Development of Path Planning Approach Using Improved A-star Algorithm in AGV System

Yan Zhang[1], Ling-ling Li[2], Hsiung-Cheng Lin[3], Zewen Ma[1], Jiang Zhao[1]

[1] School of Electrical Engineering, Hebei University of Science and Technology, China
[2] School of Electrical Engineering, Hebei University of Technology, China
[3] Department of Electronic Engineering, National Chin-Yi University of Technology, Taiwan.
yanyanfly163@163.com, lilinglinglaoshi@126.com, hclin@ncut.edu.tw, 601628752@qq.com, kdzhaojiang@163.com

Abstract

Automated guided vehicles (AGV) has been widely applied in industry, especially in warehousing, logistics and port transportation, etc. The most important issue in the AGV control system is how to determine the shortest path. Although A-star algorithm is usually used to solve this problem due to its fast computational time, it may suffer from some drawbacks, for instance, there are too many unnecessary inflection points and redundant nodes during its moving process. For this reason, the improved A-star algorithm is proposed to optimize the motion path, including the reduction of path length, number of AGV turns and path planning time. In this scheme, a key point selection strategy is used for the secondary planning based on the initial path obtained by the A-Star algorithm. Therefore, redundant inflection points and nodes in the path can be effectively removed. Additionally, the rotation direction and rotation angle of AGV at the inflection point can be thus determined. This improved A-star method can provide a more efficient path planning with shorter routes, less turn times and shorter operation time compared with previous methods, such as the A-star algorithm and the ant colony algorithm.

Keywords: Automated guided vehicles (AGV), A-star algorithm, Path planning, Heuristic search algorithm

1 Introduction

In recent years, AGV is regarded as an automated device that can achieve unmanned operations in the transportation of materials in industry [1-2]. It is known that the transport efficiency of AGV is affected by its motion path. For example, a long transport path will not only result in more AGV energy consumption but also higher transportation cost. Besides, AGV needs to be slowed down to change the direction due to excessive AGV turns. Sequentially, the produced inertia may influence the safety of AGV motion, especially in a large AGV. Accordingly, it is an indispensable task to find an efficient path for AGV.

The purpose of path planning is to plan an optimal path for control object from the start point to the end point under the circumstance of surrounding with obstacles [3-5]. At present, some path planning algorithms such as ant colony algorithm, genetic algorithm, A-star algorithm, etc have been reported [6-7]. Ant colony algorithm has good global optimization ability, but it is easy to fall into a local optimization solution due to a large amount of computation [8-10]. On the other hand, genetic algorithm has an advantage in the iterative convergence. However, its computational efficiency is low [11]. A global path planning algorithm which combines the artificial potential field method and ant colony algorithm was reported [12]. Although it can improve the convergence speed of ant colony algorithm, a rather complex calculation is required.

A-star algorithm has been successfully applied in gaming industry because of its fast computing speed with the shortest path [13-17]. It was also used in AGV path planning, but it may suffer from inflection point and redundant nodes [18-19]. Another method called Hierarchical Path-Finding A* (HPA*) can reduce the complexity in path-finding on grid-based maps with a faster searching speed, however, which usually cannot achieve the optimal solution [20]. To resolve limitations of above methods, this paper aims to improve A-star algorithm for not only preserving its advantage like fast computing speed but also deleting redundant inflection nodes or redundant nodes on the path. The optimal path planning can therefore achieve a full autonomy.

2 Fundamental of A-star Algorithm

A-star algorithm uses heuristic function to estimate the distance between an arbitrary position on the plane and target position. The algorithm takes the starting node as the current node (parent node). The surrounding nodes (children nodes) in the eight

*Corresponding Author: Hsiung-Cheng Lin; E-mail: hclin@ncut.edu.tw
DOI: 10.3966/160792642019052003023

directions of the current node are thus evaluated by the heuristic function, as shown in Figure 1. The node selected as the next parent node is based on the smallest evaluation value, and the iteration loop continues until the target point is found. Each parent node is the smallest evaluation node, where the path formed by these parent nodes is the optimal path planned by the A-star algorithm.

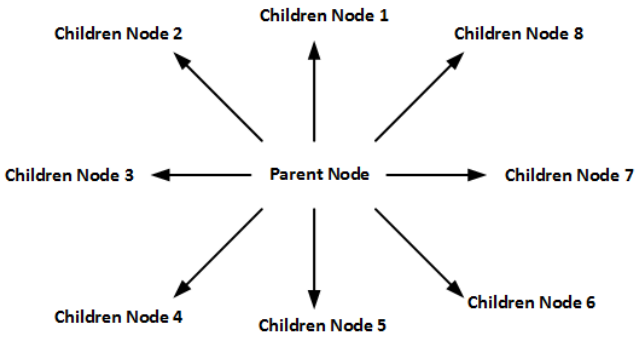


Figure 1. The direction of children nodes

Evaluation function $F(n)$ is defined as

$$F(n) = G(n) + H(n) \quad (1)$$

Where n is the current node, $G(n)$ is the cost of moving from the starting points along the generated path to the current node n, and $H(n)$ is the minimum cost estimate from the current node n to the ending nodes. $F(n)$ is the total cost of the path.

According to the coordinates of the current node n and target nodes (x_1,y_1) and (x_2,y_2), $H(n)$ is defined as

$$H(n) = sqrt[(x_1 - x_2)^2 + (y_1 - y_2)^2] \quad (2)$$

$$F(n) = G(n) + sqrt[(x_1 - x_2)^2 + (y_1 - y_2)^2] \quad (3)$$

A-star algorithm flow is shown in Figure 2. There are two data tables in the A-star algorithm flow, where one is open table and another one is close table. The data in the open table need to be sorted, and the data in the close table is the node of the path.

Step 1: Place the starting node into the open table and calculate $F(n)$. **Step 2**: Place the node of minimum $F(n)$ in the open table, and place the $F(n)$ of the node into the close table. **Step 3**: Determine if the current node in the close table is an ending node. If yes, the A-star algorithm process ends. If not, continue the step 4. **Step 4**: Expand the children nodes of the current nodes: calculate their $F(n)$ and place the nodes and $F(n)$ of the nodes into open table. **Step 5**: Sort the open table.

