Journal of Intelligent & Robotic Systems (2022) 104: 42
https://doi.org/10.1007/s10846-021-01561-5

REGULAR PAPER

Multi-AGV Tracking System Based on Global Vision and AprilTag in Smart Warehouse

Qifan Yang[1] · Yindong Lian[1] · Yanru Liu[1] · Wei Xie[1,2] · Yibin Yang[1]

Received: 20 July 2021 / Accepted: 14 December 2021 / Published online: 26 February 2022

Abstract

With the development of smart warehouses in Industry 4.0, scheduling a fleet of automated guided vehicles (AGVs) for transporting and sorting parcels has become a new development trend. In smart warehouses, AGVs receive paths from the multi-AGV scheduling system and independently sense the surrounding environment while sending poses as interactive information. This navigation method relies heavily on on-board sensors and significantly increases the information interactions within the system. Under this situation, a solution that locates multiple AGVs in global images of the warehouse by top cameras is expected to have a great effect. However, traditional tracking algorithms cannot output the heading angles required by the AGV navigation and their real-time performance and calculation accuracy cannot satisfy the tracking of large-scale AGVs. Therefore, this paper proposes a multi-AGV tracking system that integrates a multi-AGV scheduling system, AprilTag system, improved YOLOv5 with the oriented bounding box (OBB), extended Kalman filtering (EKF), and global vision to calculate the coordinates and heading angles of AGVs. Extensive experiments prove that in addition to less time complexity, the multi-AGV tracking system can efficiently track a fleet of AGVs with higher positioning accuracy than traditional navigation methods and other tracking algorithms based on various location patterns.

Keywords Multi-AGV system · Tracking · Global vision · AprilTag · YOLOv5

1 Introduction

In Industry 4.0 factories and warehouses, AGVs are connected as a collaborative community for efficiently sorting and transporting increasing parcels [1]. With the development of the Cyber-Physical System (CPS) and Information and Communications Technology (ICT) [2, 3], AGVs locally sense the surrounding environment by on-board sensors, such as camera, lidar, odometer, and IMU [4, 5], while being scheduled by a global control center with a sophisticated multi-AGV scheduling system to avoid AGV conflicts [6, 7]. Specifically, the navigation methods based on some fixed markers, such as QR codes and magnetic stripes, have extremely high location accuracy, which are widely used in warehouse and factory [8, 9]. After that, Ye et al. proposed an LIO-mapping approach, using IMU and odometer to assist lidar in order to locate AGVs precisely [10]. Similarly, Xiong et al. combined on-board camera and lidar to achieve the indoor location of several AGVs [11]. However, these methods require each AGV to be equipped with high-precision sensors, and a sufficient LAN bandwidth is also necessary for real-time information exchange between the control center and AGVs.

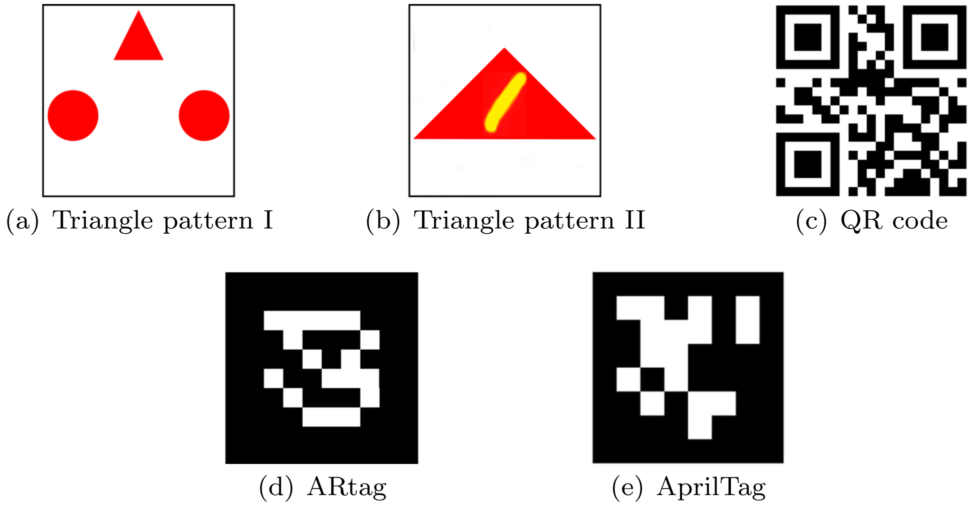
To address this issue, Dönmez et al. [12, 13] proposed a triangular servo control system, where the top camera obtained global images that were used to control an AGV with the triangular pattern I in Fig. 1(a). This global navigation method reduces the dependence of AGVs on on-board sensors. Meanwhile, AGVs no longer need to process the surrounding information in on-board controllers but distribute the calculations to servers in the system, which reduces the costs of AGVs while improving transportation efficiency. However, their tracking system has some limitations, and the tracking of multi-AGV is not considered, which makes it unsuitable for the warehouse and factory with a fleet of AGVs [14]. For this reason, Yang et al. [15] proposed a triangle tracking system based

✉ Wei Xie
weixie@scut.edu.cn

[1] School of Automation Science and Engineering, South China University of Technology, Guangzhou, China

[2] Guangdong Provincial Key Laboratory of Technique and Equipment for Macromolecular Advanced Manufacturing, South China University of Technology, Guangzhou, China

Fig. 1 Characteristic patterns used in robot systems



(a) Triangle pattern I

(b) Triangle pattern II

(c) QR code

(d) ARtag

(e) AprilTag

on the triangle pattern II, as shown in Fig. 1(b), which could distinguish multiple AGVs by recognizing digits on the top of AGVs. However, it cannot control AGVs to avoid obstacles autonomously because its recognition targets are limited to the AGVs that printed characteristic patterns. Therefore, it is not competent for complex and changeable factories and warehouses.

For navigating multiple AGVs by the global vision, the tracking algorithm is necessary to locate AGVs and output their real-time information, such as coordinates and heading angles. As a classical detector, the RCNN detects objects by the convolutional neural network (CNN) and support vector machine (SVM) [16]. It is used in the Deep SORT algorithm to obtain bounding boxes of objects. After that, the Kalman filter, cascading matching, and IOU matching are combined for tracking objects [17, 18]. Compared with RCNN, the YOLO object detection method converts the target detection into a regression problem and takes less calculation time [19–21]. In order to improve the real-time performance of the tracking algorithm, combining YOLOv3 and Deep SORT for multi-object tracking is a feasible solution [22].

However, there is no excessive requirement for the calculation accuracy of object rotation angles in these traditional multi-object tracking algorithms. Their tracking results are usually limited to accurate bounding boxes and object classes [23, 24]. For this reason, the main challenge for the tracking system is to calculate AGV heading angles while tracking. Meanwhile, a high recognition rate is also necessary to avoid conflicts due to misrecognition. In the global image, similar AGVs cannot be accurately located and distinguished by their appearances. For tracking multiple AGVs accurately, characteristic patterns used to locate AGVs and store their information are necessary. Usual characteristic patterns include QR code, ARtag, and AprilTag, etc [25, 26], as shown in Fig. 1(c), (d), and (e). With the strong storage capacity and high recognition rate, the QR code is popular in many fields. Besides, PR2 robots use ARtags to locate objects and then grab them through robot arms to transport cargos in ROS operating system [27]. As a visual fiducial system with a low error rate, short calculation time, and high detection rate, the AprilTag is also used in many robot systems [25, 28, 29]. Therefore, we print different AprilTags on various AGVs to locate them and record their information. Another critical challenge of a multi-AGV tracking system is that all AGVs should be controlled in time. In large warehouses and factories, the real-time control of hundreds of AGVs is a complex problem, which puts forward high requirements on the real-time performance of algorithms.

The main contributions of this paper are as follows: (1) We propose a multi-AGV tracking system based on the global vision and combine it with a multi-AGV scheduling system, so there is no need for cascading matching like mainstream tracking algorithms. In addition, some recognition processes are omitted and the images needed to track AGVs are cut hierarchically, resulting in the significant improvement of the real-time performance. (2) The improved YOLOv5 with oriented bounding box (OBB) and AprilTag system are integrated into the tracking system for distinguishing multiple AGVs and outputting their locations and angles. To the best of our knowledge, the visual tracking system for large-scale AGVs has not been discussed. (3) Compared with other traditional navigation methods, the tracking method based on global vision proposed in this paper offers a trade-off solution between flexibility and accuracy since there is no need to install fixed markers including QR codes and magnetic stripes for navigation but it has higher location accuracy compared to other free navigation methods, such as lidar navigation. Simulation experiments prove that the tracking system proposed in this paper can track multiple AGVs in real-time. Compared with other algorithms, the tracking system has lower computational complexity and higher location accuracy.

2 Multi-AGV Tracking System Applied to the Warehouse

2.1 Warehouse System Based on Global Vision

Figure 2 illustrates the warehouse system based on the global vision, where the control center is built on the robot operating system (ROS), consisting of a multi-AGV scheduling system and a multi-AGV tracking system. The top camera records global images for tracking and recognizing multiple AGVs in the warehouse system. When parcels enter the warehouse through logistics and conveyor belts, the scheduling system registers their information. Meanwhile, the tracking system calculates the coordinates of each AGV in real-time. After an AGV carries a parcel from the conveyor belt, the scheduling system plans the path according to the task information and receives the AGV information from the tracking system. Next, the control center sends instructions to multiple AGVs through ROS distributed communication based on WiFi so that AGVs deliver parcels to their designated areas. Finally, parcels are thrown into pipelines for centralized storage on the next floor. Particularly, the blue rectangle is the pipeline edge, which can also be regarded as an obstacle. As for dashed lines, they represent the field of view of the camera on the top, AGVs can be tracked in this range.

