1	(0, 0)	(20, 20)	20×20
2.2	(0, 0)	(50, 50)	50×50
2.3	(0, 0)	(100, 100)	100×100

The three pictures in Fig. 18 are the path comparison of the three grid maps with different specifications The routes of the paths generated by the traditional A-star algorithm and the geometric A-star algorithm are highlighted in red and blue, respectively.

Fig. 18a-c are obtained by the rasterization of the same satellite map, so the size of the actual scene represented by the three grid maps is the same.

It can be seen from Table 5 that regarding the path generated by the geometric A-star algorithm planning; the number of nodes has been reduced by more than 19%, the number of turns has been reduced by more than 50%, the maximum turning angle has been reduced by 66.7 %, and the total distance has been reduced by more than 13% compared with the traditional A-star algorithm. The larger the grid's specifications, the better the effect of the geometric A-star algorithm. In the 100×100 map, the number of nodes in the path generated by the geometric A-star algorithm planning has been reduced by 38.7%, and the number of turns has been reduced by 84.1%, the total distance has been decreased by 41.1%, and the reduction in running time is 41.1%.

B. SIMULATION COMPARISON IN DIFFERENT ENVIRONMENTS

In order to verify the effectiveness of the improved algorithm, additional experiments need to be carried out with different

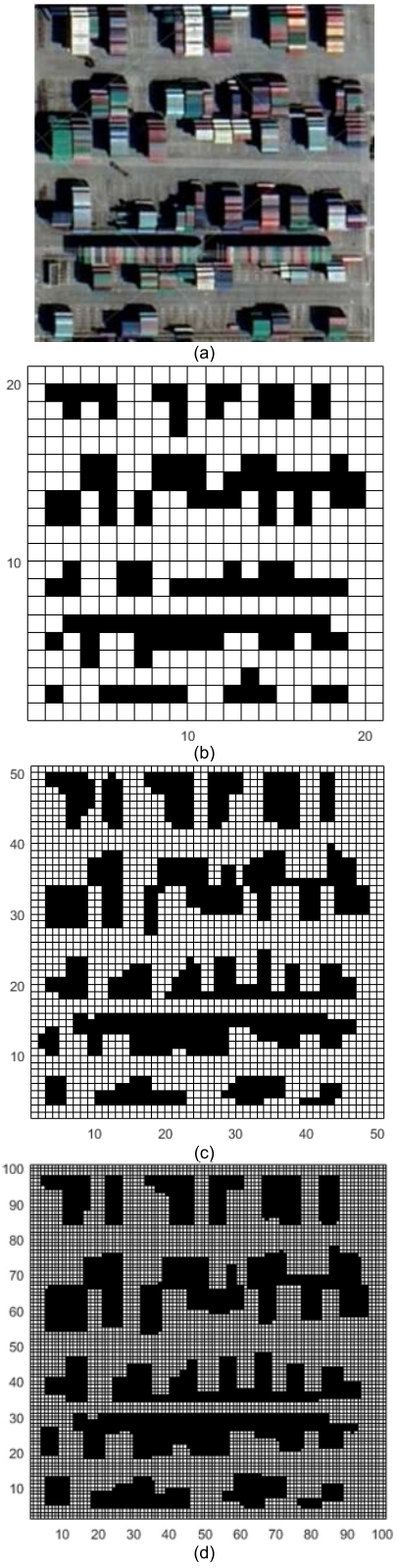

FIGURE 17. Satellite map 2 of port and three grid maps with different specifications:(a) Satellite map 2 pf port; (b) 20 × 20 grid map; (c) 50 × 50 grid map; (d) 100 × 100 grid map.

scenarios. Due to the high similarity of the yard layout in the port, our research scenario mainly focuses on the yard,