The flowchart of extending the surrounding nodes of the parent nodes by unit step is shown in Figure 3. If the node is located on the planning path, the node whose $F(n)$ is minimal is called the parent node. The nodes around the parent node unit step are called the children nodes, and then the node P is a children node of the current parent node. The same children node may be extended when expending the nodes on the

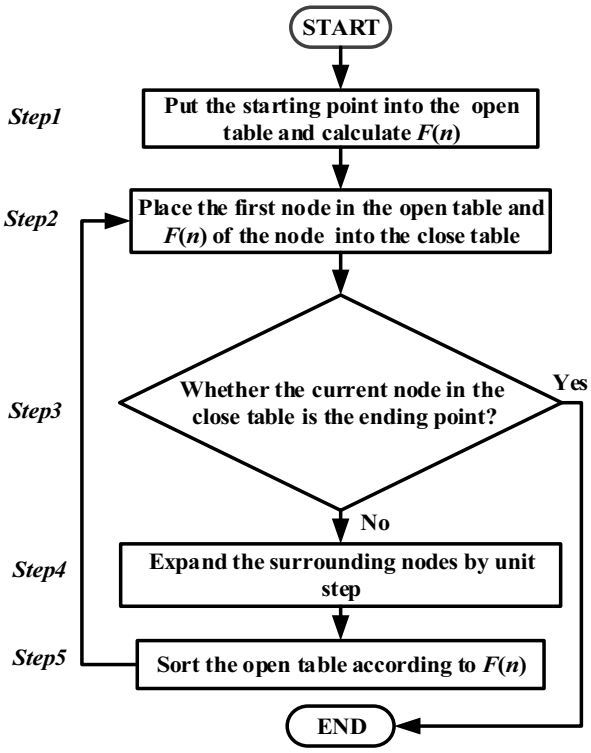


Figure 2. Flowchart of A-star algorithm

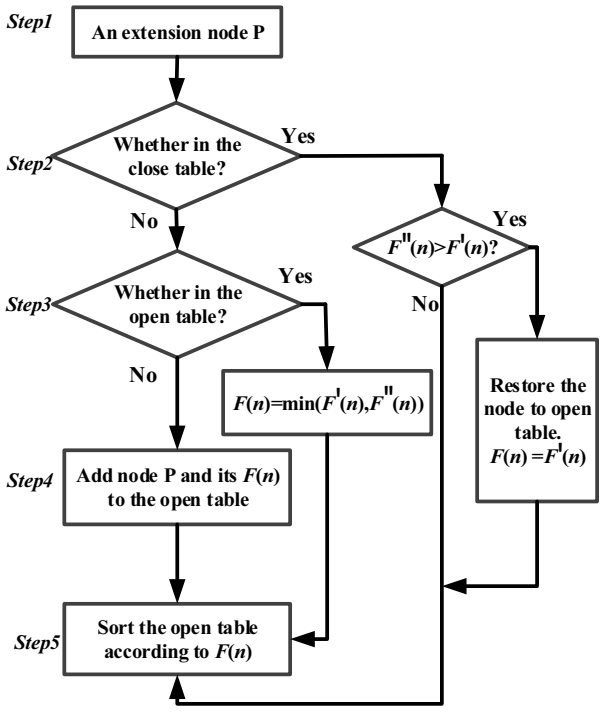


Figure 3. Process of extending the surrounding nodes

path by unit step. Therefore, $F'(n)$ is calculated from $F(n)$ when P is the children nodes of the current parent node, and $F''(n)$ is calculated from $F(n)$ when P is the children nodes of another previous parent node.

Step 1: Expand a node P. **Step 2**: Determine whether the node P is already in the close table. If not, run the step 3. Otherwise, determine whether its $F'(n)$ is less than $F''(n)$. If not, run the step 5. Otherwise,

restore the node to the open table. Let $F(n)=F'(n)$, and run step 5. **Step 3**: Determine whether the node P is already in the open table. If not, run the step 4. Otherwise, let $F(n)=\min(F'(n), F''(n))$ and run step 5. **Step 4**: Add node P and its $F(n)$ to the open table. **Step 5**: Sort the open table according to $F(n)$.

3 The Principle of Improved A-star Algorithm

The improved A-star algorithm consists of two parts: (1) Delete collinear nodes and redundant inflection nodes of the initial path obtained by the A-Star algorithm. At the last, only the path that contains the starting node, the end node and the key inflection is left. (2) Calculate the rotation direction and rotation angle of AGV at the inflection based on node coordinates of the improved path.

3.1 The Method of Deleting Nodes in the Same Line

Define the node B as the current node of the path, where the coordinate is (x_2,y_2). Node A is the previous node of node B, where the coordinate is (x_1,y_1). Node C is the next node of node B, where the coordinate is (x_3,y_3).

$$S_{AB}=\frac{y_2-y_1}{x_2-x_1} \tag{4}$$

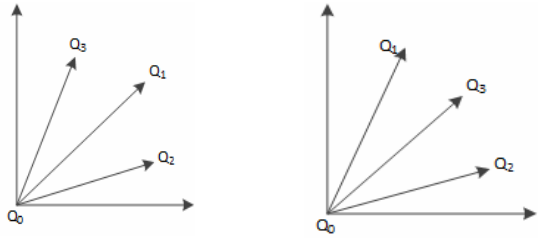
$$S_{BC}=\frac{y_3-y_2}{x_3-x_2} \tag{5}$$

where S_{AB} is the slope of the segment AB, and S_{BC} is the slope of the segment BC.

If $S_{AB}=S_{BC}$, the nodes A, B, C are in the same line. As a result, the node B is confirmed a redundant node, and it needs to be removed.

3.2 The Method of Deleting Superfluous Inflection Points

As shown in the Figure 4, there are four points in the plane: Q_0 is defined as the origin, and the coordinates Q_1, Q_2, Q_3 are respectively at different locations. The vector formed by Q_0 and Q_1 is $\overrightarrow{Q_0Q_1}$, the vector formed by Q_0 and Q_2 is $\overrightarrow{Q_0Q_2}$, and the vector formed by Q_0 and Q_3 is $\overrightarrow{Q_0Q_3}$, $\overrightarrow{Q_0Q_1}$, $\overrightarrow{Q_0Q_2}$ and $\overrightarrow{Q_0Q_3}$ are regarded as a three-dimension whose Z-axis coordinates are zero. The cross product by $\overrightarrow{Q_0Q_1}$ and $\overrightarrow{Q_0Q_2}$ is shown in formula (6), and the cross product by $\overrightarrow{Q_0Q_1}$ and $\overrightarrow{Q_0Q_3}$ is shown in formula (7). Note that the vectors $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ are unit vectors.