The multi-AGV scheduling system is mainly composed of a hierarchical planning algorithm that includes global planning and local planning [15]. The hierarchical planning divides the entire warehouse map into several areas according to different functions, which are represented by A_i in Fig. 3(a). In addition, the black rectangles are pipelines used as the AGV destinations that are closed to traffic while white grids are accessible. In terms of corridors, they are all the two-side way and connect different areas, so that a larger map can be extended conveniently. After that, the sorting task can be regarded as controlling the AGV from one area to another target area. Every area includes an intersection and many corridors, namely Cor_i as shown in Fig. 3(b), and the basic unit of area is the grid. For preventing collisions, the grid is set slightly larger than an AGV, and it can only accommodate an AGV during the scheduling process. The global planning is used to obtain the area set of the AGV from the current area to the task target area. When an AGV enters a new area according to the global path, a specific grid path is planned by the local planning to make AGV leave this area without collisions. The hierarchical planning algorithm simplifies the complex multi-object scheduling problem and makes a trade-off between real-time performance and efficiency.

2.2 Characteristic Pattern Selection

In the warehouse system based on the global vision, each AGV only occupies a small part of the global image. Besides, AGVs usually have the same appearance, so it is

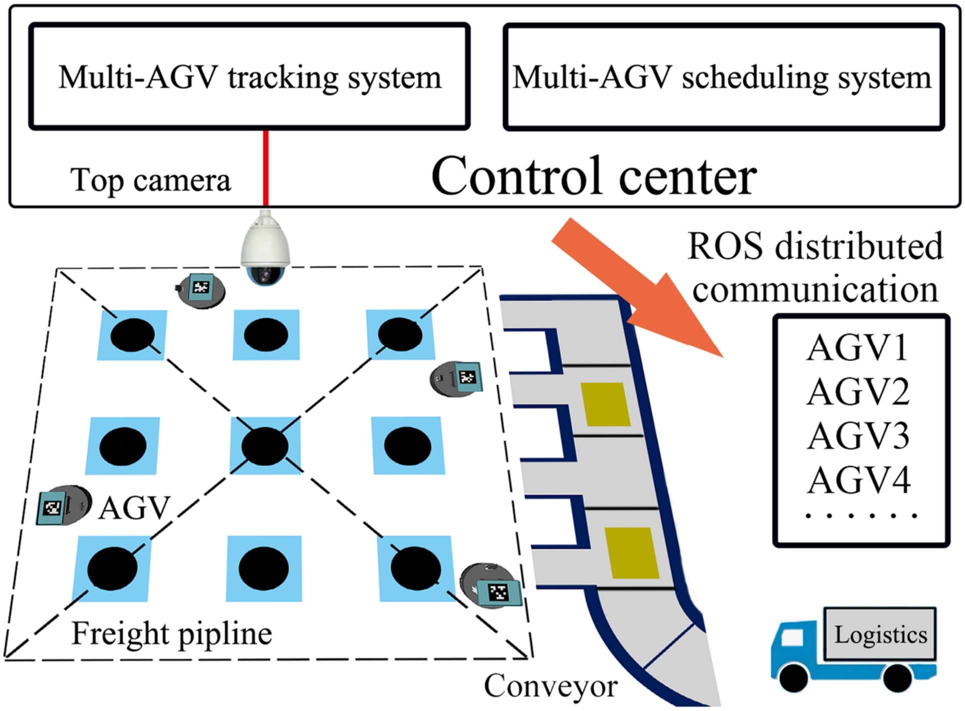
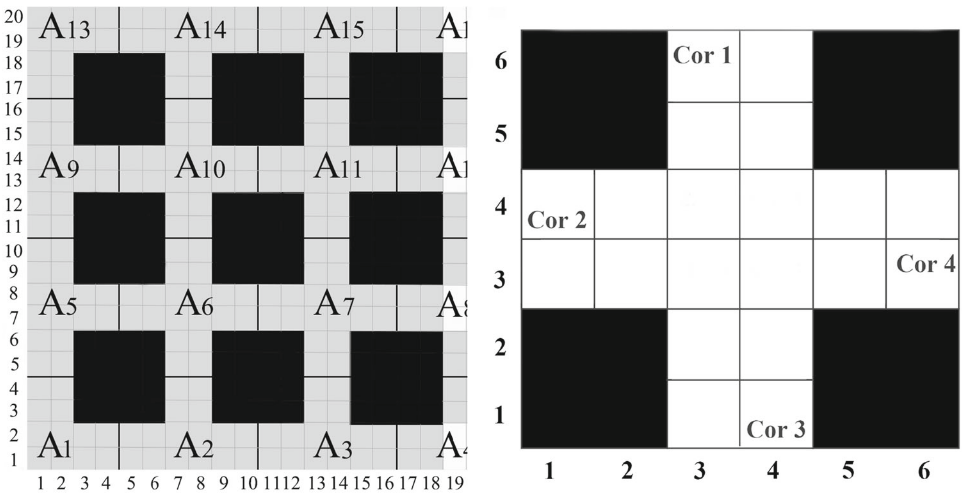


Fig. 2 Warehouse system based on the global vision

Fig. 3 Topological structure of the warehouse system



(a) Global map (b) Local map

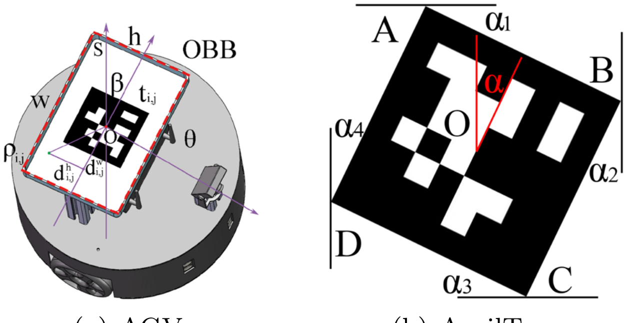
challenging to distinguish a large number of AGVs only by their external characteristics.

For tracking multiple AGVs through the global vision, it is necessary to place characteristic patterns on AGV tops to obtain their accurate locations and distinguish each other. It is worth noting that the characteristic pattern needs to satisfy the following requirements:

- It is asymmetric and can be used to calculate the AGV heading angle.
- For improving the system robustness, it should be fault-tolerant to prevent AGVs from disappearing due to decoding errors.
- It can store some necessary information, including AGV number and task information.
- As the cameras are installed on the top of the warehouse, the characteristic pattern should be accurately recognized even at low resolution.

Different polygons are usually used as characteristic patterns for robot navigation [12, 15, 30]. But they often include less information and cannot meet the requirements for distinguishing a large number of AGVs. Meanwhile, polygons are relatively simple, without the ability to detect and correct errors, and their false positive rates are high, so other objects are easily misidentified as characteristic patterns. It is worth noting that the most characteristic patterns used for navigation are geometrical shapes, which are convenient for storing information and calculating target poses. However, circular patterns are rarely selected since they cannot be used to calculate the AGV heading angle. Additionally, characteristic codes significantly reduce the false positive rates through encoding, and many types of characteristic codes are widely used in shopping malls and factories. Traditional characteristic codes focus on decoding and locating while ignoring the calculation of rotation angles or only adopting a rough estimation based on a few characteristic points to achieve the recognition. Such characteristic codes are undoubtedly not suitable for tracking a fleet of AGVs.

Consequently, we choose AprilTags as characteristic patterns on top trays of AGVs, as shown in Fig. 4. Specifically, the midpoints of AprilTag and AGV are the same in terms of x-coordination and y-coordination. Furthermore, the AprilTag points to the direction of the front of the AGV. In this direction, its decoding result is in family codes that do not require additional rotations. Compared with QR codes, AprilTags take up more pixels per bit and are more suitable for low-resolution scenes. Specifically, they have different kinds of tag families such as 16h5, 25h9, 36h11 [28]. Taking 36h11 as an example, each code in the tag family contains 36 information bits, and the minimum Hamming distance between any two codes is 11. This type of code can correct and detect 5-bit errors. At the same time, it is specific after rotating to signify that the Hamming distances between the rotated code and others are greater than 11. Besides, the AprilTag is located by the



(a) AGV (b) AprilTag

Fig. 4 Characteristic pattern on AGV

black rectangular frame on the periphery, which has better robustness at low resolution than QR code using three finder patterns to determine the position and rotation angle.

Particularly, the decoding result of the AprilTag will change after rotating 90°. Take the code numbered 1 (d5d628584) as an example. When it is rotated three times, its decoding results are 2573b3403, 21a146bab, and c02cdcea4, respectively. We can find that their Hamming distances from the code numbered 1 are 18, 24, and 18. Furthermore, their distances from other codes in the tag family are also greater than 11. Therefore, we can calculate the rotation angle of the AprilTag while decoding.

2.3 Improved Multi-AGV Detection Algorithm

When using the global image to track AGVs, it is time-consuming to traverse all the AprilTags in the current image to obtain the information of each AGV, and this method is easy to be interfered with by external noise and lead to misidentification. Because of the excellent performance of deep learning algorithms in vehicle and pedestrian tracking [31, 32], we use the multi-object tracking algorithm based on deep learning to determine the AGV pose and then decode the AprilTag on the AGV top to achieve a better tracking effect.

However, some traditional multi-object detection and tracking algorithms ignore the heading angles of objects which are necessary for AGV navigation. Under this condition, we improve the traditional object detection algorithm YOLOv5 by using an oriented bounding box (OBB) which not only outputs the object position but also calculates the object heading angle [33, 34].

