

RESEARCH ARTICLE

PP-ST: An Indoor Mobile Robot Path Tracking Algorithm

YIBO CAO¹, JIAHENG ZHAO, HAIWEN ZHU², ZHENGDONG YANG, AND JINGWEN FAN

School of Software, South China Normal University, Guangzhou, Guangdong 510631, China

Corresponding author: Yibo Cao (422705879@qq.com)

This work was supported in part by the Key Technology Research (Second Batch) Program of Ningbo under Grant 2021E007; and in part by the Scientific Research Foundation of Graduate School, South China Normal University, under Grant 21RJKC15.

ABSTRACT Aiming at the problems that Pure Pursuit algorithm is too slow to approach the target path and take shortcuts at the corners under long forward-looking distance, and large heading jitter at the corners of the target path under short forward-looking distance, this paper proposes a path tracking algorithm based on Pure Pursuit and Stanley. First, a new nearest point selection strategy is proposed according to the need of algorithm fusion. Then, new algorithm coefficients are designed in order to adapt the algorithm to indoor environment. Finally, a new fusion method is proposed to solve the above problems. In order to verify the effectiveness of the proposed algorithm, it was tested in three different paths and compared with the improved Pure Pursuit algorithm proposed by two other related researchers in 2023. The experimental results show that the proposed fusion algorithm is 62.8% faster than the improved Pure Pursuit algorithm proposed by Macenski et al. in approaching the target path in indoor environment, and the proposed fusion algorithm's distance error is 40% less than that of the improved Pure Pursuit algorithm proposed by Macenski et al. in indoor environment; the proposed fusion algorithm's average heading angle error at the corners of the target path is 20.3% less than that of the improved Pure Pursuit algorithm proposed by Wu, Jiangdong et al. in indoor environment, and the proposed fusion algorithm's maximum heading angle error is 30.9% less than that of the improved Pure Pursuit algorithm proposed by Wu, Jiangdong et al. in indoor environment. Overall, the fusion algorithm is more suitable than the other two algorithms for mobile robot path tracking in indoor environments.

INDEX TERMS Forward-looking distance, path tracking, pure pursuit algorithm, Stanley algorithm.

I. INTRODUCTION

The mobile robot [1] constructs a map of its environment in an indoor environment with the help of sensor informations and Simultaneous Localization And Mapping (SLAM) [2] techniques for path planning [3] and path tracking [4]. Path tracking is an important component of mobile robot motion, and the goal of path tracking [5] is to control the robot to accurately follow the reference path given by path planning. The common path tracking algorithms include five main types: the Pure Pursuit algorithm [6], [7], [8], the Stanley algorithm [9], [10], [11], [12], the Proportion Integral Differential (PID) algorithm [13], the Linear Quadratic Regulator (LQR) algorithm [14], [15], and the

Model Predictive Control (MPC) algorithm [16], [17], [18]. Pure Pursuit algorithm and Stanley algorithm use only the geometric relationship between the path and the moving robot to realize tracking, which has simpler implementation principle, better anti-interference ability and better tracking results. Compared with the Pure Pursuit algorithm and Stanley algorithm, the PID algorithm cannot realize the precise control of the system state, and the parameter adjustment of the PID algorithm requires a certain amount of experience and skill, and often takes a long time to get the required parameters; The Model Predictive Control (MPC) algorithm needs to iteratively solve the optimization problem at each time step, and the solution of the optimization problem is often more time-consuming, and the action of the mobile robot need have stronger real-time requirements, and the Pure Pursuit algorithm and the Stanley algorithm

The associate editor coordinating the review of this manuscript and approving it for publication was Jamshed Iqbal¹.

have strong real-time performance. The Linear Quadratic Regulator (LQR) algorithm, compared to the Pure Pursuit and Stanley algorithms, cannot be used on paths with rapidly changing curvature and is less robust to external disturbances. In comparison, Pure Pursuit algorithm and Stanley algorithm are more suitable for mobile robots in indoor environments.

The idea of Pure Pursuit algorithm is to first determine the forward-looking point based on the forward-looking distance [19], and then determine the steering angle based on the heading angle and the distance between the current position and the forward-looking point position, and control the robot moving forward along the arc path. The advantages of Pure Pursuit algorithm [20] are that it has good proximity tracking effect, can preview the sudden change of the path direction, etc. The disadvantage of Pure Pursuit algorithm is that the tracking performance is limited by the forward-looking distance. If the forward-looking distance is too large, it will cause the tracking path to be too smooth, which will result in shortcuts [21] and too slow speed near the target path, thus greatly reducing the tracking accuracy and tracking efficiency; if the forward-looking distance is too small, the robot's heading will be jittered. Currently, many scholars study how to solve the shortcomings of the Pure Pursuit algorithm by choosing an optimal forward-looking distance. For example, some scholars adjust the forward-looking distance in a way that the forward-looking distance is directly proportional to the robot speed [22], [23]; Yu et al. [24] use a fuzzy rule controller to adjust the forward-looking distance; Serna et al. [25] estimate the forward-looking distance dynamically based on the speed and lateral error.