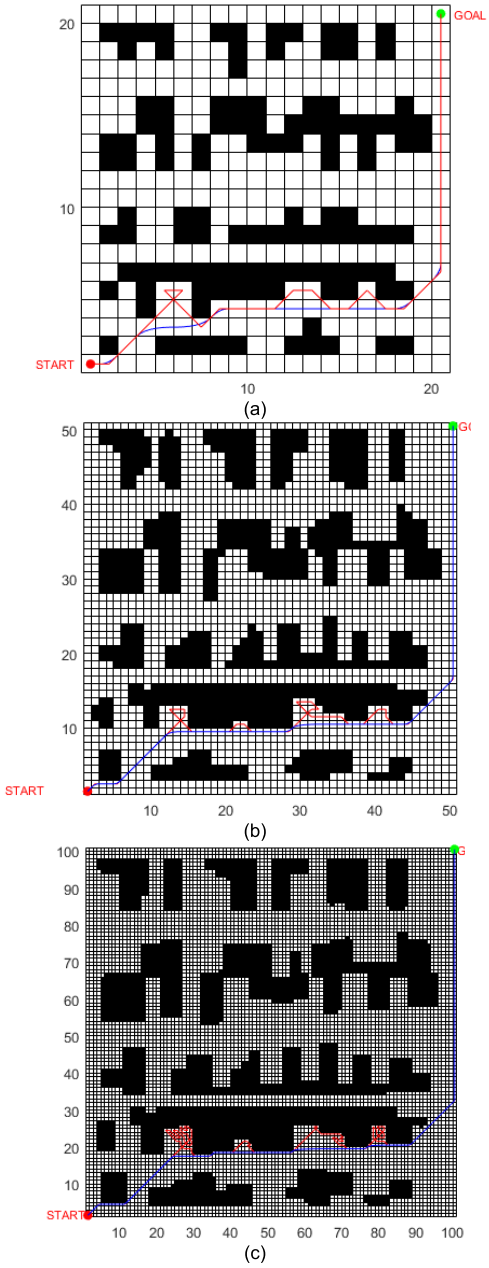

FIGURE 18. Comparison of path of grid maps with three different specifications: (a) Path comparison in the 20 × 20 grid map 2.1; (b) Path comparison in the 50 × 50 grid map 2.2; (c) Path comparison in the 100 × 100 grid map 2.3.

which only needs to use the common A-star algorithms and the geometric A-star algorithms for path planning simulation experiments in three port environments with the same specifications and different scenarios.

1) GRID MAPS OF THREE PORT ENVIRONMENTS

Select three different scenes of the port environment and convert it into the 100 × 100 grid maps, as shown in Fig. 19, where the Fig. 19a-c are the satellite map 3, 4, 5 of port, and the grid maps obtained by rasterisations.

TABLE 5. Comparison of experimental data in three grid maps with different specifications.

Map size	Path parameters	Traditional A-star	Geometric A-star	Reduced proportion
20×20	Running time/s	80.768	69.602	13.8%
	Number of nodes	36	29	19.4%
	Number of turns	14	6	57.1%
	Max turning angle	135°	45°	66.7%
	Total distance/m	201.92	174.005	13.8%
50×50	Running time/s	215.68	177.796	17.6%
	Number of nodes	96	71	26%
	Number of turns	25	7	72%
	Max turning angle	135°	45°	66.7%
	Total distance/m	215.68	177.796	17.6%
100×100	Running time/s	609.198	358.69	41.1%
	Number of nodes	230	141	38.7%
	Number of turns	69	11	84.1%
	Max turning angle	135°	45°	66.7%
	Total distance/m	304.599	179.345	41.1%

By comparing the simulation results of the two algorithms in different grid maps, the effectiveness of the improved algorithm can be observed.

To ensure the directions of the planned routes in the three test environments are roughly the same, setting the start point in the grid map is at the minimum coordinate point, and the goal point is at the maximum coordinate point.

Grid maps 3, 4, and 5 have the same specifications. In order to verify the effectiveness of the algorithm and control irrelevant variables, this paper set the start point and goal point of the path planning at the same position in the three maps. Therefore, in the grid map 3, 4 and 5; the start point is set to (0, 0), and the goal point is set to (100, 100).