As the core of calculating the heading angle is the loss function, we choose the PIoU (Pixels-IoU) loss to solve the problem. The PIoU loss is a pixel-by-pixel manner, which has good performance on objects with a high aspect ratio and complex background. For utilizing the PIoU loss, we define

$$F(\rho_{i,j}|r) = K(d^w_{i,j}, w)K(d^h_{i,j}, h) \tag{1}$$

where K is an improved sigmoid kernel function to make $F(\rho_{i,j}|r) \sim 1$ when the pixel is in the OBB. For a pixel $\rho_{i,j}$ and a red OBB frame in Fig. 4(a), the distance $d^h_{i,j}$ from the pixel to the center line $t_{i,j}$ and the distance $d^w_{i,j}$ from the intersection with the center line to the center point O are used to determine whether the pixel is in the OBB.

Also, we define

$$S_{r\cap r'} \approx \sum_{\rho_{i,j}\in R_{r,r'}} F(\rho_{i,j}|r)F(\rho_{i,j}|r') \tag{2}$$

$$S_{r\cup r'} \approx \sum_{\rho_{i,j}\in R_{r,r'}} F(\rho_{i,j}|r)+F(\rho_{i,j}|r')-F(\rho_{i,j}|r)F(\rho_{i,j}|r') \tag{3}$$

where $S_{r\cap r'}$ is the intersection area of r and r', and $S_{r\cup r'}$ is the union area of r and r'. When the predicted box r is based on a positive anchor and the ground-truth box r' is a matching ground-truth box, (r, r') is a positive pair. $R_{r,r'}$ is the smallest horizontal box covering r and r', Therefore the loss function of PIoU is defined as:

$$L_{PIoU} = \frac{-\sum_{(r,r')\in T}}{|T|}[lnS_{r\cap r'} - lnS_{r\cup r'}]. \tag{4}$$

where T refers to all positive pairs. After that, the rotation angle of OBB $\beta_k \in [-90°, 90°]$ is obtained by the improved YOLOv5 as shown in Fig. 4(a) and the AGV clockwise rotation is positive, starting with plumb line OS. Particularly, as the AprilTag has thousands of different types, it is difficult to train a model that outputs the full angle $[0°, 360°]$ of all AGVs with different AprilTags accurately. So the rectangular trays are used to train the improved YOLOv5, which has a high positioning effect but thus limits the range of output angles as $[-90°, 90°]$. After that, the AGV heading angle $\theta^O_k \in [0°, 360°]$ from the improved YOLOv5 is calculated by

$$\theta^O_k = \begin{cases} 90° + \beta_k + 180° \cdot s_k, & \theta_{k-1} \leq 180°. \\ 270° + \beta_k - 180° \cdot s_k, & \theta_{k-1} > 180°. \end{cases} \tag{5}$$

$$s_k = \begin{cases} 1, & \omega_{k-1} \cdot \beta_{k-1} > 0, \omega_{k-1} \cdot \beta_k < 0. \\ 0, & otherwise. \end{cases} \tag{6}$$

The angular velocity at the last time is defined as ω_{k-1} and s_k is the flag to determine the heading angle. Then, the exact AGV heading angle is determined according to the rectangular tray with unknown orientation. In Fig. 4(a), the AGV pixel coordinate in the global image is $O(u, v)$. Due to the influence of intrinsic parameters, we need to calibrate the camera for obtaining AGV coordinates in the world coordinate system [35, 36]. Meanwhile, as the AGV has a certain height, it is also necessary to map the coordinates of the characteristic pattern on the AGV to the ground. The specific mapping function used to obtain the actual AGV pose can be formulated as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{H-h}{H} \cdot M \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{7}$$

where (x, y) is the coordinate of the AGV bottom in the image coordinate, H is the height of the top camera, h is the AGV height, and M is a mapping matrix determined by camera calibration parameters to remove image distortions. Then, the actual coordinate (X, Y) is defined as:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \frac{2 \cdot \tan(0.5 \cdot FOV) \cdot H}{Re} \begin{bmatrix} x \\ y \end{bmatrix} \tag{8}$$

where Re is the global image resolution, and FOV is the field of view of top cameras. After that, the AGV heading angle is calculated for determining the concrete pose of each AGV.

After detecting all AprilTags in the warehouse through the tracking system, we can obtain the number and location information of each AGV by decoding. Since the corners of the characteristic pattern are easy to blur, we extract contours of characteristic patterns and fit each edge of the square using the least square method [37]. The angle α in Fig. 4(b) is defined as the weighted average of the rotation angle of each edge α_i.

$$\alpha_i = arctan\left(\frac{n\sum_{j=1}^{n} x_j y_j - \sum_{j=1}^{n} x_j \sum_{j=1}^{n} y_j}{n\sum_{j=1}^{n} x_j x_j - \sum_{j=1}^{n} x_j \sum_{j=1}^{n} x_j}\right) \tag{9}$$

$$\alpha = \sum_{i=1}^{4} \frac{k_i \alpha_i}{\sum_{j=1}^{4} k_j} \tag{10}$$

Particularly,

$$k_i = \frac{\sum_{j=1}^{n} y_j^2}{\sum_{j=1}^{n} y_j^2 - \sum_{j=1}^{n} (y_j - Y_j)^2} \tag{11}$$

where n is the number of point sets (x_j, y_j) on one edge of the characteristic pattern. We denote k_i as the weight of edge i, which is set according to the discrete degree of point set of edge i.

As shown in Fig. 4(b), we can obtain any positive characteristic pattern by rotating the current pattern α degrees. However, the AGV direction is unknown, so its heading angle cannot be determined. Under this condition, we rotate the characteristic pattern three times, each time by 90°. If the decoding result is consistent with the code in the tag family, we add the angles of these two parts to obtain the AGV heading angle θ^A from the AprilTag.

$$\theta_k^A = 90° \cdot r + (45° - \alpha) \tag{12}$$

where r belongs to $0 \sim 3$, which means that the decoding result is matched in the tag family after rotating r times. Finally, the accurate AGV heading angle is formulated as follows:

$$\theta_k = \begin{cases} 0.5 \cdot (\theta_k^O + \theta_k^A), & if\ AprilTag\ is\ detected. \\ \theta_k^O, & if\ AprilTag\ is\ not\ detected. \end{cases} \tag{13}$$

2.4 Implementation of Multi-AGV Tracking System

We choose a stable and rapid detection model, YOLOv5, to detect AGVs in the global image due to the relatively single environment and apparent AGV appearances. It is worth mentioning that the processing speed of YOLOv5 is faster than previous versions, and more network models can be replaced as needed, which increases the flexibility of YOLOv5. For example, if the system requires high recognition accuracy, a more sophisticated neural network is recommended. Compared with directly detecting the rectangular area containing AprilTags in the global image to identify AGVs, YOLOv5 has better robustness and adapts to different lighting conditions. Furthermore, it is also possible to detect moving workers or other common obstacles in the warehouse so that AGVs can autonomously avoid obstacles in time.

Since there is a multi-AGV scheduling system in the warehousing system, the trajectory of each AGV is known. The purpose of the tracking system is to determine the poses of all AGVs and send them to the scheduling system for correcting deviations. Through this closed-loop control system, all AGVs can move accurately according to the plans made by the scheduling system. The process of the multi-AGV tracking system is illustrated in Fig. 5.

Initially, the warehouse is divided into many areas after partition. Then, the global image from the top camera is processed by the improved YOLOv5 to obtain the OBBs including all AGVs. If there is an AprilTag in the OBB and the distance error between the real AGV location and predicted location is slighter than the threshold, the AGV pose is obtained by equation 5 without decoding. Otherwise, if there is a parcel covering the AprilTag, the AGV pose will be updated according to its OBB. As for AGVs that do not follow the given trajectory, their AprilTags are decoded for avoiding misrecognition. However, when some external interference blurs the AprilTag, it is necessary to compare it with all the AprilTags of AGVs recorded in the same area with the abnormal AGV. In other words, only these AGVs will appear near the AGV with this blurry AprilTag.

After obtaining all AGV information, extended Kalman filtering (EKF) is used for predicting and locating multiple AGVs in subsequent frames. Moreover, the processing object of the improved YOLOv5 is changed to local images based on the predicted AGV location, reducing the amount of calculation. Specifically, the first step is estimating the AGV parameters:

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + v_{k-1}\Delta t cos\theta_{k-1} \\ y_{k-1} + v_{k-1}\Delta t sin\theta_{k-1} \\ \theta_{k-1} + \omega_{k-1}\Delta t \\ v_{k-1} \\ \omega_{k-1} \end{bmatrix} \tag{14}$$

where X_k is discrete, including coordinates (x_k, y_k), heading angle θ_k, velocity v_k, and angular velocity ω_k, with sampling time Δt. In the first frame, we detect AGVs through the improved YOLOv5 with OBB and then determine the initial pose X_1 and the number of each AGV by recognizing its AprilTag.