(a) Q_2, Q_3 are located on both sides of $\overrightarrow{Q_0Q_1}$

(b) Q_2, Q_3 are located on the same side of $\overrightarrow{Q_0Q_1}$

Figure 4. Determine the position of two points

$$\overrightarrow{Q_0Q_1}\times\overrightarrow{Q_0Q_2}=\begin{bmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\\ x_{Q_1} & y_{Q_1} & 0\\ x_{Q_2} & y_{Q_2} & 0\end{bmatrix}=(x_{Q_1}y_{Q_2}-x_{Q_2}y_{Q_2})\boldsymbol{k}=a\boldsymbol{k} \tag{6}$$

$$\overrightarrow{Q_0Q_1}\times\overrightarrow{Q_0Q_3}=\begin{bmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\\ x_{Q_1} & y_{Q_1} & 0\\ x_{Q_3} & y_{Q_3} & 0\end{bmatrix}=(x_{Q_1}y_{Q_3}-x_{Q_3}y_{Q_1})\boldsymbol{k}=b\boldsymbol{k} \tag{7}$$

The cross product of the two vectors is a vector which is perpendicular to these two vectors. It is stipulated that the outward direction is the positive direction. Note that the a and the b in the above formulas (6) and (7) are the coefficients of cross products. According to right hand rule, if the cross product which crossed by $\overrightarrow{Q_0Q_2}$ and $\overrightarrow{Q_0Q_1}$ is inward, i.e. $a<0$, so that $\overrightarrow{Q_0Q_2}$ is in clockwise direction of $\overrightarrow{Q_0Q_1}$. That is to say the point Q_2 is in the clockwise direction of $\overrightarrow{Q_0Q_1}$. The cross product crossed by $\overrightarrow{Q_0Q_3}$ and $\overrightarrow{Q_0Q_1}$ is outward, i.e. $b>0$, so that $\overrightarrow{Q_0Q_3}$ is in counterclockwise direction of $\overrightarrow{Q_0Q_1}$. That is to say the point Q_3 is in counterclockwise direction of $\overrightarrow{Q_0Q_1}$. As above, it can be concluded: When $a \cdot b<0$, the nodes Q_2, Q_3 are located on both sides of $\overrightarrow{Q_0Q_1}$, as shown in Figure 4(a). When $a \cdot b>0$, the nodes Q_2, Q_3 are located on same side of $\overrightarrow{Q_0Q_1}$, as shown in Figure 4(b).

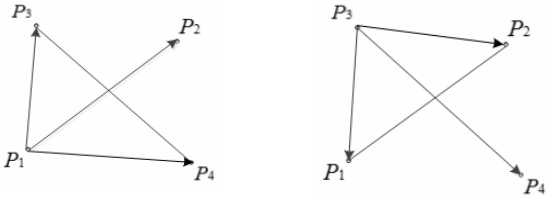
Judgment of segments intersection is described as follows. In Figure 5, there are four points in the plane, where P_1, P_2, P_3, P_4 form line segments P_1P_2 and P_3P_4. If P_1, P_2 are located on both sides of segment P_3P_4, and P_3, P_4 are also located on both sides of segment P_1P_2, these two line segments are said to intersect. The following formulas are defined as

$$\overrightarrow{P_3P_2}\times\overrightarrow{P_3P_4}=c_1\boldsymbol{k} \tag{8}$$

$$\overrightarrow{P_3P_2} \times \overrightarrow{P_3P_1} = c_2\boldsymbol{k} \quad (9)$$

$$\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_4} = c_3\boldsymbol{k} \quad (10)$$

$$\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3} = c_4\boldsymbol{k} \quad (11)$$



(a) P_3, P_4 are located on both sides of segment P_1P_2 (b) P_1, P_2 are located on both sides of segment P_3P_4

Figure 5. Determine intersection of segments

The c_1, c_2, c_3, c_4 in the formulas (8)-(11) are the cross product coefficients and the vector $\boldsymbol{k}$ is a unit vector. If $c_1 \cdot c_2<0$ and $c_3 \cdot c_4<0$, the line segment P_1P_2 intersects P_3P_4.

Judgment of obstacle is described as follows. The method of deleting superfluous inflection nodes is to connect the previous node and the next node of the current node. If there is no obstacle on the segment which connects the previous node with the next node, remove the current point. If there is an obstacle on the segment, it needs to ensure whether the segment intersects one of the diagonals of the obstacle. If yes, there is an obstacle. If not, it means that there is no obstacle.

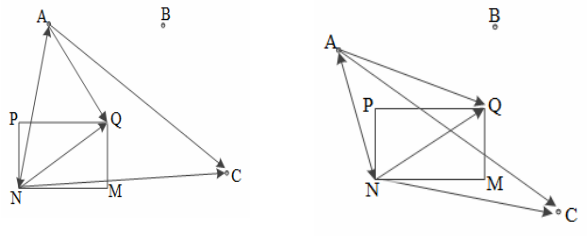
In Figure 6, define the node B(x_B, y_B) as the current node of the path. Node A(x_A, y_A) is the previous node of node B. Node C(x_C, y_C) is the next node of node B. Point P, Q, N, M are defined as four vertexes of the obstacles, and their coordinates are (x_P, y_P), (x_Q, y_Q), (x_N, y_N), (x_M, y_M) respectively.

$$\overrightarrow{AC} \times \overrightarrow{AQ} = [(x_A - x_C)(y_A - y_Q) - (y_A - y_c)(x_A - x_Q)]\boldsymbol{k} = w_1\boldsymbol{k} \quad (12)$$

$$\overrightarrow{AC} \times \overrightarrow{AN} = [(x_A - x_C)(y_A - y_N) - (y_A - y_c)(x_A - x_N)]\boldsymbol{k} = w_2\boldsymbol{k} \quad (13)$$

$$\overrightarrow{NQ} \times \overrightarrow{NA} = [(x_Q - x_N)(y_Q - y_A) - (y_Q - y_N)(x_Q - x_A)]\boldsymbol{k} = w_3\boldsymbol{k} \quad (14)$$

$$\overrightarrow{NQ} \times \overrightarrow{NC} = [(x_Q - x_N)(y_Q - y_C) - (y_Q - y_N)(x_Q - x_C)]\boldsymbol{k} = w_4\boldsymbol{k} \quad (15)$$



(a) There is no obstacle (b) There is an obstacle

Figure 6. Judgment of obstacle

The w_1, w_2, w_3, w_4 in the formulas (12)-(15) are the cross product coefficients, where the vector $\boldsymbol{k}$ is a unit vector. If $w_1 \cdot w_2<0$ and $w_3 \cdot w_4<0$, the segment AC intersects NQ, If not, the segment AC does not intersect NQ, as shown in Figure 6. Similarly, it can also determine whether line segment AC intersects PM. If line segment AC intersects one of the two diagonals, it means that there are obstacles in path AC, as shown in Figure 6(b). In this situation, the original path remains unchanged, still A-B-C. On the contrary, it means that there are no obstacles in path AC as shown in Figure 6(a). The node B which is a redundant inflection point is removed, and nodes A and C are connected, where the path changes from A-B-C to A-C.