The idea of Stanley's algorithm is to determine the steering angle based on the distance between the front wheel and the nearest point on the target path. Compared to the Pure Pursuit algorithm, the Stanley algorithm differs in that it does not rely on the forward-looking distance to control the steering and movement of the robot and has the advantage of being able to approach the target path quickly.

In summary, the ability of Stanley algorithm to quickly approach the path can compensate for the slow approach speed of Pure Pursuit algorithm with long forward-looking distance [26]; using Pure Pursuit algorithm with short forward-looking distance [27] to enter the curve first and then using Stanley algorithm to quickly approach the path can solve the problem of taking shortcuts and large heading jitter by using the Pure Pursuit algorithm at the corner of the path. Therefore, in this paper, we propose an algorithm that combines the three advantages of fast path approaching, no jitter at the corners and no shortcuts at the corners [28], [29] by fusing the above two algorithms.

In the rest of the paper, related work is presented in Section II, which mainly includes the Ackermann geometric model used for the whole paper study and the principles of the two tracking algorithms. In Section III, the main work of this paper is presented, including the new nearest point selection

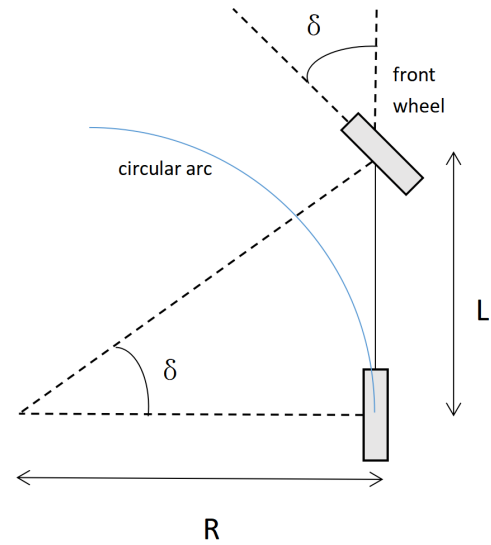


FIGURE 1. Ackermann geometric model.

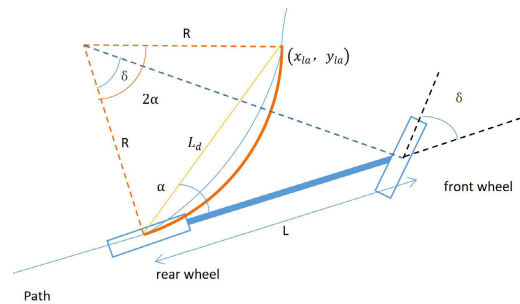


FIGURE 2. Geometry diagram of pure pursuit algorithm.

strategy, the fusion method, the fusion algorithm coefficient settings and the fusion algorithm control flow. In Section IV, the experimental results concerning the fusion algorithm are analyzed and compared with other Pure Pursuit algorithms. In Section V, the conclusions are drawn.

II. RELATED WORK

A. ACKERMANN GEOMETRY MODEL

A common Ackermann geometric model is shown in Fig. 1 [30], and let the radius of curvature of the vehicle steering in the model be R , the distance between the front and rear wheels be L , and the front wheel steering angle be δ . The relationship between the steering radius R and the front wheel steering angle δ is shown in Eq. 1:

$$\delta = \tan^{-1} \left(\frac{L}{R} \right) \quad (1)$$

B. PURE PURSUIT ALGORITHM

Fig. 2 shows the geometry diagram of the Pure Pursuit algorithm [31]. In the figure, α is the angle between the robot's heading and the line connecting the robot's rear

wheels and the target point (X_{la}, Y_{la}) , L_d is the forward-looking distance that is the distance between the robot's rear wheels and the target point [32], R is the radius of curvature that the robot needs to follow when steering using the Pure Pursuit algorithm, and δ is the front wheel steering angle. Based on the law of sines we can derive the following equation [33]:

$$\frac{L_d}{\sin 2\alpha} = \frac{R}{\sin (\frac{\pi}{2} - \alpha)} \tag{2}$$

After simplification, we get:

$$R = \frac{L_d}{2 \sin \alpha} \tag{3}$$

The relationship between steering angle and α can be obtained by combining Eq. 1 and 3 as follows:

$$\tan \delta = \frac{L}{\frac{L_d}{2 \sin \alpha}} = \frac{2L \sin \alpha}{L_d} \tag{4}$$

Therefore, the required steering angle to enable the robot to reach the target point can be given by the following equation:

$$\delta = \tan^{-1} \left(\frac{2L \sin(\alpha)}{L_d} \right) \tag{5}$$

As can be seen from Eq. 5, the Pure Pursuit algorithm has only one parameter which is the forward-looking distance (L_d) need to be adjusted [34]. The forward-looking distance is like a proportional gain factor. If the forward-looking distance is kept small, the robot tends to track the trajectory more accurately and the steering angle changes rapidly, but may cause the heading angle to oscillate [35]. If the forward-looking distance is kept large, the response becomes slower and may even see larger corner cuts in some cases, such as taking shortcuts around path corners, thus reducing tracking quality and safety [36], as shown in Fig. 3.