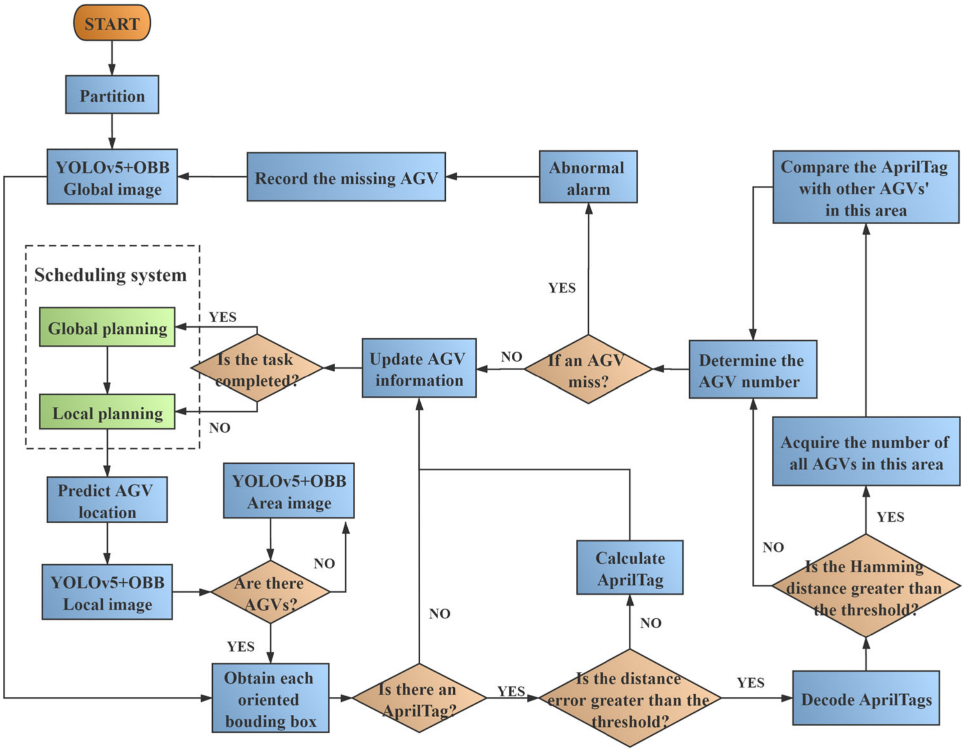


Fig. 5 Flowchart of the multi-AGV tracking system

After that, we predict the AGV location in the next frame.

$$X_k^- = f(X_{k-1}^+) \tag{15}$$

$$\Sigma_k^- = F_k \Sigma_k^+ F_k^T + Q \tag{16}$$

We use the predicted AGV pose X_k^- as the center to intercept a rectangular, detect the AprilTag, and then use its boundary to calculate the AGV pose Z_k. Particularly, F is the Jacobian of the nonlinear system f and Σ_k^- represents the prior covariance matrices. In the simulation environment, since AGV can operate strictly according to instructions, Q is the associated covariance of the dynamic system noise which is defined as diag$\left\{1^{-6} I_3, 0_{2\times 2}\right\}$.

To reduce computation time, the decoding process is ignored and the AGV number inherits the value of the previous frame.

Then, the EKF model is updated by:

$$K_k = \Sigma_k^- H_k^T (H_k \Sigma_k^- H_k^T + R)^{-1} \tag{17}$$

$$X_k^+ = X_k^- + K_k(Z_k - h(X_k^-)) \tag{18}$$

$$\Sigma_k^+ = (I_5 - K_k H_k)\Sigma_k^- \tag{19}$$

where K_k is the Kalman gain, H_k is the Jacobean of the linear measurement function h, I_5 is the 5-dimensional identity matrix and R is the associated covariance of the dynamic system noise which is set to $1^{-3} \cdot I_3$, which represents the degree of confidence in terms of location calculated by the tracking system. It is worth noting that the smaller Q and R are, the less our system trust the predicted value and measurement value during the locating process. After that, Z_k is the measurement value including AGV coordinates and heading angle. Finally, posterior pose X_k^+ and its covariance matrix Σ_k^+ are obtained, respectively.

After predicting AGV poses, we construct square bounding boxes that are 1.2m (about twice the AGV length) and centered on the predicted AGV coordinates. They are sent into YOLOv5 as local images for obtaining accurate poses and AGV numbers, which will be recorded separately according to the area where the AGV is located. During the AGV movement according to the multi-AGV scheduling system, AGV poses are obtained by the AprilTags or oriented bounding boxes directly, and then they are sent to the scheduling system to correct scheduling errors. Compared with traditional DBT (Detection-Based-Tracking) methods, such as Deep SORT, the cascading matching is omitted, and the real-time performance of the tracking system is improved. It is worth noting that if some AGVs are missing, which cannot be detected through bounding boxes predicted by EKF, the tracking system marks these AGVs, uses the improved YOLOv5 again to process an area image or the global image, and re-tracks all AGVs. Specifically, the local image from the EKF is the smallest, the area image is larger, and the global image is the largest. Using hierarchical input images to track AGVs reduces unnecessary calculations in the tracking system. If there are obstacles or warehouse staff, the detector can also recognize them and send this event to the scheduling center for emergency processing. Unlike other navigation methods that require fixed markers and pre-built maps, the tracking method based on global vision can easily locate AGVs far from established paths as long as the top camera covers the current environment. After that, the A* path planning algorithm is used to control the missing AGV back to the specified path. The calculation process of the multi-AGV tracking system is as follows.

Firstly, the multi-AGV tracking system receives global images G_k and AGV paths planned by the scheduling system. S_a represents area sets of all AGVs, which connect task start areas and target areas. S_g stores grid sets, which are the AGV positions in the next few periods and are determined when the AGV enters a new area. In the first frame, namely $k = 1$, the improved YOLOv5 is used for determining oriented bounding boxes $OBBs$ of all AGVs, which are rough estimates of AGV poses (lines 1 through 3 of Algorithm 1). In other frames, bounding boxes are predicted by the EKF and previous AGV poses, including coordinates (x_i, y_i), heading angle θ_i, linear velocity v_i, and angular velocity ω_i (line 5 of Algorithm 1). After that, the oriented bounding boxes are obtained by using images of different sizes according to the emergency level $flag_e$. When the system is running normally, $flag_e = 0$ and the bounding boxes (BBs) predicted by the EKF are sent to the improved YOLOv5. On the contrary, the emergency level will increase and a wider area is chosen to search for missing AGVs (lines 11 through 18 of Algorithm 1).

When detecting an AGV in the OBB, the tracking system determines its number by comparing the coordinates of the oriented bounding box with all grid paths Sg designed by the scheduling system, and the AGV number n with the smallest gap is selected (lines 20 through 21 of Algorithm 1). If there is an AprilTag in the OBB and the distance error between the real location $OBBs[j]$ and predicted location Sg_n is greater than a threshold μ_L defined as the AGV length, the AprilTag should be decoded for determining its actual number (lines 22 through 26 of Algorithm 1). Otherwise, an AGV without the AprilTag means there is a parcel on its tray covering the AprilTag, so its poses are updated by the OBB directly (line 36 of Algorithm 1). Particularly, AGV_n has the risk of misrecognition when its Hamming distance D_n is greater than a designed threshold μ_T that is defined as 8 and is slightly smaller than the minimum Hamming distance between any two AprilTags. So this algorithm retrieves the number of AGVs, namely N_A, registered in the area Sa_n where AGV_n are located, and the distances between the decoding result Rd_n of AGV numbered n and decoding results Rd_u of other AGVs are calculated. Finally, the AGV number is determined as n and all AGV poses (x_n, y_n, and θ_n) are calculated by the AprilTag contour (lines 27 through 34 of Algorithm 1).

Algorithm 1 Multi-AGV tracking algorithm.

```
INPUT: The global image $G_k$ and parameters of AGVs
    including the set of areas $Sa$ and the set of grids $Sg$.
OUTPUT: Coordinates $(x, y)$ and heading angles $\theta$ of all
    AGVs
 1: if $k == 1$ then
 2:     $OBBs$ =YOLOv5($G_k$)
 3: else
 4:     for $i = 0$ to $N$ do
 5:         $BBs[i]$ =EKFpredict($x_i, y_i, \theta_i, v_i, \omega_i$)
 6:     end for
 7: end if
 8: $j = 0, flag_e = 0$
 9: while $j < N$ do
10:     if $k > 1$ then
11:         switch ($flag_e$)
12:         case 0:
13:         $OBBs[j]$ =YOLOv5($BBs[j]$)
14:         case 1:
15:         $OBBs$ =YOLOv5($Sa_j$)
16:         case 2:
17:         $OBBs$ =YOLOv5($G_k$)
18:         end switch
19:     end if
20:     if $OBBs[j] > 0$ then
21:         $n$ =match($OBB[j], Sg$)
22:         $AprilTag[j]$ =findContours($OBBs[j]$)
23:         if $AprilTag[j] > 0$ then
24:             if distance($OBBs[j], Sg_n$) $> \mu_L$ then
25:                 $n, Rd_n, D_n$ =decode($AprilTag[j]$)
26:             end if
27:             if $D_n > \mu_T$ then
28:                 $N_A$ =find($Sa_n$)
29:                 for $u = 0$ to $N_A$ do
30:                     $Dis[u] = (Rd_n|Rd_u)$
31:                 end for
32:                 $n$ =Min($Dis$)
33:             end if
34:             $x_n, y_n, \theta_n$ =calculate($AprilTag[j]$)
35:         else
36:             $x_n, y_n, \theta_n = OBBs[j]$
37:         end if
38:         $X_n, Y_n$ =find($Sg_n$)
39:         $v_n, \omega_n$ =control($X_n, Y_n, x_n, y_n, \theta_n$)
40:         $j = j + 1, flag_e = 0$
41:     else
42:         if $flag_e == 0$ then
43:             $flag_e = 1$
44:         else
45:             $flag_e = 2$
46:         end if
47:     end if
48: end while
```

To ensure stable operation, the target coordinates of AGV_n in the next frame are determined by its grid path Sg_n. They are combined with the actual coordinates and heading angle as inputs to calculate the linear velocity and angular velocity, which are used in the next tracking process. (lines 38 through 40 of Algorithm 1). Finally, if an AGV appears outside the bounding boxes from the EKF, the emergency level will change (lines 42 through 46 of Algorithm 1). Then, the global image will be sent to the improved YOLOv5 for re-tracking all AGVs ($flga_e$=2) after checking a smaller image of the area recorded by this AGV recently ($flga_e$=1). As there is a high probability that the missing AGV will still appear in the same area located in the previous frame, it is unnecessary to examine the global image first. So we choose hierarchical input images according to the operation of the system (lines 11 through 18 of Algorithm 1).