The flowchart of improved A-star algorithm is shown in Figure 7. In the flowchart: P_i is the nodes of the initial path, where i is the label of the nodes of the initial path. **Step 1**: Obtain P_i, and let i=1. **Step 2**: Determine whether P_{i+1} is an ending node. If not, go to step 3. Otherwise, end the algorithm. **Step 3**: Determine whether the P_i, P_{i+1}, P_{i+2} are in the same line. If not, go to step 4. Otherwise, delete the node P_{i+1} and return to step 2. Since node $P_{i+1}$is deleted, the label of the nodes after P_i are decremented by one. **Step 4**: Determine whether there are obstacles between Pi and P_{i+2}. If not, delete the node P_{i+1} and return to step 2. Otherwise, go to step 5. **Step 5**: Let i=i+1 and return to step 2.

3.3 Determination of Rotation Angle and Direction

Define nodes A, B, C are three nodes on the path, as shown in Figure 8. The coordinates of nodes A, B, and C are (x_1, y_1), (x_2, y_2), (x_3, y_3), respectively. The angle between $\overrightarrow{AB}$ and x axis is α_1, and the angle between $\overrightarrow{BC}$ and x axis is α_2. Extend $\overrightarrow{AB}$ to coincide the vector $\overrightarrow{BD}$, where the angle between $\overrightarrow{BD}$ and $\overrightarrow{BC}$ axis is θ.

The vector can be expressed as:

$$\overrightarrow{AB} = \overrightarrow{BD} = (x_2 - x_1)\boldsymbol{i} + (y_2 - y_1)\boldsymbol{j} \quad (16)$$

$$\overrightarrow{BC} = (x_3 - x_2)\boldsymbol{i} + (y_3 - y_2)\boldsymbol{j} \quad (17)$$

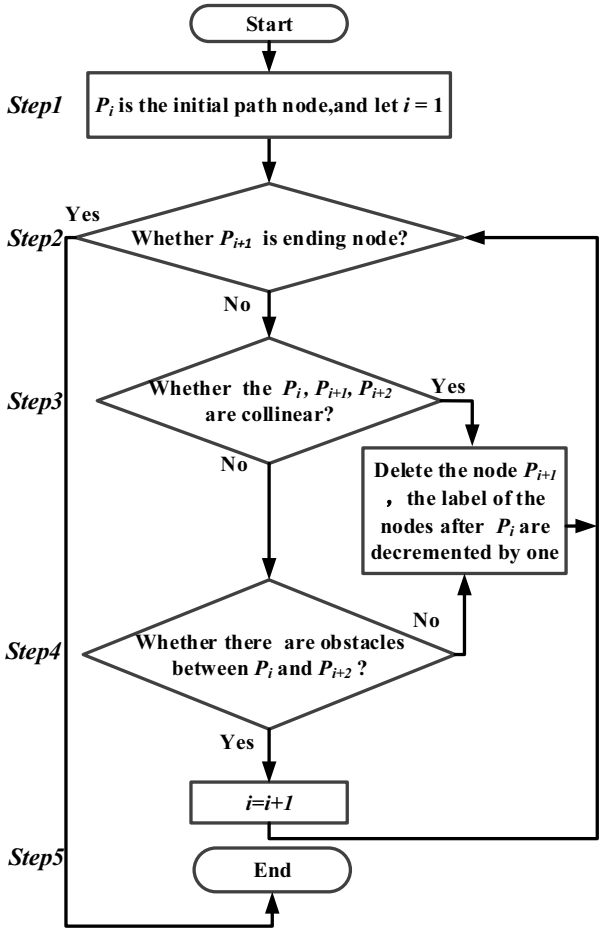


Figure 7. Flowchart of improved A-star algorithm

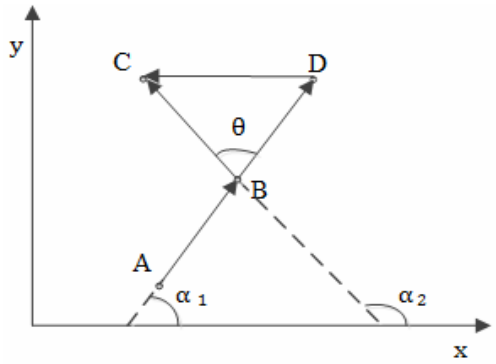


Figure 8. Vector of the path

$$\cos\theta = (\overrightarrow{BD}\cdot\overrightarrow{BC})/\sqrt{|\overrightarrow{BD}||\overrightarrow{BC}|}$$
$$= \frac{(x_3-x_2)(x_2-x_1)+(y_3-y_2)(y_2-y_1)}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}\sqrt{(x_3-x_2)^2+(y_3-y_2)^2}} \quad (18)$$

Let $\beta = \alpha_1 - \alpha_2$, at $\beta > 0$ && $y_3 \geq y_2$ && $y_2 \geq y_1$, the rotation angle of AGV is θ. Based on the above method, the rotation angle and turning direction of the AGV under various cases can be calculated, as shown in Table 1.

Table 1. The judgment of rotation angle and turning direction

Case	The value of β	Relatio-nship between y_3 and y_2	Relatio-nship between y_2 and y_1	The rotation angle and direction of AGV
1	β>0	$y_3 \geq y_2$	$y_2 \geq y_1$	AGV counterclockwise rotation is θ
2	β>0	$y_3 \geq y_2$	$y_2 < y_1$	AGV clockwise rotation is θ
3	β>0	$y_3 < y_2$	$y_2 \geq y_1$	AGV clockwise rotation is θ
4	β>0	$y_3 < y_2$	$y_2 < y_1$	AGV counterclockwise rotation is θ
5	B<0	$y_3 \geq y_2$	$y_2 \geq y_1$	AGV clockwise rotation is θ
6	B<0	$y_3 \geq y_2$	$y_2 < y_1$	AGV counterclockwise rotation is θ
7	B<0	$y_3 < y_2$	$y_2 \geq y_1$	AGV counterclockwise rotation is θ
8	B<0	$y_3 < y_2$	$y_2 < y_1$	AGV clockwise rotation is θ

The flowchart is shown in Figure 9:

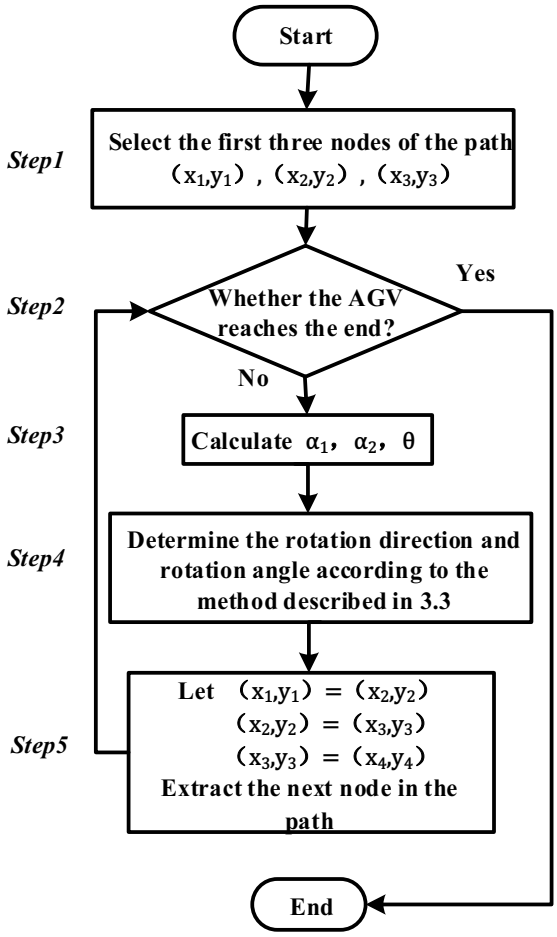


Figure 9. The adjustment of rotation angle at the inflection point

In conclusion, the AGV can adjust its own pose at the inflection point, and it is possible to walk out the path trajectory. Accordingly, AGV is able to break the limits of the grid to achieve full autonomy.

4 Performance Results and Analysis

4.1 Standardize the Obstacle

The AGV and the obstacle have a certain volume, and actual obstacles are irregular patterns. If the planned path is too close to obstacles, AGV is easy to collide with the obstacles along the planned path. In order to avoid this phenomenon, the obstacle is standardized, as shown in Figure 10.

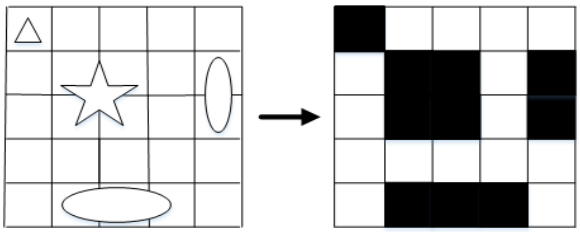


Figure 10. The handing of obstacles

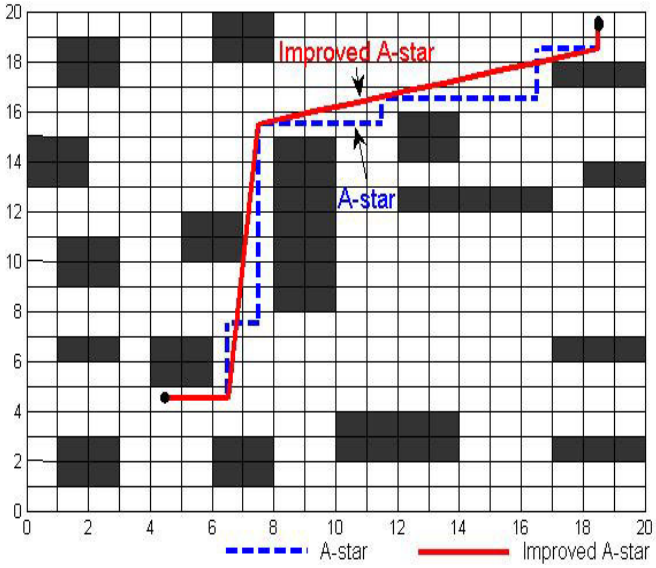
Standardizing the obstacle is to expand the obstacle area into a rectangle range, where the black grids represent the obstacles, and the white grids represent the area where the AGV can pass. After standardization of the obstacle, even if the planned path is close to the black grids, AGV is still safe to run along the planned path due to the safety distance maintained between AGV and obstacles.

4.2 Comparison between A-star Algorithm and Improved A-star Algorithm

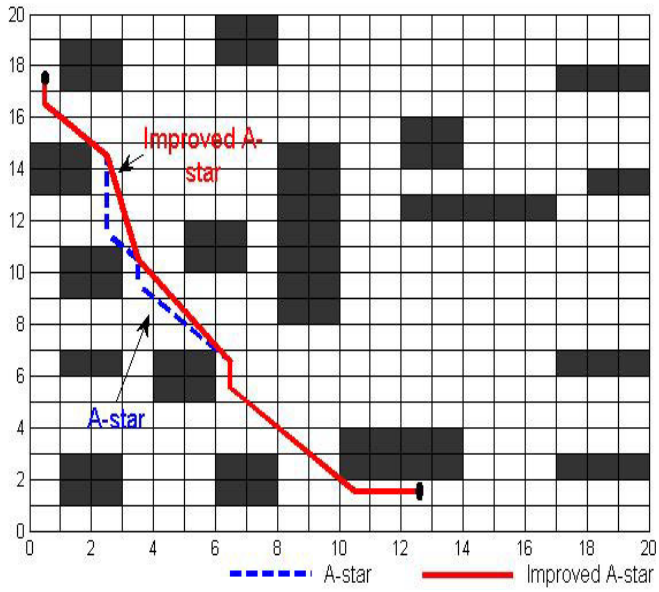
The comparison of performance efficiency for AGV in the same environment is shown in Table 2. The comparison of the path for A-star algorithm and improved A-star algorithm is shown in Figure 11.The AGV paths from the starting point (4.5, 4.5) to the ending point (18.5, 19.5) is shown in Figure 11(a), and the path from the starting point (12.5, 1.5) to the ending point (0.5, 17.5) is shown in Figure 11(b). The path length is calculated when the length of each grid is 1.