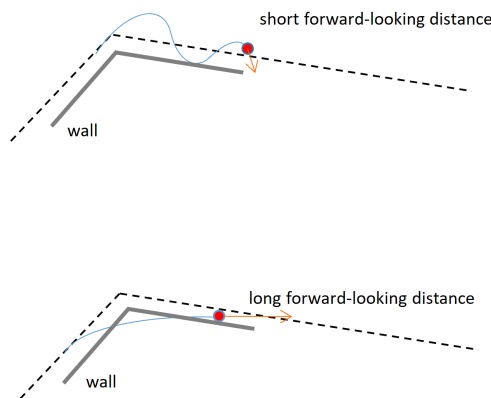


FIGURE 3. Diagram of tracking results at different forward-looking distances.

C. STANLEY ALGORITHM

The geometry diagram of Stanley's algorithm is shown in Fig. 4. In the figure, e_p is the distance from the center of the front wheel to the nearest point of the target path, V is

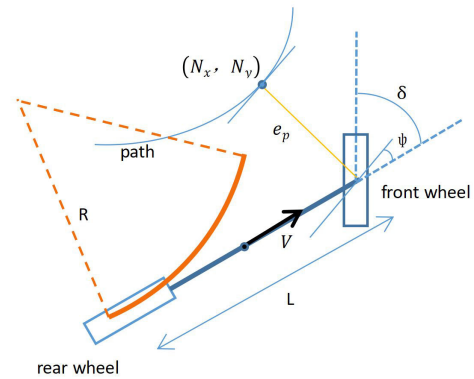


FIGURE 4. Geometry diagram of Stanley's algorithm.

the robot travel velocity [37], (N_x, N_y) is the nearest point on the target path to the front wheel, ψ is the heading error angle, and δ is the front wheel steering angle. The steering angle δ in Stanley's algorithm consists of two parts: 1. The steering angle ψ that is the angle between the robot's current heading and the tangential direction of the nearest point on the target path. 2. The steering angle caused by the lateral error.

According to Stanley's algorithm, the relationship between the steering radius R and the lateral distance e_p can be expressed in Eq. 6, where k is the coefficient used to adjust the convergence to the target path speed and L is the distance between the front and rear wheels.

$$R = \frac{LV}{ke_p} \tag{6}$$

Combining Eq. 1 and 6 we can derive the relationship between steering angle and distance e_p , as shown in Eq. 7.

$$\delta_{Stanley} = \psi + \tan^{-1} \frac{ke_p}{V} \tag{7}$$

III. METHODS

A. OVERALL CONTROL FLOW OF THE FUSION ALGORITHM

The flow of the fusion algorithm is shown in Fig. 5. First, the target path position and the initial position of the robot are obtained. Second, the closest distance of the robot from the target path is calculated. Third, the tracking algorithm is selected based on the closest distance. Fourth, output the steering angle that the robot needs according to the selected tracking algorithm. Fifth, the robot outputs its actual position. Sixth, determine whether the robot has reached the end of the target path, and if so, the tracking ends, otherwise, return to the second step.

B. NEW NEAREST POINT SELECTION STRATEGY

The new nearest point selection strategy has two main improvements: taking the rear wheel of the robot as the center point and optimizing the nearest distance calculation process.

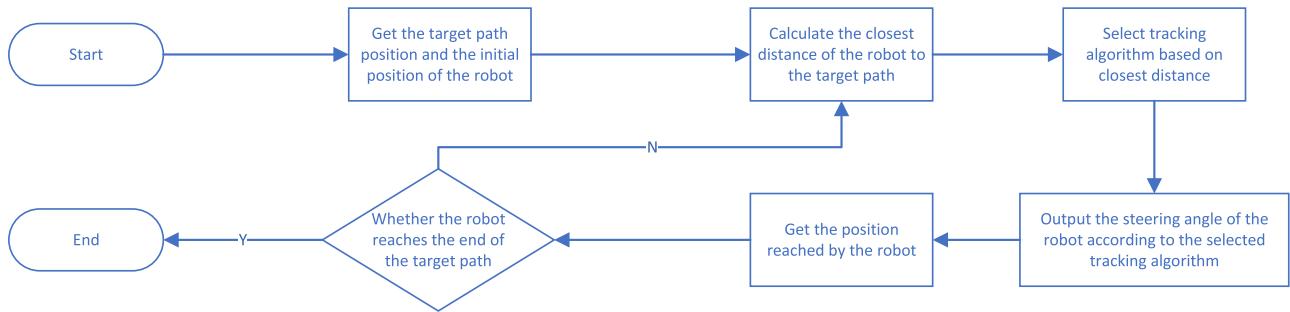


FIGURE 5. Overall control flow of the fusion algorithm.