2) SIMULATION RESOLUTION IN THE THREE SATELLITE MAPS OF DIFFERENT PORT

The three pictures in Fig. 20 are the path planning results generated by the common A-star algorithm and the geometric A-star algorithm in three grid maps of different ports. The red path and the blue path respectively represent the paths generated by the traditional A-star algorithm and the geometric A-star algorithm.

The generated path can be seen in Fig. 20. The improvement effect is not obvious in the simple map (map 3). The geometric A-star algorithm in the complex map (map 4 and map 5) can better optimize the sawtooth and cross path than the traditional A-star algorithm. Therefore, the geometric A-star algorithm is more suitable for ports with more complex environments.

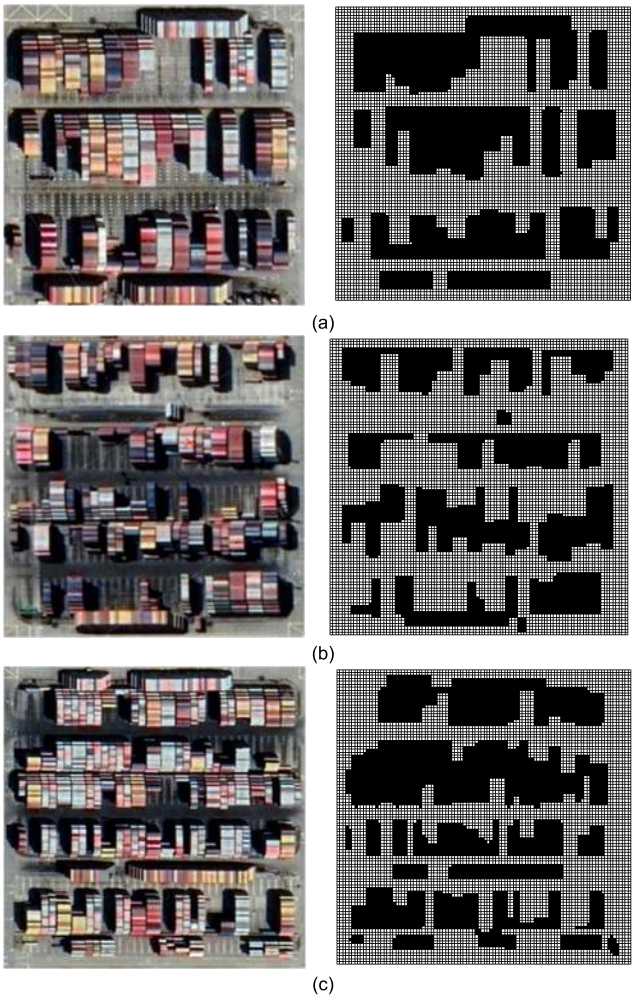

FIGURE 19. Three different satellite map of port and their corresponding grid maps: (a) Sellite map 3 and its grid map 3; (b) Sellite map 4 and its grid map 4; (c) Sellite map 5 and its grid map 5.

The simulation results of the traditional A-star algorithm and the geometric A-star algorithm in the three grid maps of the different port are shown in TABLE 6. In the simple map (map 3), compared with the traditional A-star algorithm, the path generated by the geometric A-star algorithm is basically the same, and the total distance is only shortened by 0.18 %. In the complex maps (map 4 and map 5), compared with the traditional A-star algorithm, the number of nodes in the path generated by the geometric A-star algorithm is reduced by more than 24%, the number of turns is reduced by more than 56%, the maximum turning angle is reduced by 66.7%, and the total distance has been shortened by more than 15%. There has been no significant increase in running time. It shows that the geometric A-star algorithm can plan a better path in different environments.