Table 2. The comparison of performance efficiency for AGV

Starting point Ending point	Algorithm	The grid number of the path nodes	The length of path	Turn times
(4.5, 4.5) (18.5, 19.5)	A-star	30	29.000	9
	Improved A-star	5	25.447	3
(12.5, 1.5) (0.5, 17.5)	A-star	19	22.142	8
	Improved A-star	8	21.132	6



(a)The path from the start point (12.5, 1.5) to the end point (0.5, 17.5)



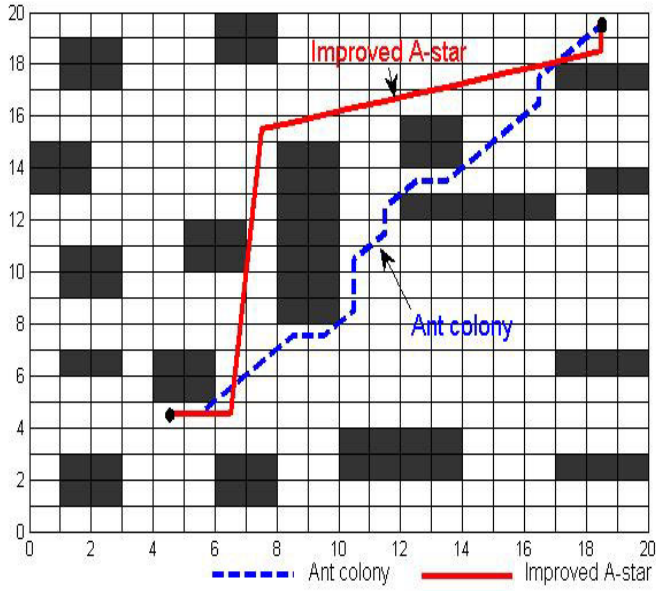
(b)The path from the start point (4.5, 4.5) to the end point (18.5, 19.5)

Figure 11. Comparison of the path for A-star algorithm and improved A-star algorithm

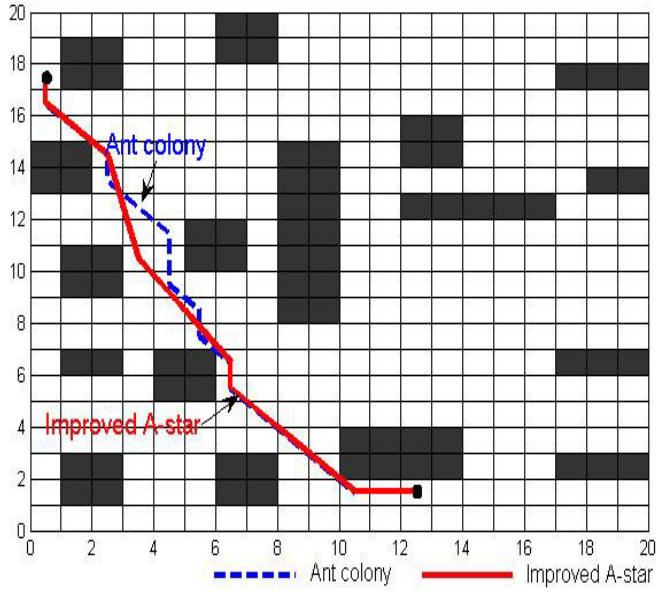
Table 1 shows that from the starting point is (4.5, 4.5) to the ending point is (18.5, 19.5), the path length of the improved A-star algorithm compared with the traditional A-star algorithm is shortened about 3.533 (12%), and the inflection points decreased by 6 times (66%). From the starting point (12.5, 1.5) to the ending point (0.5, 17.5), the path length of the improved A-star algorithm compared with the traditional A-star algorithm is shortened about 1.010 (4%), and the inflection points is decreased by 2 times (25%). As above, it is obvious that the improved A-star algorithm presents a better solution, i.e., shorter route and less inflection points.

4.3 Comparison between Ant Colony Algorithm and Improved A-star Algorithm

In order to further verify the effectiveness and feasibility of this method, the improved A-star algorithm is compared with the ant colony algorithm under the same environment, shown in Figure 12.



(a) The path from the start point (12.5, 1.5) to the end point (0.5, 17.5)



(b) The path from the start point (4.5, 4.5) to the end point (18.5, 19.5)

Figure 12. Comparison of the path for Ant colony algorithm and improved A-star algorithm

The AGV paths at the starting point (4.5, 4.5) and the ending point (18.5, 19.5) are shown in Figure 12(a). The paths at the starting point (12.5, 1.5) and the ending point (0.5, 17.5) are shown in Figure 12(b).

Table 3 shows that from the starting point (4.5, 4.5) to the ending point is (18.5, 19.5), the path length of the improved A-star algorithm compared with the Ant colony algorithm is increased about 1.897 (8%), and the inflection points is decreased by 8 times (70%). From the starting point (12.5, 1.5) to the ending point (0.5, 17.5), the path length of the improved A-star algorithm compared with the Ant colony algorithm is shortened about 1.010 (4%), and the inflection points is decreased by 4 times (40%). In conclusion, it is found that the length of the paths between Ant colony algorithm and improved A-star algorithm has only slight difference, but the path planned by improved A-star algorithm has less inflection points.

Table 3. The comparison of performance efficiency for AGV

Starting point Ending point	Algorithm	The grid number of the path nodes	The length of path	Turn times
(4.5, 4.5) (18.5, 19.5)	A-star	30	29.000	9
	Improved A-star	5	25.447	3
(12.5, 1.5) (0.5, 17.5)	A-star	19	22.142	8
	Improved A-star	8	21.132	6

4.3 The Analysis of the Time Complexity between Improved A-star Algorithm and Ant Colony Algorithm

The time complexity of an algorithm depends on the execution time, which is the sum of the frequency of all the statements in the algorithm. It is represented by $T(n)$, where n is the scale of the problem. When the problem is very complex, $T(n)$ becomes difficult to get an accurate solution. For this reason, the gradual time complexity is the order of magnitude of time complexity $T(n)$, which can be used to describe the trend of time complexity $T(n)$. Assume that the function $P(n)$ has the same order of magnitude as $T(n)$, that is

$$\lim_{n\to\infty}\frac{T(n)}{P(n)}=C(C\neq 0) \tag{19}$$

The order of magnitude of $P(n)$ can approximately represent by the time complexity $T(n)$. Referred to as

$$T(n)=\mathrm{O}(P(n)) \tag{20}$$

O( ) means to get the order of magnitude of the function $P(n)$. $P(n)$ usually equals the number of executions of the most executed statement in the algorithm. The time complexity can be sorted as follows:

$$\mathrm{O}(1)<\mathrm{O}(\log_2 n)<\mathrm{O}(n)<\mathrm{O}(n^2)<\ldots<\mathrm{O}(2^n) \tag{21}$$

Analysis of time complexity in the improved A-star algorithm. Most of statements executed in A-star algorithm are the sorting open list, and accordingly it needs to calculate the sub node $F(n)$ and sort in the open list when a node is planned. In this proposed model, the quick sort algorithm is used to sort in the open list, and its time complexity is $O(n\times\log_2 n)$ [21]. Each path node has 8 sub nodes so that it needs to sort the open list 8 times, and the time complexity is $O(8\times n\times\log_2 n)$. Assume that the A-star algorithm has gone through all the nodes, and its time complexity can be represented as

$$T(n_{\text{A-star}}) = O(8\times n^2\times\log_2 n) \quad \textbf{(22)}$$

The improved A-star algorithm can determine whether the nodes of the path are redundant turning points or redundant nodes after all the steps are executed. Assume that the A-star algorithm has gone through all the nodes, it needs to adjust n-2 times. Therefore, the time complexity can be represented as $O(8\times n^2\times(\log_2 n)+n-2)$. Since the order of magnitude only takes the highest power of the function, the time complexity of the improved A-star algorithm can approximate to the time complexity of the A-star algorithm.