In addition to the fault tolerance of AprilTag, we combine decoding methods with the hierarchical planning algorithm to improve the robustness. For example, if there are 4 AGVs in the same area and their numbers are 1,2,3, and 4, their information will be stored in the area when they enter this area. According to the coding rules, their decoding results are d5d628584, d97f18b49,dd280910e, and e479e9c98, respectively. When detecting AGV numbered 1, the minimum Hamming distances between d5d628584 and the other three codes are 16, 14, and 17. For this reason, the error detection and correction capabilities in this area are increased from 5, 5 to 6, 6. It means that this AGV can be decoded correctly even if the AprilTag on AGV misses 6 bits of information, which is also suitable for the detection of other AGVs in this area. Moreover, if the AGV numbered 1 has a 7-bit error, its number is most likely to be misidentified as 3. However, the error must be at the intersection of 1 and 3, which is a small probability. So the AGV recognition rate is improved compared with traditional AprilTags.

3 Experiment and Comparation

3.1 Experimental Framework

We have built a simulation warehouse system based on the global vision using ROS and Gazebo. As shown in Fig. 6(a), four top cameras installed on the top of the second floor are used to monitor the warehouse of $12m \times 12m$, and their field of view (FOV) is 80°. The warehouse is divided into two floors for sorting and storage. After that, AGVs carry parcels at the right conveyor belt and deliver them to target areas including corresponding pipelines. After pouring parcels into corresponding pipelines, AGVs complete sorting tasks, return to starting areas, carry other parcels and start new tasks again. We simulate 20 AGVs and print AprilTags with family tag 36H11 on their tops. Particularly, each AGV in the Gazebo-based simulation environment shares the same model with the actual AGV. The tracking system calculates the distance of each AGV from a predetermined path assigned by the scheduling system to correct its deviation during the movement. The tracking results are presented in Fig. 6(b), where blue bounding boxes outside AGVs are predicted by the EKF, representing the preliminary estimates of AGV coordinates. In addition, green orientated bounding boxes are the detection results of improved YOLOv5, and then AprilTags in these boxes can be used to calculate AGV numbers, coordinates, and heading angles.

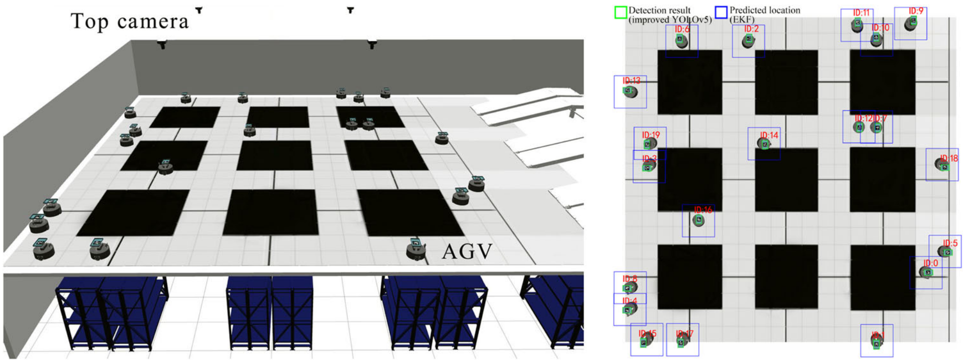
It is worth mentioning that 5000 pictures that were similar to Fig. 6(b) were collected, of which 80% were used for training while 20% were used for testing. Because the test environment is relatively fixed and the object is single, the model has an excellent recognition effect and can recognize all AGVs. To move smoothly, AGVs need to receive velocity instructions real-timely, which puts forward a higher requirement for the tracking system. Therefore, we compare the accuracies of heading angles calculated by various characteristic patterns and the real-time performances of various tracking algorithms in this section.

3.2 Contrastive Experiments of Location Methods Based on Global Vision

The simulation experiment aims at testing the accuracies of various characteristic patterns and calculation methods. We print four characteristic patterns on AGVs, including the QR code, two types of triangular patterns, and AprilTag in Fig. 7. All AGVs receive the same velocity instructions, including constant velocity instructions and random angular velocity instructions that are the Gaussian white noise with an average value of 0. After that, each AGV follows a straight line with small heading angles for comparing the calculation accuracies of different algorithms. In the simulation warehouse, we use four top cameras which are installed vertically downwards to capture the global image. The height of each camera is set to 6m, and a characteristic pattern occupies about 60 × 60 pixels in the global image.

We record the global image sequence of 100 frames, corresponding AGV locations, and IMU values. Then, we calculate the heading angles and locations of different AGVs in the global image. Since there is no external interference in the simulation environment, we assume that the AGV locations recorded in Gazebo and the heading angles calculated by the IMU data are both actual for comparing the location accuracies of different characteristic patterns. The actual distances from the given straight line and heading angles calculated from the IMU data in each frame are shown in Fig. 7(a) and (b), respectively.

Fig. 6 Simulation warehouse based on the global vision



(a) Simulation warehouse in Gazebo

(b) Tracking results

The QR code on the AGV1 is Version1, namely a 21 × 21 matrix. We use the improved Zxing algorithm [38] for recognizing QR codes, which has a higher recognition rate than other open-source algorithms. In addition to locating, the coordinates of three finder patterns are used to determine the heading angle of each AGV.

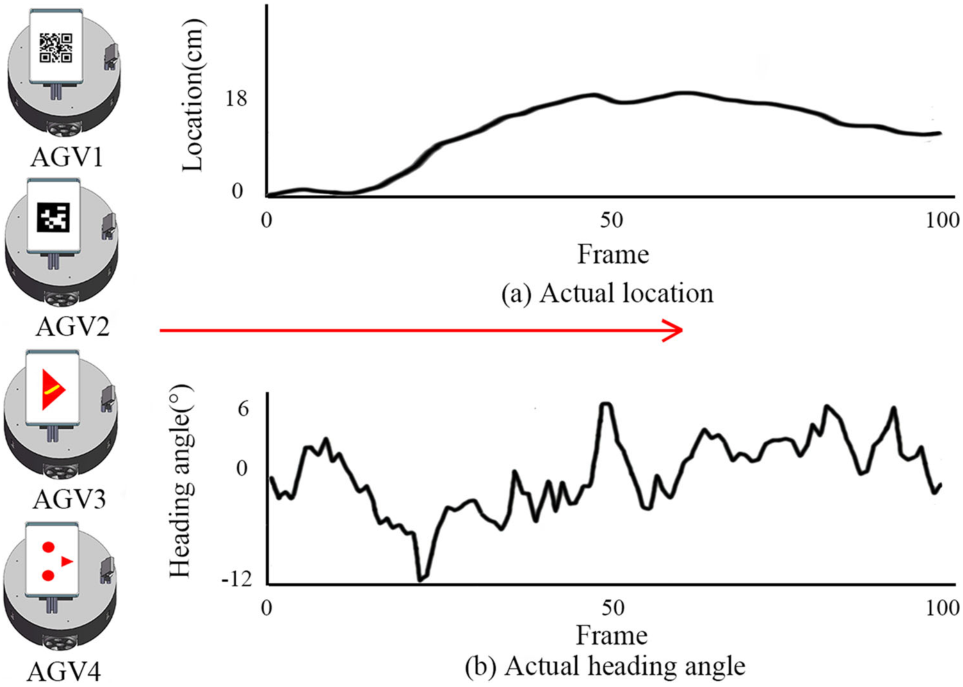
The triangle pattern II on the AGV3 includes a triangle and a digit [15], which is used to determine the AGV number and heading angle by extracting the digit and corner points. Similarly, Dönmez et al. calculated the midpoints of three patterns in the AGV4 as characteristic points and then set up a point-to-point navigation method. However, they did not calculate AGV heading angles [12, 13]. Here, the midpoints of three sub-patterns are used to calculate heading angles.

Referring to the convolutional neural network (CNN) example in MATLAB deep learning toolbox [39], we build a regression network based on CNN to calculate the rotation angle of the digit as the AGV heading angle. We rotate sampled images with digits in units of 0.05° as the training set. After detecting an AGV, CNN calculates its heading angle according to the bounding box containing a digit.

In Fig. 8(a), the black line is AGV heading angles calculated by the IMU data in the sequence of 100 frames. The distances of other curves close to the black line represent the calculation accuracies of different location methods. After that, we use box plots, as shown in Fig. 8(b), to compare their calculation accuracies intuitively. Simulation results show that the tracking system that combines the improved YOLOv5 and AprilTag system is most accurate. As a characteristic code storing enough information, the AprilTag makes the location effect of the improved YOLOv5 more stable and reduces the occurrence of large deviations.