C. SIMULATION COMPARISON BETWEEN DIFFERENT ALGORITHMS

In order to illustrate the effectiveness of the geometric A-star algorithm, its performances are evaluated with a series

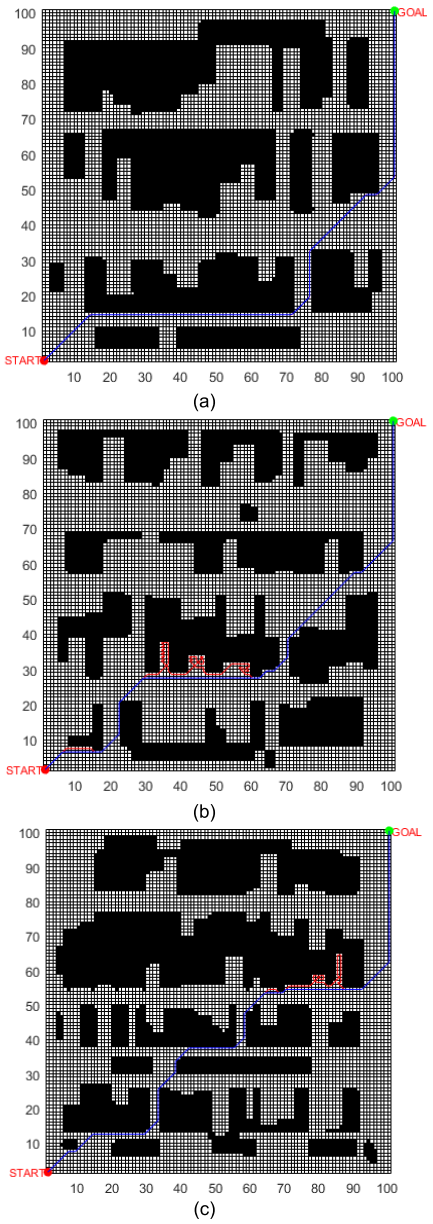

FIGURE 20. Comparison of routes in three satellite maps of different port: (a) Path comparison in grid map 3; (b) Path comparison in grid map 4; (c) Path comparison in grid map 5.

of algorithms (Dijkstra, A-star, Bestfirt, BFS, Bidirectional A-star and DFS) applied to a reference grid map and path generation according to a few given parameters. The main process of the comparative experiment includes two steps. First, a port-related satellite map is extracted from Google Earth Maps (Fig. 21 satellite map 6).

Next, this satellite map is rasterized and converted to a grid map, this gives a grid map of 100 × 100 pixels (Fig. 22). The start and goal points are predefined from the bottom left (0, 0) to the top right of the grid map (100,100). Seven algorithms are then applied and computed. Fig. 22b to Fig. 22h show the simulation results of the 7 algorithms. In each figure the red route represents the path generated by the algorithm,

TABLE 6. Comparison of experimental data in three grid maps of different environments.

Map number	Path parameters	Traditional A-star	Geometric A-star	Reduced proportion
3	Running time/s	351.5	349.67	0.18%
	Number of nodes	160	160	0%
	Number of turns	7	7	0%
	Max turning angle	45°	45°	0%
	Total distance/m	175.15	174.835	0.18%
4	Running time/s	453.58	336.24	25.9%
	Number of nodes	193	110	43%
	Number of turns	60	9	85%
	Max turning angle	135°	45°	66.7%
	Total distance/m	226.79	168.12	25.9%
5	Running time/s	609.198	358.69	41.1%
	Number of nodes	409.22	345.264	15.6%
	Number of turns	182	138	24.2%
	Max turning angle	39	17	56.4%
	Total distance/m	135°	45°	66.7%

FIGURE 21. Satellite map 6.

while the blue and yellow point areas denote the node area searched by the algorithm. The denser the node area, the more nodes need to be expanded and higher the computational time is.

The red route in the Fig. 22 represents the path generated by the algorithm, and the blue and yellow point area represents the node area searched by the algorithm. The denser the node area, the more nodes that need to be expanded and the higher computation time will be.

It can be seen from the path generated and the expanded node area as denoted by Fig. 22a to Fig. 22h that the expanded node area generated by the BestFirst algorithm, DFS algorithm and the geometric A-star algorithm are the smallest.

Table 7 shows the computational times and geometrical properties of the generated paths for all algorithms. Overall, it clearly appears that the Geometric A-star algorithm performs much better in most if not cases with respect to all parameters as well as regarding computational time.