Analysis of time complexity in the ant colony algorithm. Assume that N_c is the number of iterations, and m is the number of ants. When the next node is selected in ant colony algorithm, all the pheromones need to be updated except for the nodes that has been gone by. When the first node is determined, the pheromone of n nodes needs to be updated n times. Similarly, when the second node is chosen, the pheromone of n-1 nodes needs to be updated n-1 times, and so on. Assume that the ant has gone through all the nodes, the number of updating the pheromone statement is $(n\times(n+1))/2$, that is the frequency of the largest statement when one ant goes through the nodes by one iteration. When m ants go through the nodes by N_c iteration, the frequency of statement executed most times can be inferred as $[m\times N_c\times(n\times(n+1))/2]$. The time complexity of ant colony algorithm can be thus expressed as [22]

$$T(n_{\text{Ant}}) = O[m\times N_c\times n\times(n+1)/2] \quad \textbf{(23)}$$

The range of m is $\sqrt{n}-0.5n$ if let m=$\sqrt{n}$. The time complexity of ant colony algorithm can be approximated as

$$T(n_{\text{Ant}}) = O[N_c\times(n^{\frac{5}{2}}+n^{\frac{3}{2}})/2] \quad \textbf{(24)}$$

The time complexity curves of improved A-star algorithm versus ant colony algorithm (N_c takes 100) are shown in Figure 13, where the scale of the problem is the number of map nodes.

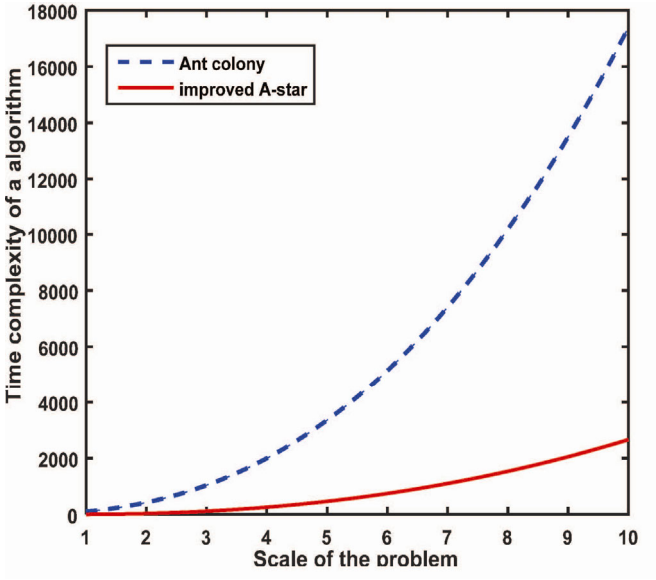


Figure 13. The time complexity curves of improved A-star algorithm versus ant colony algorithm

Based on the formulas (21) and (23), Figure 13 reveals that the time complexity of ant colony algorithm is relatively higher than that of improved A-star algorithm. Also, it is found that the gap between these two methods increases as the number of n increases. It means that the calculating time of ant colony algorithm is much longer than that of improved A-star algorithm especially in case of more complex problem.

Table 4 shows that from the starting point (4.5, 4.5) to the ending point (18.5, 19.5), the calculation time of Ant colony algorithm takes 7.847s. On the other hand, the calculation time of improved A-star algorithm takes only 0.549s, shortened about 93%. From the starting point (12.5, 1.5) to the ending point (0.5, 17.5), the calculation time of Ant colony algorithm takes 7.362s, and the calculation time of improved A-star algorithm takes only 0.186s, shortened about 97%. Obviously, the calculation time of improved A-star algorithm is conformed much shorter than Ant colony algorithm.

Table 4. The comparison of calculation time(s) for improved A-star algorithm and ant colony algorithm

Starting point Ending point	Algorithm	Calculation time (s)
(4.5, 4.5)	Ant colony	7.847
(18.5, 19.5)	Improved A-star	0.549
(12.5, 1.5)	Ant colony	7.362
(0.5, 17.5)	Improved A-star	0.186

5 Conclusions

In a warehouse environment, it is a crucial issue for AGV to choose an effective path to increase transport efficiency. In this paper, an improved A-star algorithm can eliminate superfluous inflection nodes in the path so that a better path can be determined. Simulation

show that compared with the A-star algorithm, the path length of the method is reduced by about 2%-25%, the number of inflection points reduced by about 20%-70%. Compared with the Ant colony algorithm, the number of inflection points reduced by about 30%-70%, the program operation time is reduced by about 95%. It is obvious that the proposed model is superior in shortening moving path length, reducing the energy consumption and avoiding frequent turning for AGV.

Acknowledgements

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions. This work has been developed for Research Starting Foundation for Ph.D. of Hebei university of Science and Technology and the Project on General Aviation Platform of Hebei university of Science and Technology.

References

[1] I. Draganjac, D. Miklić, Z. Kovačić, G. Vasiljević, S. Bogdan, Decentralized Control of Multi-AGV Systems in Autonomous Warehousing Applications, *IEEE Transactions on Automation Science and Engineering*, Vol. 13, No. 4, pp. 1433-1447, October, 2016.

[2] R. D. Yan, S. J. Dunnett, L. M. Jackson, Novel Methodology for Optimising the Design, Operation and Maintenance of a Multi-AGV System, *Reliability Engineering & System Safety*, Vol. 178, pp. 130-139, October, 2018.

[3] Y. Y. Gao, X.G. Ruan, H. J. Song, Path Planning Method for Mobile Robot Based on a Hybrid Learning Approach, *Control and Decision*, Vol. 27, No. 12, pp. 1822-1827, December, 2012.