The triangle pattern I has similar calculation accuracy with the tracking system and improved YOLOv5, whereas

Fig. 7 Poses of four AGVs in each frame



(a) Actual location

(b) Actual heading angle

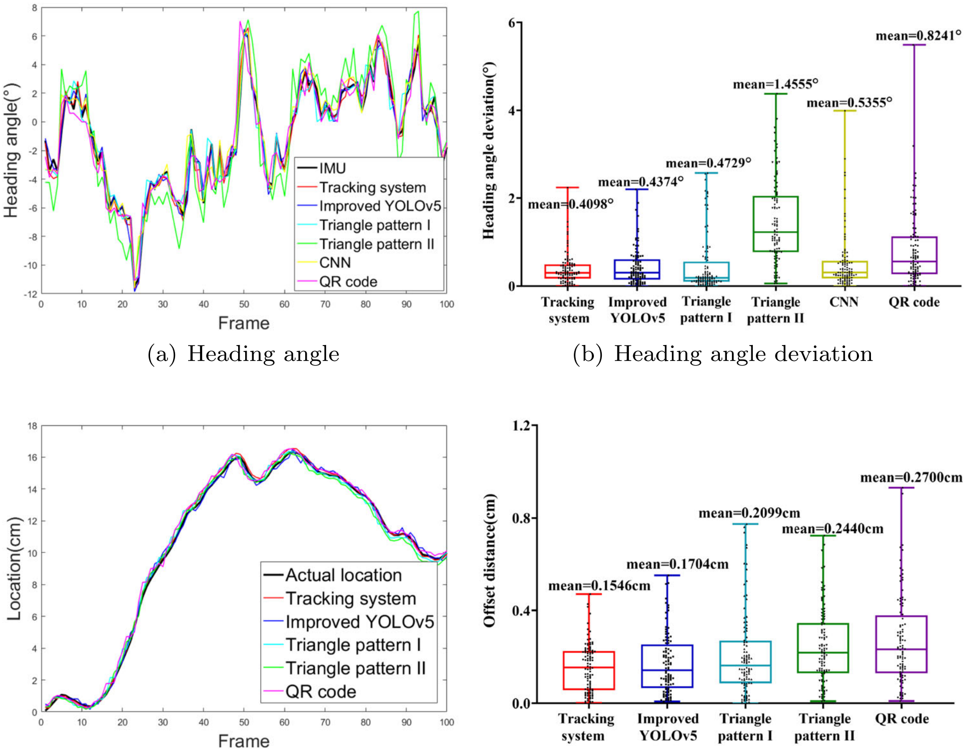
the locating effect of triangle pattern II is poor. Since corners are easy to lose, only extracting corners of the triangle pattern II to calculate the heading angle is a rough method. In contrast, using midpoints of multiple sub-patterns to calculate the heading angle is more effective, because midpoints are calculated by multiple contours of characteristic patterns, avoiding errors caused by missing a single point. However, the amount of information stored in triangle pattern I is limited, and each sub-pattern is required to have a certain distance. If several characteristic points are close to each other, the error of one pixel will lead to greater heading angle deviation.

Then, the midpoints of finder patterns in the QR code are used to calculate the heading angle. Since finder patterns occupy a small area, its location accuracy and recognition rate are low in this case. As for the CNN, 10,000 images of digits with different rotation angles are used as the training set. Simulation results show that this method has high accuracy, but there is no outstanding advantage over other characteristic patterns. As shown in Fig. 8(c) and (d), all location methods can be located with centimeter-level accuracy in the simulation experiment, their location accuracies are similar, but the tracking system has a slight advantage. It is worth noting that CNN is only used to calculate heading angles and determine AGV numbers in this experiment. Further research is needed for locating accurately.

To compare the accuracies of tracking algorithms at different resolutions, we set up cameras of different heights in the warehouse. Intuitively, heightening the camera expands the scope of the global image and reduces the number of cameras required by the warehouse system. But at the same time, the resolution of each characteristic pattern is also reduced, resulting in a worse locating effect. When the number of top cameras is reduced to 1, for capturing the entire warehouse, the camera height is set to 8m. Under this situation, the characteristic pattern of each AGV accounts for about 40 × 40 pixels. With other conditions fixed, the locating effects of these characteristic patterns are shown in Fig. 9.

When the pattern resolution decreases, the location influences of the tracking system, triangle pattern I, CNN, and improved YOLOv5 are small. However, the stability of the triangle pattern II is poor, and there is a big gap compared to others since decreasing the resolution of the characteristic pattern significantly reduces its accuracy. Compared with Fig. 8, the result of the QR code is missing as it is difficult to be recognized in this case by most open source algorithms. Therefore, it is necessary to choose the AprilTag with a large identification area and enough information as the characteristic pattern of the AGV.

As shown in Fig. 9(b) and (d), the calculation deviations of all methods become larger as the resolution decreases.



(a) Heading angle

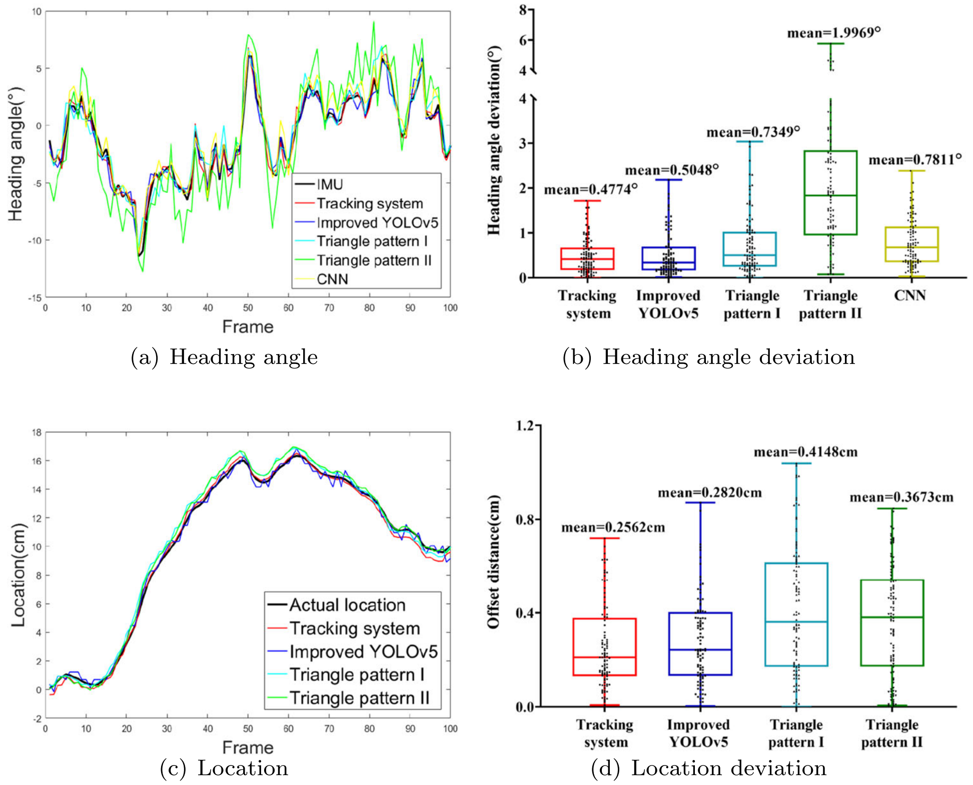
(b) Heading angle deviation

(c) Location

(d) Location deviation

Fig. 8 AGV pose calculated by different algorithms (The height of the top camera is 6m)

Fig. 9 AGV pose calculated by different algorithms (The height of the top camera is 8m)



(a) Heading angle

(b) Heading angle deviation

(c) Location

(d) Location deviation

Except for the triangle pattern II, other characteristic patterns have similar locating effects and satisfy the requirements of AGV navigation. Particularly, using CNN to calculate the heading angle is also robust when the characteristic pattern becomes smaller. However, this method needs to cooperate with another network to recognize the AGV number, which increases the complexity of tracking. Besides, the information stored by the digit is little, so it cannot be used to distinguish a large number of AGVs. As a consequence, the tracking system proposed in this paper can track the AGV and obtain its information at different resolutions precisely. Compared with other characteristic patterns and location methods, it is the most suitable for tracking multiple AGVs in the smart warehouse.

It is worth noting that we can get the exact location of the AGV conveniently in the Gazebo that can verify the location accuracy of the tracking method based on the global vision. Particularly, the parameters such as pixels and field of view (FOV) in the simulation camera are consistent with those in the actual scene. To be honest, the catch is that there is no camera distortion, but if the camera can be accurately calibrated, our tracking method can be smoothly transplanted to the actual scene.

3.3 Actual Experiments and Comparisons

This session reports laboratory experiments with two AGVs and a top camera as well as a comparison between the proposed tracking method based on global vision and other current methods of localization in warehouses. These AGVs are consistent with the one in the simulated environment and are commanded to move in circles around the experimental environment, where the surveillance camera is equipped at the height of 3m and calibrated. During this process, each tray and AprilTag on the AGV is detected with high accuracy, as shown in the green boxes in Fig. 10. Meanwhile, the AGV number, coordinates, heading angle, and AGV status are displayed in real-time. Particularly, *O* is the origin of the coordinates and positive values represent the AGV rotates clockwise.