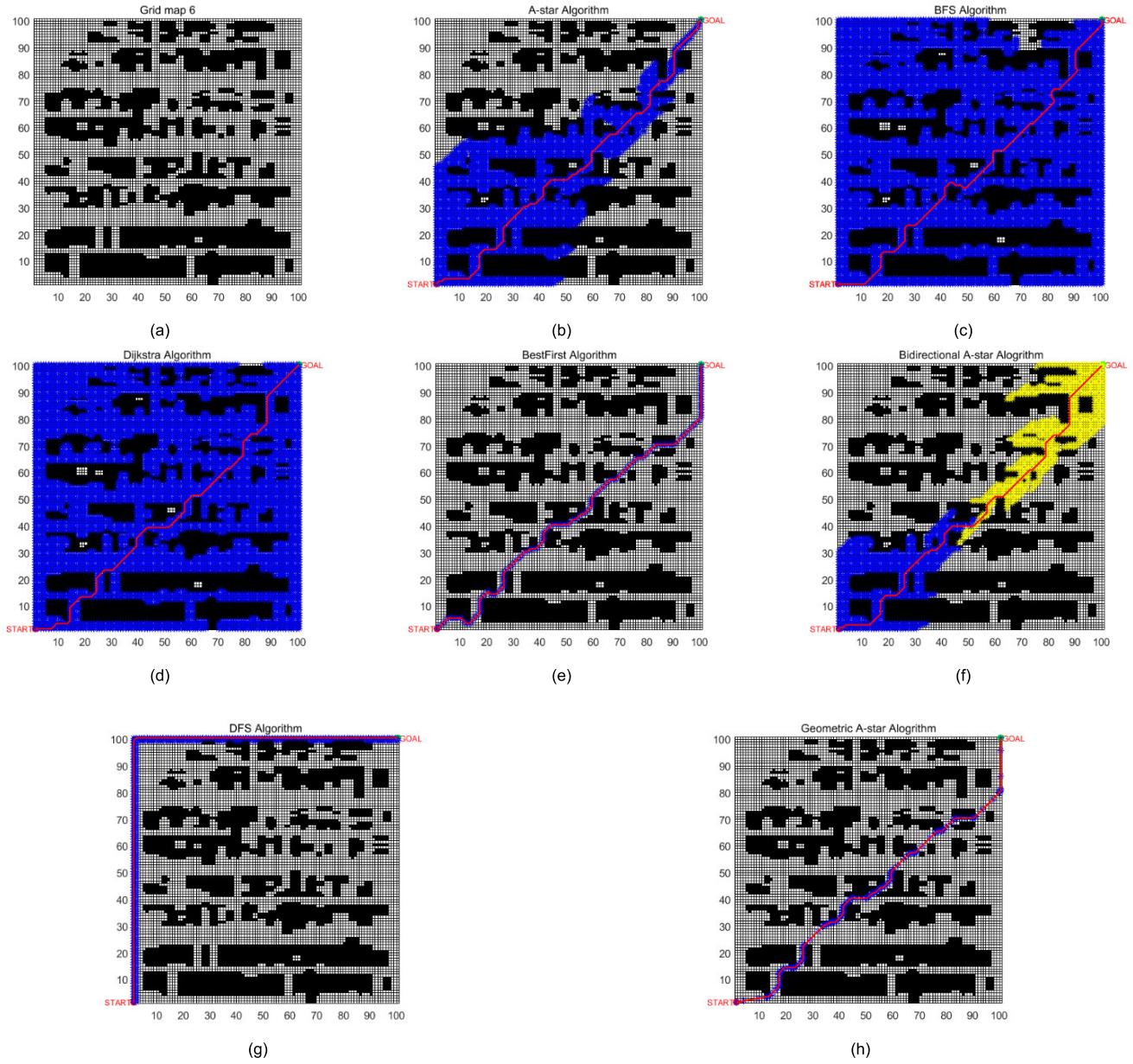

FIGURE 22. Grid map 6 and path results in the grid map 6 generated by 7 algorithms: (a) Grid map 6; (b) Path result generated by A-star algorithm; (c) Path result generated by BFS algorithm; (d) Path result generated by dijkstra algorithm; (e) Path result generated by bestfirst algorithm; (f) Path result generated by bidirectional a-star algorithm; (g) Path result generated by DFS algorithm; (h) Path result generated by geometric a-star algorithm.