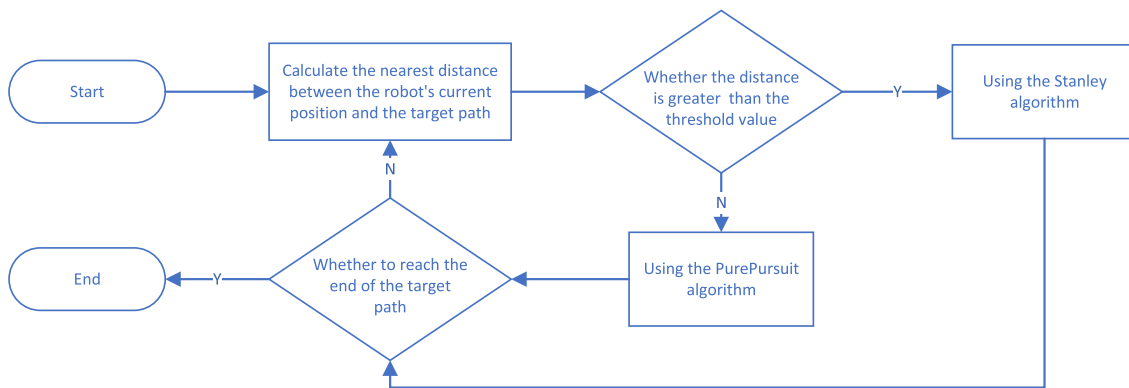


FIGURE 6. Flow chart of fusion method.

1) PICK THE CENTER POINT OF THE ROBOT

The Pure Pursuit algorithm takes the rear wheel as the center to search the target point on the path, while the Stanley algorithm takes the front wheel as the center to calculate the shortest distance from the path, so the two algorithms need to unify their center points, and this paper chooses the rear wheel as the center point. After determining the rear wheel as the center point, the fusion algorithm can calculate the closest distance of the rear wheel from the path, which is used as the basis of judgment when the fusion algorithm switches algorithms; the Stanley algorithm module can calculate the position of the front wheel based on the position of the rear wheel, the distance between the front and rear wheels of the robot and the robot's heading angle, and then calculate the closest distance of the front wheel from the path; the Pure Pursuit algorithm module can use the rear wheel as the center to search the target point on the path.

2) OPTIMIZE THE NEAREST DISTANCE CALCULATION PROCESS

The main element of the optimization method for nearest distance calculation is to select path points only from the path points within a small area of radius R_s , and the center of the small area is the center point of the robot. This eliminates the need to traverse the entire path in each algorithm cycle, which reduces the need of computational power and increases the algorithm response time.

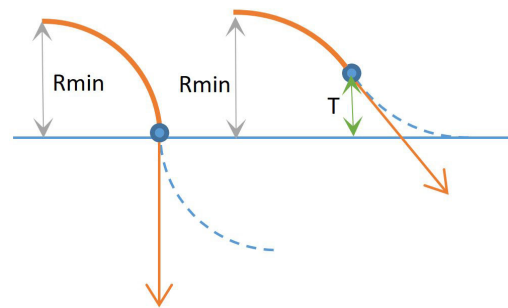


FIGURE 7. Threshold schematic.

C. FUSION METHOD

The fusion method is to select the tracking algorithm based on the distance between the robot's center point and the nearest point on the path. When the robot deviates from the target path at a distance greater than the threshold value, the Stanley algorithm is used for tracking; otherwise, the Pure Pursuit algorithm is used for tracking. The flow chart of the fusion method is shown in Fig. 6.

D. FUSION ALGORITHM COEFFICIENT SETTING

We need to design the coefficients which are applicable in indoor environment. The coefficients needed in the fusion

algorithm are the forward-looking distance, the convergence velocity to the target path coefficient k , the threshold and the radius of the computational range of the closest distance R_s . Indoor environments have less space than outdoor environments, and the complexity of indoor environments is higher than that of outdoor environments, Pure Pursuit and Stanley are often used in self-driving cars in outdoor environments, and this paper is to apply Pure Pursuit and Stanley to mobile robots in indoor environments, which are much smaller than self-driving cars in terms of size, and the algorithmic coefficients of Pure Pursuit and Stanley, which are applied to self-driving cars, are not suitable for mobile robots, so we need to find out algorithmic coefficients applicable to mobile robots in indoor environments.