[4] A. Wan, J. Xu, H. Chen, S. Zhang and K. Chen, Optimal Path Planning and Control of Assembly Robots for Hard-Measuring Easy-deformation Assemblies, *IEEE/ASME Transactions on Mechatronics*, Vol. 22, No. 4, pp. 1600-1609, August. 2017.

[5] J. R. Cheng, B. Y. Liu, K. Q. Cai, X. Y. Tang, B. Y. Zhang, ETC Intelligent Navigation Path Planning Method, *Journal of Internet Technology*, Vol. 19, No. 2, pp. 619-631, March. 2018.

[6] H. Yang, S.-Y Li, A Novel Recursive MOESP Subspace Identification Algorithm Based on Forgetting Factor, *Control Theory& Application*, Vol. 26, No. 1, pp. 69-72, January, 2009.

[7] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, L. Jurišica, Path Planning with Modified a Star Algorithm for a Mobile Robot, *Procedia Engineering*, Vol. 96, pp. 59-69, 2014.

[8] P. Wang, H.-T. Lin., T.-S. Wang, An Improved Ant Colony System Algorithm for Solving the IP Traceback Problem, *Information Sciences*, Vol. 326, No. 1, pp. 172-187, January, 2016.

[9] Z. Wang, X. Feng, H. D. Qin, H. M. Guo, G. J. Han, An AUV-Aided Routing Protocol Based on Dynamic Gateway Nodes for Underwater Wireless Sensor Networks, *Journal of Internet Technology*, Vol. 18, No. 2, pp. 333-343, Mar. 2017.

[10] T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, Heuristic Approaches in Robot Path Planning A Survey, *Robotics and Autonomous System*, Vol. 86, pp. 13-28, December, 2016.

[11] A. K. Guruji, H. Agarwal, D. K. Parsediya, Time-efficient A* Algorithm for Robot Path Planning, *Procedia Technology*, Vol. 23, pp. 144-149, 2016.

[12] J. H. Liu, J. G. Yang, H. P. Liu, Robot Global Path Planning Based on Ant Colony Optimization with Artificial Potential Field, *Transations of the Chinese Society for Agricultural Machinery*, Vol. 46, No. 9, pp. 18-27, September, 2015.

[13] R. Kala, A. Shukla, R. Tiwari, Robotic Path Planning in Static Environment Using Hierarchical Multi-neuron Heuristic Using Search and Probability Based Fitness, *Neurocomputing*, Vol. 74, No. 14-15, pp. 2314-2335, July, 2011.

[14] J. Ma, J. B. Wang, X. Zhang, Application of A-star Algorithm in Unmanned Vehicle Path Planning, *Computer Technology and Development*, Vol. 26, No. 11, pp. 153-156, November, 2016.

[15] Y. Chen, Y. Cai, J. Zheng, D. Thalmann, Accurate and Efficient Approximation of Clothoids Using Bézier Curves for Path Planning, *IEEE Transactions on Robotics*, Vol. 33, No. 5, pp. 1242-1247, October, 2017.

[16] Y. Fu, M. Ding, C. Zhou, H. Hu, Route Planning for Unmanned Aerial Vehicle (UAV) on the Sea Using Hybrid Differential Evolution and Quantum-Behaved Particle Swarm Optimization, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 43, No. 6, pp. 1451-1465, November, 2013.

[17] W. J. Lu, G. X. Zhang, S. Ferrari, An Information Potential Approach to Integrated Sensor Path Planning and Control, *IEEE Transactions on Robotics*, Vol. 30, No. 4, pp. 919-934, August, 2014.

[18] D. J. Wang, Indoor Mobile-robot Path Planning Based on an Improved A-star Algorithm, *Tsinghua Univ (Sci&Tech)*, Vol. 52, No. 8, pp. 1085-1089, August, 2012.

[19] J. H. Han, Y. H. Seo, Mobile Robot Path Planning with Surrounding Point Set and Path Improvement, *Applied Soft Computing*, Vol. 57, pp. 35-47, August, 2017.

[20] X. Zhao, Z. Wang, C. K. Huang, Y. W. Zhao, Mobile Robot Path Planning Based on an Improved A* Algorithm, *Robot*, Vol. 40, No. 6, pp. 1-8, November, 2018.

[21] M. Q. Liu, Research on Time Complexity of Sorting Algorithm, *Software Guide*, Vol. 11, No. 6, pp. 35-38, June, 2012.

[22] S. Perez-Carabaza, E. Besada-Portas, J. A. Lopez-Orozco, J. M. de la Cruz, Ant Colony Optimization for Multi-UAV Minimum Time Search in Uncertain Domains, *Applied Soft Computing*, Vol. 62, pp. 789-806, January, 2018.

Biographies

Yan Zhang received his M.S. and Ph.D. degrees from Hebei University of Technology, Tianjin, China, in communication and information system, in 2011 and in Electric Engineering, in 2015. He joined the School of Electrical Engineering at Hebei University of Science and Technology, Shijiazhuang, China, as a Lecture, His research interests include robot and AGV path planning, multiagent systems.

Ling-ling Li received her B.S. degree in Industrial Process Measurement and Control Instrument from Tianjin University, Tianjin, China, in 1989, M.S. degree in Control Theory and Control Engineering from Hebei University of Technology, Tianjin, China, in 2001, and Ph.D. degree in Electric Machines and Electric Apparatus from Hebei University of Technology, Tianjin, China, in 2004. Since 2006, she has been a professor with the School of Electrical Engineering, Hebei University of Technology, Tianjin, China. Her research interests include reliability of electrical apparatus, power system and new energy.

Hsiung-Cheng Lin graduated from National Taiwan Normal University for his bachelor degree in 1986, Taiwan. He received Master and Ph.D. degree from Swinburne University of Technology, Australia, in 1995 and 2002 respectively. He is currently a full professor in the Department of Electronic Engineering at National Chin-Yi University of Technology. His special fields of interest include power electronics, neural network, network supervisory system and adaptive filter design.

Zewen Ma was born in Shijiazhuang city of Hebei province in China, in 1993. He received the B.S. degree in electrical engineering from Hebei University of Science and Technology, Shijiazhuang, China, in 2016. Currently, he is a graduate student in the Department of Electrical Engineering, Hebei University of Science and Technology.

Jiang Zhao received her B.S. degree in Automation from Hebei University of Science and Technology, Shijiazhuang, China, in 1982, M.S. degree in Control Theory and Control Engineering from Tianjin University, Tianjin, China, in 1989, From February to June 2001, she was a advanced visiting professor with King's College London. Since 1996, she has been a professor with the School of Electrical Engineering, Hebei University of Science and Technology, Shijiazhuang, China. His research interests include process parameter detection and intelligent control technology.