TABLE 7. Comparison of experimental data between different algorithms.

Map	Path parameters	A-star	BFS	Dijkstra	Bestfirst	Bidirectional A-star	DFS	Geometric A-star	Reduce scope/%
6	Running time/s	316.334	322.962	316.334	396.642	316.334	394.83	295.142	6.7~25.5
	Number of nodes	131	131	131	136	131	198	109	16.8~44.9
	Number of turns	36	33	27	32	30	1(exclude)	27	0~25
	Max turning angle	45°	135°	45°	135°	45°	90°	45°	0~33.3
	Number of expansion nodes	2246	5936	6047	136	1611	198	109	19.9~98.2
	Total distance/m	158.167	161.481	158.167	197.851	158.167	197.415	147.571	6.7~25.5

V. CONCLUSION

This paper introduces a new path planning algorithm, so-called geometric A-star algorithm, and whose objective is to improve the irregular path generated by the common A-star algorithm. Combined with the advantages of an interpolation algorithm, the experiments show that our algorithm has a high success rate, generates feasible paths, while the screening speed is relatively fast. Firstly, a feasible path is generated by the A-star algorithm, then the irregular path is optimized by a screening function, and finally the whole path is smoothed by an interpolation algorithm. Overall, our approach can effectively reduce the number of turns in the AGV motion process, shorten the path length, and make the AGV complete the turn with a small turning angle.

There are several directions for future work. First, and so far, the size of the AGV has not been taken into account as considered as a punctual feature, this might be taken into account when computing obstacle avoidances.

Second, our algorithm is mainly applied to static scenes, and the impact of mobile vehicles in dynamic scenes has not been considered. Future work should study complex dynamic scenes, this being an important expectation in real-world contexts. Third, due to the AGV's motion mode which is actually a main straight line, the motion of the real AGV is not completely reflected. Last but not least, we plan to implement the whole approach and algorithm into an experimental AGV setup in a real port environment.

REFERENCES

[1] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Ensemble coordination approach in multi-AGV systems applied to industrial warehouses," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 922–934, Jul. 2015.

[2] Y. Yang, M. Zhong, Y. Dessouky, and O. Postolache, "An integrated scheduling method for AGV routing in automated container terminals," *Comput. Ind. Eng.*, vol. 126, pp. 482–493, Dec. 2018.

[3] C. Chen, D. Tran Huy, L. K. Tiong, I.-M. Chen, and Y. Cai, "Optimal facility layout planning for AGV-based modular prefabricated manufacturing system," *Autom. Construct.*, vol. 98, pp. 310–321, Feb. 2019.

[4] M. Zhong, Y. Yang, Y. Dessouky, and O. Postolache, "Multi-AGV scheduling for conflict-free path planning in automated container terminals," *Comput. Ind. Eng.*, vol. 142, Apr. 2020, Art. no. 106371.

[5] M. De Ryck, M. Versteyhe, and K. Shariatmadar, "Resource management in decentralized industrial automated guided vehicle systems," *J. Manuf. Syst.*, vol. 54, pp. 204–214, Jan. 2020.

[6] I. Draganjac, D. Miklic, Z. Kovacic, G. Vasiljevic, and S. Bogdan, "Decentralized control of multi-AGV systems in autonomous warehousing applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1433–1447, Oct. 2016.

[7] C. D. B. Borges, A. M. A. Almeida, I. C. P. Júnior, and J. J. D. M. Sá Junior, "A strategy and evaluation method for ground global path planning based on aerial images," *Expert Syst. Appl.*, vol. 137, pp. 232–252, Dec. 2019.