After reading a large number of papers related to the Pure Pursuit algorithm [30], we found that the selection of the forward-looking distance in the Pure Pursuit algorithm is generally between about 0.5 times and 1.5 times the distance between the front and rear wheel spacing of the mobile robot trolley being used, with the best tracking accuracy at 0.5 times, and smoother tracking at 1.5 times, and the distance between the front and back wheels of the mobile robot trolleys used in this paper is 0.13m. After many experiments of trial and verification and in order to improve the tracking accuracy, the forward looking distance of 0.05m is more suitable for the mobile robot used in this paper; Similarly, by reading Stanley's algorithm related papers [38], we found that the convergence velocity to the target path coefficient value k is generally from 1 to 10, k is the slowest convergence to the target path when k is 1, and the fastest convergence to the target path when k is 10. After many experiments of trial and verification, and in order to improve the convergence velocity to the target path, we get k is 5, which is more suitable for the mobile robot used in this paper. In the extreme case, the mobile robot steers with the minimum steering radius to approach the target path, and in order to prevent the mobile robot from steering with the minimum steering radius too late to return to the correct heading, the threshold T is chosen to be half of the minimum steering radius, as shown in Fig. 7, the blue solid line indicates the target path, the blue solid dot indicates the position of the mobile robot, the orange solid line indicates the traveled trajectory, the orange arrow line indicates the mobile robot heading, the blue dashed line indicates the future trajectory of the mobile robot, the gray double-headed line indicates the minimum steering radius R_{min} , and the green double-headed line indicates the threshold value T . In this paper, the left and right wheelbase of the mobile robot is 0.235m, and the minimum steering radius is about 0.11m, so the threshold T in this paper is 0.05m; setting a nearest-distance computation range radius R_s avoids each algorithmic cycle from traversing the target path once, and saves the arithmetic power. After many experiments, we found that the mobile robot needs at most 0.5m traveling distance to complete the steering or close to the path, so we only need to consider the target path within the radius R_s every time, and for the sake of stability, we set

TABLE 1. Coefficient table of fusion algorithm.

Coefficient name	Coefficient value
Forward Looking Distance L_d	0.05m
Convergence to the target path velocity coefficient k	5
Threshold value T	0.05m
Nearest distance calculation range radius R_s	0.6m

the nearest distance computation range radius R_s to 0.6m. All the coefficients are shown in Tab. 1.

E. CONTROL FLOW TO SOLVE THE PROBLEM OF APPROACHING PATH TOO SLOW

As shown in Fig. 8, at the beginning the robot deviates from the target path far away and uses the Stanley algorithm for tracking to approach the target path quickly, and then uses the Pure Pursuit algorithm for tracking when it deviates from the target path at a distance less than the threshold value.

F. CONTROL FLOW CHART TO SOLVE DEVIATING FROM THE PATH AND TAKING SHORTCUTS AT PATH CORNER

As shown in Fig. 9, when the robot encounters a corner on the target path, at this time, it first uses Pure Pursuit algorithm with short forward-looking distance to enter the corner, and then uses the Stanley algorithm to track when the robot deviates from the target path beyond the threshold value, so that the robot quickly approaches the path, and then uses the Pure Pursuit algorithm when the robot deviates from the target path by a distance less than the threshold value.

IV. EXPERIMENTS AND ANALYSIS

We first conducted experiments in these environments of deviated path and path corner in order to verify the effectiveness of the fusion algorithm, and then a hybrid path was used for simulation and analysis in order to verify the overall performance of the proposed fusion algorithm. The paths included in the hybrid path are: circular path, bow path, arch path and Z-path. These four paths represent common driving states of the mobile robot, including circular driving, bow route driving, obstacle avoidance and lane changing. It is worth mentioning that the obstacle avoidance path in this paper refers to one of the already generated paths to be tracked, and the algorithm in this paper only needs to track the paths.

TABLE 2. Parameters of the experimental robot.

Parameter Name	Parameter Value	Unit
Distance between the front and rear wheels L	0.13m	m
Distance between left and right wheels D	0.235m	m
Initial Location(x,y)	(0,0)	m
Initial heading θ	0	rad
Velocity v	0.3	m/s

To highlight the experimental results, we compare the performance of the fusion algorithm with two other Pure Pursuit algorithms. We mainly compare with two improved Pure Pursuit algorithms proposed by Macenski [39] et al. and

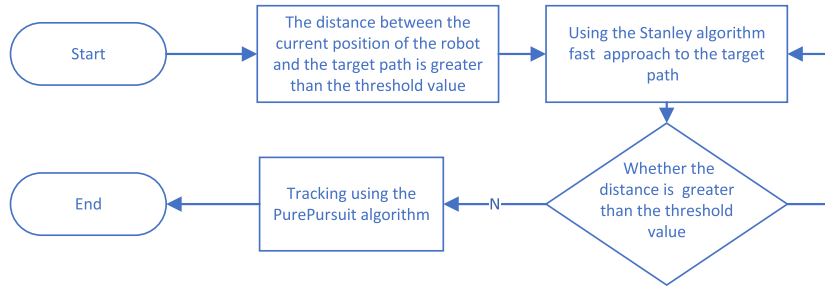


FIGURE 8. Control flow chart to solve approaching path too slow.