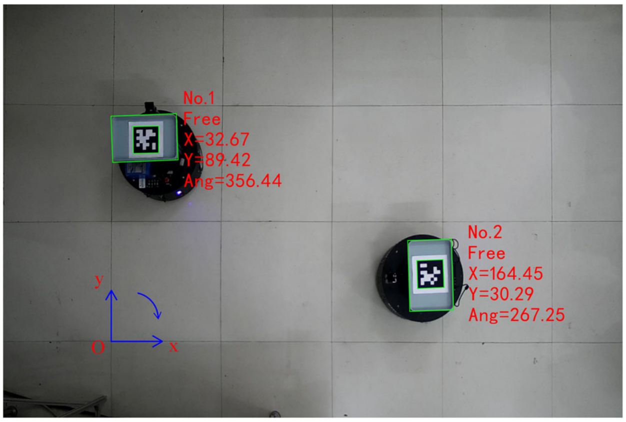


Fig. 10 Tracking results in the laboratory experiment

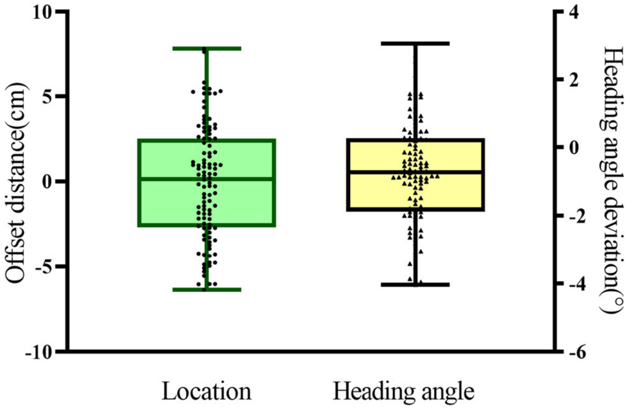


Fig. 11 Location errors in 100 laboratory experiments

In order to verify the effect of our algorithm in the real environment, we use global vision to track two AGVs in the laboratory and record their actual coordinates when AGVs stop. From Fig. 11, it can be clearly recognized that the mean and standard deviation of offset distance and heading angle deviation are 2.6cm, 2.8cm, 1.4°, and 1.5° respectively in terms of the tracking system based on the global vision in 100 experiments.

After that, we compare the tracking method based on global vision (global navigation) with other traditional navigation methods rely on state-of-the-art approaches as described in [8–10], and [11] respectively. In Table 1, the tracking method based on the global vision in the simulation environment has higher accuracy compared with other traditional navigation methods. However, affected by camera distortion and other noises, the tracking errors of this method are increased to 2.7 cm and 1.4° in the real environment. Therefore, the location accuracy of global vision can be further improved, and this method has greater development potential as the improvement of camera resolution and performance. Although the navigation accuracy of QR code navigation [8] and magnetic navigation [9] which have been used in the actual warehouse system is higher, they all rely on fixed markers (QR codes or magnetic markers), resulting in inflexible AGV systems. Other flexible navigation methods, such as LIO (lidar, IMU, and odometer) [10], are affected by cumulative errors seriously. As for a fusion location method [11] proposed by Xiong et al., some character patterns printed on the top of the warehouse are recognized by the vehicle camera to improve the positioning accuracy of lidar navigation. However, it uses fixed character codes to assist navigation, which also limits the movements of AGVs. In conclusion, the multi-AGV tracking method based on global vision still has a certain gap in accuracy compared with traditional AGV navigation methods, but it offers a trade-off solution between flexibility and accuracy. Although the method proposed in this paper cannot be applied to the outdoor environment, it can be easily applied to all kinds of the warehouse environment.

3.4 Real-Time Performance of Different Algorithms

In addition to the high locating accuracy, the tracking algorithm needs to satisfy the real-time requirements in the industry. For this reason, we test the running time of various tracking algorithms, including the Deep SORT [18], multi-AGV tracking system, and triangle tracking system [15]. To improve the real-time performance, we also select the YOLOv5 as the detection model of Deep SORT. Since the Deep SORT tracks objects after detecting, its complexity is high. This algorithm is suitable for the situation where the object movement trajectory is not clear. But for the warehouse system based on the global vision, the AGV trajectory can be obtained in advance according to the scheduling system. Thus, both the Deep SORT algorithm and multi-AGV tracking system obtain the AGV path information from the scheduling system for Kalman filtering update.

As shown in Fig. 12, the triangle tracking system proposed by Yang et al. extracts contours of triangle pattern II in the global image and selects contours with three corner points for AGV location, so its calculation complexity is low [15]. In addition, the motion model of AGV is not taken into account, so there is no need to interact with the scheduling system. Therefore, all AGV poses are identified and sent to the scheduling system in real-time. But for complex environments, only extracting polygon outlines to detect AGVs is prone to errors, and it is not reliable for scenes with uneven brightness. Therefore, such a method is difficult to apply in actual warehouses and factories. Compared with directly extracting the characteristic pattern, detecting AGVs through deep learning and decoding the characteristic pattern on its top have a better effect.

Table 1 Location accuracy of different navigation methods

Errors	Global navigation (simulation)	Global navigation (reality)	QR Code [8]	Magnetics [9]	LIO- mapping [10]	Fusion location [11]
Offset distance (cm)	0.3	2.7	0.5	0.8	5.7	0.9
Heading angle deviation(°)	0.5	1.4	1	0.5	6.2	1.7

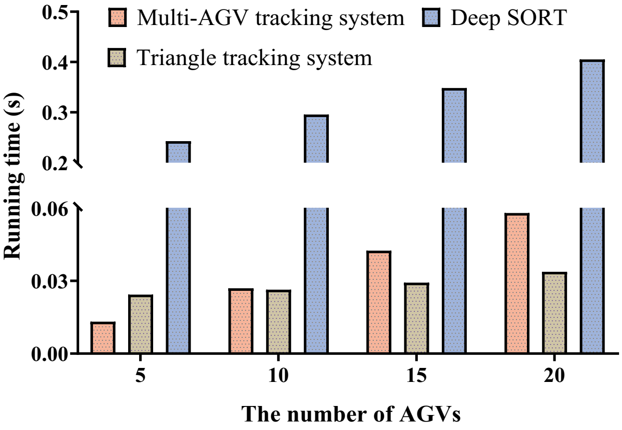


Fig. 12 Running time of different tracking algorithms

Although the triangle tracking system is faster than the tracking system, it reduces lots of necessary calculations, resulting in worse environmental adaptability and location effect. In contrast, the multi-AGV tacking system is combined with a scheduling system to improve the robustness and reduce the tracking time. Specifically, hierarchical input images are used in the improved YOLOv5, and most of the tracking processes rely on small local images. Furthermore, the tracking system omits the step of cascading matching and some decoding processes, so it is more real-time than Deep SORT but still has better tracking and emergency handling capabilities. Results from simulation experiments show that the running time of the multi-AGV tracking system increases with the number of AGVs. When there are hundreds of AGVs, the amount of calculation can be distributed to many servers according to different cameras for achieving real-time control of large-scale AGVs. Moreover, the multi-AGV tracking system has obvious advantages in accuracy and robustness, and the control center can achieve real-time control of multiple AGVs after tracking and scheduling.

4 Conclusion

This paper presents a multi-AGV tracking system that combines with the multi-AGV scheduling system, AprilTag system, improved YOLOv5 with OBB, and global vision. The tracking system obtains rough estimates of AGV locations in the global warehouse image based on the scheduling system and EKF, and then the AprilTag system is used to locate AGVs and output their numbers accurately.

By combining the strengths of each system, the cascading matching and most decoding processes are omitted when tracking large-scale AGVs. Furthermore, sending the hierarchical input images to the improved YOLOv5 can significantly reduce the computational load of the system on the basis of tracking all AGVs. When an AGV deviates from the established path, it still can be tracked and scheduled in the global image. These advantages improve the robustness and ability to handle emergencies of the tracking system. After that, we combine the AprilTag system with the scheduling system to improve its recognition rate that achieves 100% in the simulation experiment.

Experiments show that the multi-AGV tracking system can track a large number of AGVs with centimeter-level accuracy, and the calculation deviation of AGV heading angles does not exceed 1°. Meanwhile, the tracking algorithm is fast enough, even if combined with the scheduling algorithm, it can still control multiple AGVs in real-time. In addition, the feasibility of the tracking system based on global vision is verified in the simulation warehouse with large-scale AGVs. Although the multi-AGV tracking method based on global vision still has a certain gap in accuracy compared with traditional AGV navigation methods, it offers a trade-off solution between flexibility and accuracy since AGVs can move freely and accurately around the warehouse without fixed markers. After that, a mass of calculations are transferred to servers, reducing the AGV dependence on the on-board sensors and controllers. It provides a basis for adding complex neural networks and sophisticated control algorithms to the real-time scheduling of AGVs. In future work, we will study deep learning to reduce the dependence of the multi-AGV tracking algorithm on characteristic patterns and then use it in real warehouses.

Acknowledgements This work was supported by National Nature Science Foundation of China under the grant (61973125 and 61803161) and YangFan Innovative & Entrepreneurial Research Team Project of Guangdong Province (2016YT03G125).

Author Contributions Qifan Yang proposed a multi-AGV tracking system based on the global vision and combined it with a multi-AGV scheduling system and conducted simulation and experimental validation. Yindong Lian achieved locating and navigating multiple AGVs by using extended Kalman filtering (EKF). Yanru Liu achieved the improved YOLOv5 algorithm to locate the tray on the AGV. Wei Xie assisted the experiment and contributed to writing the manuscript. Yibin Yang designed the AGV model in the Gazebo. At last, all authors read and approved the final manuscript.

Data or code Availability Data or code may only be provided with restrictions upon request.

Declarations

Ethics approval This paper does not contain research that requires ethical approval.

Consent to participate All authors have consented to participate in the research study.

Consent for Publication All authors have read and agreed to publish this paper.

Conflict of Interests All authors declare that they have no conflict of interest.