[8] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robot. Auto. Syst.*, vol. 86, pp. 13–28, Dec. 2016.

[9] M. Candeloro, A. M. Lekkas, and A. J. Sørensen, "A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels," *Control Eng. Pract.*, vol. 61, pp. 41–54, Apr. 2017.

[10] J.-S. Chou, M.-Y. Cheng, Y.-M. Hsieh, I.-T. Yang, and H.-T. Hsu, "Optimal path planning in real time for dynamic building fire rescue operations using wireless sensors and visual guidance," *Autom. Construct.*, vol. 99, pp. 1–17, Mar. 2019.

[11] K. J. C. Fransen, J. A. W. M. van Eekelen, A. Pogromsky, M. A. A. Boon, and I. J. B. F. Adan, "A dynamic path planning approach for dense, large, grid-based automated guided vehicle systems," *Comput. Oper. Res.*, vol. 123, Nov. 2020, Art. no. 105046.

[12] A. Majeed and S. Lee, "A fast global flight path planning algorithm based on space circumscription and sparse visibility graph for unmanned aerial vehicle," *Electronics*, vol. 7, no. 12, p. 375, Dec. 2018.

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

[14] L. Xie, S. Xue, J. Zhang, M. Zhang, W. Tian, and S. Haugen, "A path planning approach based on multi-direction A* algorithm for ships navigating within wind farm waters," *Ocean Eng.*, vol. 184, pp. 311–322, Jul. 2019.

[15] D. Tsatcha, É. Saux, and C. Claramunt, "A bidirectional path-finding algorithm and data structure for maritime routing," *Int. J. Geographical Inf. Sci.*, vol. 28, no. 7, pp. 1355–1377, Jul. 2014.

[16] X. Liu, Y. Li, J. Zhang, J. Zheng, and C. Yang, "Self-adaptive dynamic obstacle avoidance and path planning for USV under complex maritime environment," *IEEE Access*, vol. 7, pp. 114945–114954, 2019.

[17] P. K. Das, H. S. Behera, and B. K. Panigrahi, "A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning," *Swarm Evol. Comput.*, vol. 28, pp. 14–28, Jun. 2016.

[18] A. Bakdi, A. Hentout, H. Boutami, A. Maoudj, O. Hachour, and B. Bouzouia, "Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control," *Robot. Auto. Syst.*, vol. 89, pp. 95–109, Mar. 2017.

[19] C. Xiong, D. Chen, D. Lu, Z. Zeng, and L. Lian, "Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization," *Robot. Auto. Syst.*, vol. 115, pp. 90–103, May 2019.

[20] H. Park and J.-H. Lee, "*B*-spline curve fitting based on adaptive curve refinement using dominant points," *Comput.-Aided Des.*, vol. 39, no. 6, pp. 439–451, 2007.

[21] E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, pp. 497–516, 1957.

[22] M. Luo, X. Hou, and J. Yang, "Surface optimal path planning using an extended Dijkstra algorithm," *IEEE Access*, vol. 8, pp. 147827–147838, 2020.

[23] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Eng.*, vol. 96, pp. 59–69, 2014.

[24] W. E. Howden, "The sofa problem," *Comput. J.*, vol. 11, no. 3, pp. 299–301, Mar. 1968.

[25] M. Goldenberg, "The heuristic search research framework," *Knowl.-Based Syst.*, vol. 129, pp. 1–3, Aug. 2017.

[26] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Comput.*, vol. 21, no. 19, pp. 5829–5839, Oct. 2017.

[27] T. T. Mac, C. Copot, D. T. Tran, and R. D. Keyser, "A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization," *Appl. Soft Comput.*, vol. 59, pp. 68–76, Oct. 2017.

[28] B. Fu, L. Chen, Y. Zhou, D. Zheng, Z. Wei, J. Dai, and H. Pan, "An improved A* algorithm for the industrial robot path planning with high success rate and short length," *Robot. Auto. Syst.*, vol. 106, pp. 26–37, Aug. 2018.