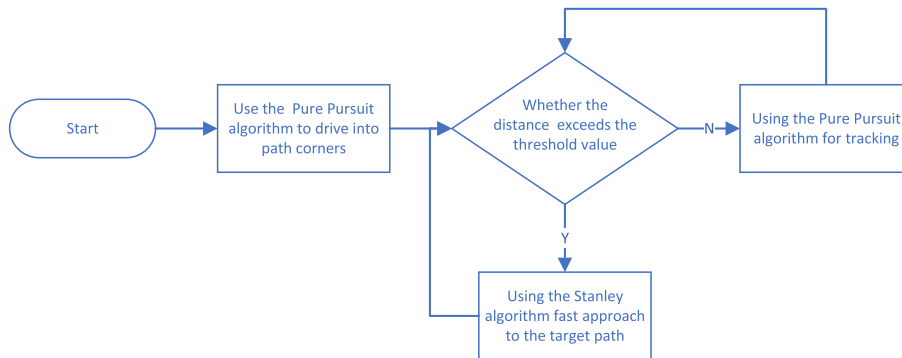


FIGURE 9. Control flow chart at the corner.

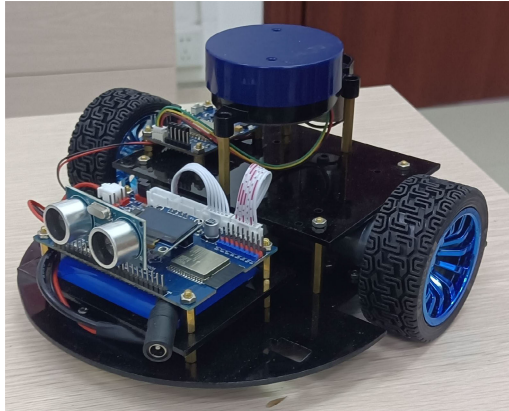


FIGURE 10. Physical drawing of the experimental mobile robot.

Wu, Jiangdong [40] et al. in 2023. For ease of presentation, the next part of the paper will be organized as sm-pp for the improved Pure Pursuit algorithm proposed by Macenski et al. wjd-pp for the improved Pure Pursuit algorithm proposed by Wu et al. ppst for the fusion algorithm proposed in this paper.

The equipment used in the experiment includes a computer responsible for issuing commands and a mobile robot responsible for accepting commands and executing them. The platforms used are the operating system Ubuntu 22.04.3 LTS and the robot operating system ros2, and the computer and the mobile robot communicate with each other through wifi. The programming language used for the experiment is c++,

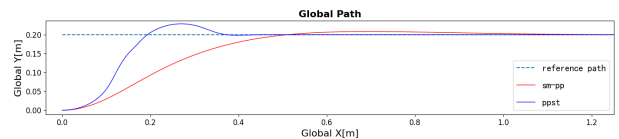


FIGURE 11. Trace path diagram for ppst and sm-pp.

and the programming language used for the data analysis is python. The computer uses the robot operating system ros2 platform to issue velocity and angular velocity control commands and receives the robot position information from the mobile robot. The mobile robot receives speed and angular velocity control commands via wifi and sends back the position information recorded by the odometer to the computer. The position information comes from the odometer, the speed is 0.3m/s, and the angular velocity is V/R (R is calculated by Pure Pursuit algorithm or Stanley algorithm).

In this experiment, the equipment parameters of the mobile robot used are shown in Tab. 2 and the mobile robot used is shown in Fig. 10.

A. VERIFY FAST APPROACH PATH

In order to verify the effect of fast approaching the target path, a straight line of 2 m long was chosen as the target path for the experiment, and the initial position of the robot was at a vertical distance of 0.2 m from the head of the path. Fig. 11 represents the tracking path diagrams of ppst and

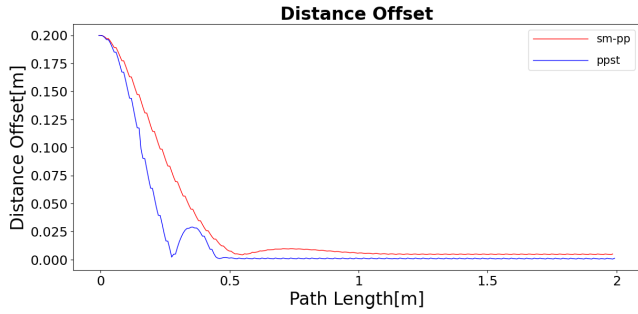


FIGURE 12. Plot of ppst and sm-pp distance error variation.

sm-pp, from which we can see that ppst approaches the path faster than 0.25 and has no heading fluctuation. In this figure, the blue dashed line indicates the path need to be tracked, and the solid lines represent the tracking trajectories of different algorithms, respectively; the starting point of the mobile robot is at the bottom left corner of the figure, and the end point is at the right side of the figure. Fig. 12 represents the variation of distance error for ppst and sm-pp, from which we can see that the distance error of ppst is generally smaller than that of sm-pp and converges to zero faster than sm-pp.