References

1. Lee, J., Kao, H.A., Yang, S.: Shanhu, service innovation and smart analytics for industry 4.0 and big data environment. Procedia Cirp **16**, 3–8 (2014). https://doi.org/10.1016/j.procir.2014.02.001Get
2. Cavanini, L., Cicconi, P., Freddi, A., Germani, M., Longhi, S., Monteriu, A., Pallotta, E., Prist, M.: A preliminary study of a cyber physical system for industry 4.0: Modelling and co-simulation of an agv for smart factories. In: 2018 Workshop on Metrology for Industry 4.0 and IoT, pp. 169–174 (2018). https://doi.org/10.1109/METROI4.2018.8428334
3. Waschull, S., Bokhorst, J.A.C., Molleman, E., Wortmann, J.C.: Work design in future industrial production : Transforming towards cyber-physical systems. Comput. Indust. Eng. **139**(105679), 1–13 (2020). https://doi.org/10.1016/j.cie.2019.01.053
4. Lynch, L., Newe, T., Clifford, J., Coleman, J., Walsh, J., Toal, D.: Automated ground vehicle (agv) and sensor technologies-a review. In: 2018 12th International Conference on Sensing Technology (ICST) IEEE, pp. 347–352 (2018). https://doi.org/10.1109/ICSensT.2018.8603640
5. Yu, S., Yan, F., Zhuang, Y., Gu, D.: A deep-learning-based strategy for kidnapped robot problem in similar indoor environment. J. Intell. Robot. Syst. **100**(3), 765–775 (2020). https://doi.org/10.1007/s10846-020-01216-x
6. Qi, M., Li, X., Yan, X., Zhang, C.: On the evaluation of agvs-based warehouse operation performance. Simul. Model. Pract. Theory **87**, 379–394 (2018). https://doi.org/10.1016/j.simpat.2018.07.015
7. De Ryck, M., Versteyhe, M., Debrouwere, F.: Automated guided vehicle systems, state-of-the-art control algorithms and techniques. J. Manuf. Syst. **54**, 152–173 (2020). https://doi.org/10.1016/j.jmsy.2019.12.002
8. Quicktron AGV: http://www.flashhold.com/ (2021)
9. Omron mobile robot LD/HD series: https://www.fa.omron.com.cn/ (2021)
10. Ye, H., Chen, Y., Liu, M.: Tightly coupled 3d lidar inertial odometry and mapping. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 3144–3150 (2019). https://doi.org/10.1109/ICRA.2019.8793511
11. Xiong, J., Liu, Y., Ye, X., Han, L., Qian, H., Xu, Y.: A hybrid lidar-based indoor navigation system enhanced by ceiling visual codes for mobile robots. In: 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1715–1720 (2016). https://doi.org/10.1109/ROBIO.2016.7866575
12. Dönmez, E., Kocamaz, A.F., Dirik, M.: A vision-based real-time mobile robot controller design based on gaussian function for indoor environment. Arab. J. Sci. Eng. **43**(12), 7127–7142 (2018). https://doi.org/10.1007/s13369-017-2917-0
13. Dönmez, E., Kocamaz, A.F.: Multi target task distribution and path planning for multi-agents. In: 2018 International conference on artificial intelligence and data processing (IDAP), pp. 1–7 (2018). https://doi.org/10.1109/IDAP.2018.8620932
14. Boutteau, R., Rossi, R., Qin, L., Merriaux, P., Savatier, X.: A vision-based system for robot localization in large industrial environments. J. Intell. Robot. Syst. **99**(2), 359–370 (2020). https://doi.org/10.1007/s10846-019-01114-x
15. Yang, Q., Lian, Y., Xie, W.: Hierarchical planning for multiple agvs in warehouse based on global vision. Simulation Modelling Practice and Theory **104**, 102124, 1–15 (2020). https://doi.org/10.1016/j.simpat.2020.102124
16. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014)
17. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., online, S.imple., tracking, r.ealtime. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3464–3468 (2016). https://doi.org/10.1109/ICIP.2016.7533003
18. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649 (2017)
19. Redmon, J., Divvala, S.K., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection, arXiv: Computer Vision and Pattern Recognition. 779–788 (2016)
20. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6517–6525 (2017). https://doi.org/10.1109/CVPR.2017.690
21. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv:Computer Vision and Pattern Recognition, arXiv:1804.02767 (2018)
22. Kapania, S., Saini, D., Goyal, S., Thakur, N., Jain, R., Nagrath, P.: Multi object tracking with uavs using deep sort and yolov3 retinanet detection, framework. In: Proceedings of the 1st ACM Workshop on Autonomous and Intelligent Mobile Systems Association for Computing Machinery, pp. 1–6 (2020). https://doi.org/10.1145/3377283.3377284
23. Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., Zhao, X., Kim, T.-K.: Multiple object tracking: A literature review, arXiv:1409.7618 (2014)
24. Xu, Y., Zhou, X., Chen, S., Li, F.: Deep learning for multiple object tracking: a survey. IET Comput. Vis. **13**(4), 355–368 (2019). https://doi.org/10.1049/iet-cvi.2018.5598
25. Wang, J., Olson, E.: Apriltag 2: Efficient and robust fiducial detection. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4193–4198 (2016). https://doi.org/10.1109/IROS.2016.7759617
26. Lee, S., Tewolde, G.S., Lim, J., Kwon, J.: Qr-code based localization for indoor mobile robot with validation using a 3d optical tracking instrument. In: 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp. 965–970 (2015). https://doi.org/10.1109/AIM.2015.7222664
27. Quigley, M., Gerkey, B., Smart, W.D.: Programming Robots with ROS: a practical introduction to the Robot Operating System, O'Reilly Media, Inc (2015)
28. Olson, E.: Apriltag: A robust and flexible visual fiducial system. In: 2011 IEEE International Conference on Robotics and Automation, pp. 3400–3407 (2011). https://doi.org/10.1109/ICRA.2011.5979561
29. Pfrommer, B., Daniilidis, K.: Tagslam: Robust slam with fiducial markers. arXiv:1910.00679 (2019)
30. Grobler, P., Jordaan, H.: Autonomous vision based landing strategy for a rotary wing uav. In: 2020 International SAUPEC/RobMech/PRASA Conference, pp. 1–6 (2020). https://doi.org/10.1109/SAUPEC/RobMech/PRASA48453.2020.9041238
31. Bae, S., Yoon, K.: Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **40**(3), 595–610 (2018). https://doi.org/10.1109/TPAMI.2017.2691769
32. Xu, Y., Zhou, X., Chen, S., Li, F.: Deep learning for multiple object tracking: a survey. IET Comput. Vis. **13**(4), 355–368 (2019). https://doi.org/10.1049/iet-cvi.2018.5598

33. Wang, J., Ding, J., Guo, H., Cheng, W., Yang, W.: Mask obb: a semantic attention-based mask oriented bounding box representation for multi-category object detection in aerial images. Remote Sens. **11**(24), 2930 (2019). https://doi.org/10.3390/rs11242930
34. Chen, Z., Chen, K., Lin, W., See, J., Yang, C.: Piou loss: Towards accurate oriented object detection in complex environments. In: European Conference on Computer Vision (ECCV2020), Springer International Publishing, pp. 195–211 (2020). https://doi.org/10.1007/978-3-030-58558-7_12
35. Zhang, Z.: Camera calibration with one-dimensional objects. IEEE Trans. Pattern Anal. Mach. Intell. **26**(7), 892–899 (2004). https://doi.org/10.1109/TPAMI.2004.21
36. Wang, Y.M., Li, Y., Zheng, J.: A camera calibration technique based on opencv. In: The 3rd International Conference on Information Sciences and Interaction Sciences, pp. 403–406 (2010). https://doi.org/10.1109/ICICIS.2010.5534797
37. Huang, W., Maomin, A., Sun, Z.: Design and recognition of twodimensional code for mobile robot positioning. In: International conference on intelligent robotics and applications, pp. 662–672 (2019). https://doi.org/10.1007/978-3-030-27538-9_57
38. Qrcodescanner: https://github.com/heiBin/QrCodeScanner (2017)
39. Train convolutional neural network for regression: https://ww2.mathworks.cn/help/deeplearning (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Qifan Yang received the B.Eng. degree in electrical engineering and automation in 2019 from Jinan University, Guangzhou, China, and is currently working toward the M.S. degree in control engineering with the South China University of Technology, Guangzhou. His research interests include machine vision and robotics control.

Yindong Lian received the B.Eng. degree from the School of Automation Science and Engineering, South China University of Technology, Guangzhou, China, in 2016, where he is currently working toward the Ph.D. degree since 2017. His current research interests include mobile robot control, planning, and scheduling, as well as intelligent sorting and warehousing.

Yanru Liu received the B.Eng. degree in 2020 from Jinan University, Guangzhou, China, and is currently working toward the M.S. degree at South China University of Technology, Guangzhou, China. Her research interests include optimal control and machine vision.

Wei Xie received the B.Eng. degree in automation and M.Eng. degree in computer application technology from the Automation Department, Wuhan University of Science and Technology, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science and technology from the Kitami Institute of Technology, Japan, in 2003.He worked as a Postdoctoral Researcher with the Satellite Venture Business Laboratory from 2003 to 2006. In 2006, he joined the School of Automation Science and Engineering, South China University of Technology, Guangzhou, China, as an Associate Professor, where he was promoted to Full Professor, in 2010. He has authored or coauthored more than 80 articles in international journals and conferences. His research interests include intelligent robot, control theory, and application.

Yibin Yang received the B.Eng. degree in 2021 from South China University of Technology, Guangzhou, China, and is currently working toward the M.S. degree at South China University of Technology, Guangzhou, China. His research interests include mechanical design, robotics control, and deep learning.