[29] R. Song, Y. Liu, and R. Bucknall, "Smoothed A* algorithm for practical unmanned surface vehicle path planning," *Appl. Ocean Res.*, vol. 83, pp. 9–20, Feb. 2019.

[30] R. A. Saeed, D. R. Recupero, and P. Remagnino, "A boundary node method for path planning of mobile robots," *Robot. Auto. Syst.*, vol. 123, Jan. 2020, Art. no. 103320.

[31] J. Han and Y. Seo, "Mobile robot path planning with surrounding point set and path improvement," *Appl. Soft Comput.*, vol. 57, pp. 35–47, Aug. 2017.

[32] L. Huang, H. Qu, P. Ji, X. Liu, and Z. Fan, "A novel coordinated path planning method using k-degree smoothing for multi-UAVs," *Appl. Soft Comput.*, vol. 48, pp. 182–192, Nov. 2016.

[33] X. Wu, L. Xu, R. Zhen, and X. Wu, "Bi-directional adaptive A* algorithm toward optimal path planning for large-scale UAV under multi-constraints," *IEEE Access*, vol. 8, pp. 85431–85440, 2020.

[34] X. Miao, J. Lee, and B.-Y. Kang, "Scalable coverage path planning for cleaning robots using rectangular map decomposition on large environments," *IEEE Access*, vol. 6, pp. 38200–38215, 2018.
[35] Y.-C. Wang and K.-C. Chen, "Efficient path planning for a mobile sink to reliably gather data from sensors with diverse sensing rates and limited buffers," *IEEE Trans. Mobile Comput.*, vol. 18, no. 7, pp. 1527–1540, Jul. 2019.
[36] A. Ravankar, A. Ravankar, Y. Kobayashi, Y. Hoshino, and C.-C. Peng, "Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges," *Sensors*, vol. 18, no. 9, p. 3170, Sep. 2018.

GANG TANG received the M.S. degree in mechanical engineering from the Harbin Institute of Technology and the Ph.D. degree in mechanical engineering from Shanghai Jiao Tong University. His research interests include artificial intelligence and its application to many real-world domains, including the maritime environments.

CONGQIANG TANG received the B.S. degree in mechanical engineering from the Hubei University of Technology, in 2017. He is currently pursuing the M.S. degree in mechanical engineering with Shanghai Maritime University. His research interest includes unmanned aerial vehicle (UAV) path planning.

CHRISTOPHE CLARAMUNT received the Ph.D. degree in computer science from the University of Burgundy and the "Habilitation à Diriger des Recherches" from the University of Rouen. He is currently a Professor of computer science and acting as the Research Chair of the French Naval Academy. Amongst many international acting visiting fellowships, he is also a Research Fellow with Shanghai Maritime University. His research interests include spatio-temporal models and theories, semantic and cognitive-based GIS, Web and wireless GIS systems, and maritime, environmental, and urban GIS.

XIONG HU received the bachelor's and Ph.D. degrees from Shanghai Jiaotong University. He was a Visiting Professor with the Queensland University of Technology (QUT), Australia. He is currently a Professor and a Ph.D. Supervisor with Shanghai Maritime University (SMU), the Dean of the Logistics Engineering College, SMU, and the Dean of the Sino-Dutch Mechatronics Engineering College, SMU. His research and teaching interests include (health) condition monitoring, (remote) control, (condition) assessment and (maintenance) management of large equipment and its structure, and intelligent processing and prediction of health condition data and signals of machines.

PEIPEI ZHOU received the B.S. and Ph.D. degrees from the South China University of Technology, Guangzhou, China, in 2011 and 2018, respectively. From 2014 to 2016, she was a Visiting Student with the Polytech'Nantes (Ecole Polytechnique de l'universite de Nantes), Nantes, France. From 2018 to 2020, she was a Postdoctoral Fellow with Sun Yat-sen University. She is currently an Associate Professor with the School of Mechatronic Engineering, Guangdong Polytechnic Normal University, Guangzhou. Her current research interests include microfluidic systems, automation, and deep learning.