TABLE 3. Distance error and angular velocity taken to approach the path.

Algorithm	average value	maximum values	distance error value
ppst(ours)	0.475rad/s	2.553rad/s	0.018m
sm-pp	0.181rad/s	1.92rad/s	0.03m

TABLE 4. Time and Global X taken to approach the path.

	ppst	sm-pp	ppst	sm-pp
first approach	1s	1.9s	0.198m	0.525m
finish approach	1.6s	4.3s	0.368m	1.244m

Tab. 3 shows the average of the distance error for ppst and sm-pp. we can see that the average distance error of ppst is 0.012m smaller than that of sm-pp, which indicates that ppst has higher tracking accuracy than sm-pp on approaching path. Tab. 3 also shows the average and maximum values of angular velocity for ppst and sm-pp, and we can see that the average and maximum angular velocities of ppst are larger than those of sm-pp, which indicates that ppst has better steering ability than sm-pp. Tab. 4 shows the time required to approach the path for the first time and the total time to complete the approach for both algorithms. ppst takes 0.9s faster than sm-pp to approach the path for the first time, and ppst takes 2.7s faster than sm-pp to complete the approach. Tab. 4 also shows the Global X-coordinates of ppst and sm-pp when approaching the path for the first time and the Global X-coordinates of ppst and sm-pp when completing the approach, from the table we can see that ppst is 0.327m faster than sm-pp at the first approach and ppst is 0.876m faster than sm-pp at the completion of the approach.

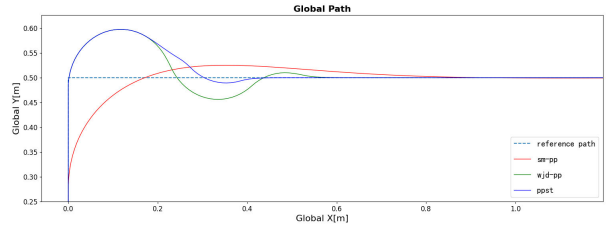


FIGURE 13. Trace path diagram of the three algorithms.

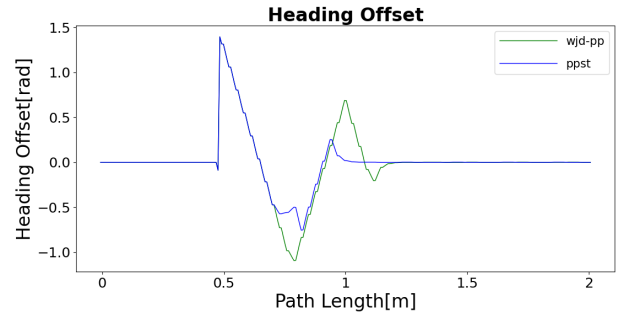


FIGURE 14. Plot of ppst and wjd-pp heading error variation.

In summary, ppst is faster than sm-pp in approaching to the target path.

B. VERIFY THAT THERE ARE NO SHORTCUTS AROUND THE CORNERS OF THE PATH AND THAT THE HEADING IS STABLE

To verify the tracking effect of the fusion algorithm at the corner of the target path, a right-angle line with a length of 2 m was selected as the target path for the experiment, and the initial position of the robot was at the position of the first part of the path.

Fig. 13 shows the tracking path diagrams of the three algorithms. From the diagram, we can see that sm-pp has a shortcut, wjd-pp does not take a shortcut, but its heading jitters a lot after crossing the corner, and ppst achieves both no shortcut and a more stable heading. In this figure, the blue dashed line indicates the path need to be tracked, and the solid lines represent the tracking trajectories of different algorithms, respectively; the starting point of the mobile robot is at the bottom left corner of the figure, and the end point is at the right side of the figure.

Fig. 14 represents the graph of the variation of the heading error of ppst and wjd-pp, and Tab. 5 shows the average and maximum values of the heading error after crossing the corner of ppst and wjd-pp. From Fig. 14 we can see that the maximum value of the heading error of ppst is smaller than that of wjd-pp, and also the overall heading error of ppst is smaller than that of wjd-pp. From Tab. 5, we can see that the average heading error of ppst is 0.077 rad less than that of wjd-pp, and the maximum heading error is 0.338 rad less than that of wjd-pp. In summary, ppst is more stable than wjd-pp in heading around corner.

TABLE 5. Average and maximum values of the heading error.

Algorithm	average value	maximum value
ppst	0.302rad	0.754rad
wjd-pp	0.379rad	1.092rad

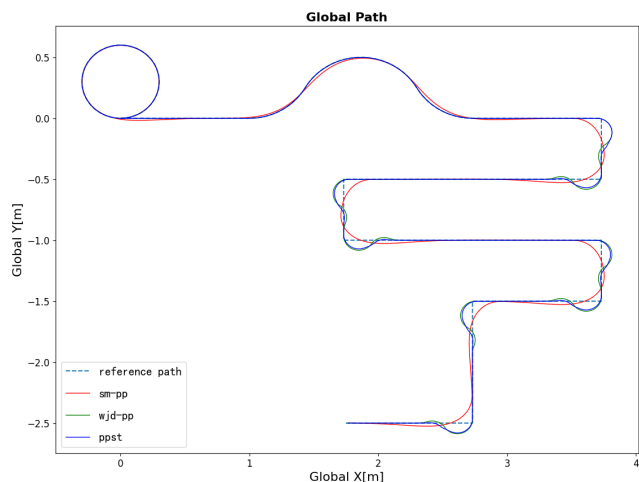


FIGURE 15. Trace path diagram of the three algorithms.

TABLE 6. The distance error and heading error of the three algorithms.

	ppst	sm-pp	wjd-pp
distance error average value	0.01 m	0.011 m	0.012 m
distance error maximum value	0.082 m	0.064 m	0.092 m
heading error average value	0.084 rad	0.092 rad	0.106 rad
heading error maximum value	1.228 rad	0.828 rad	1.316 rad

C. HYBRID PATH

In order to verify the tracking accuracy of the fusion algorithm, a hybrid path is used as the target path for the experiment, and the initial position of the robot is at the location of the path head. The paths included in the hybrid path are: a circular path with a radius of 0.3 m, a bow-shaped path with a length of 2 m and a width of 0.5 m, an arch-shaped path with a radius of 0.5 m, and a Z-path with a side length of 1m.

Fig. 15 shows the tracking path diagrams of the three algorithms. In this figure, the blue dashed line indicates the path need to be tracked, and the solid lines represent the tracking trajectories of different algorithms, respectively. The starting point of the mobile robot is in the upper left corner of the figure, and the end point is in the lower center of the figure.

Tab. 6 shows the experimental data of the distance error and heading error of the three algorithms. From Tab. 6 we can see that on the hybrid path, the average distance error and average heading error of the three algorithms are not very different, the average distance error and average heading error of ppst is the smallest, and the maximum distance error and maximum heading error of ppst is smaller than that of wjd-pp and larger than that of sm-pp. Although the maximum distance error and

maximum heading error of sm-pp are the smallest, sm-pp has the problem of taking shortcuts and approaching the path too slowly, and the tracking effect is far inferior to that of ppst. Overall, the tracking effect of ppst is the best.

V. CONCLUSION

In this paper, we propose a path tracking algorithm based on fusing Pure Pursuit algorithm and Stanley algorithm. First, a new nearest point selection strategy is proposed according to the need of the algorithm. Then, new algorithm coefficients are designed in order to adapt the algorithm to the indoor environment. Finally, a new fusion method is proposed in order to solve the shortcomings of Pure Pursuit algorithm. In this paper, we first conducted experiments in these environments of deviated path and path corner in order to verify the effectiveness of the fusion algorithm. The experimental results show that the fusion algorithm solves the problems of approaching the path too slowly when the Pure Pursuit algorithm deviates from the path, taking shortcuts at the corners of the path, and having large heading jitter. In the indoor environment, the fusion algorithm is 62.8% faster than the improved Pure Pursuit algorithm proposed by Macenski [39] et al. in approaching the target path, and the proposed fusion algorithm’s distance error is 40% less than that of the improved Pure Pursuit algorithm proposed by Macenski et al. in approaching the target path, the fusion algorithm’s average heading angle error at the corners of the target path is 20.3% less than that of the improved Pure Pursuit algorithm proposed by Wu et al. and the fusion algorithm’s maximum heading angle error is 30.9% less than that of the improved Pure Pursuit algorithm proposed by Wu et al. Then, to verify the overall performance of the proposed fusion algorithm, a hybrid path was used to conducted. The experimental results show that the average distance error and the average heading error of the fusion algorithm are the smallest, and the fusion algorithm does not have the drawbacks of the other two algorithms. Overall, the fusion algorithm is more suitable for path tracking than the other two algorithms in the indoor environment.

Tracking using the Pure Pursuit algorithm and Stanley’s algorithm can only travel in arcs and not in straight lines, the next intention is to set the steering radius to an exceptionally large value so that a small arc is equivalent to a straight line under microscopic conditions, so as to make the mobile robot straighter when traveling in a straight line. At the same time, we will set the obstacle avoidance strategy according to the motion characteristics of the Pure Pursuit algorithm at different forward looking distances.

REFERENCES

- [1] Z. Sun, R. Wang, Q. Ye, Z. Wei, and B. Yan, “Investigation of intelligent vehicle path tracking based on longitudinal and lateral coordinated control,” *IEEE Access*, vol. 8, pp. 105031–105046, 2020, doi: 10.1109/ACCESS.2020.2994437.
- [2] R. Fareh, M. Baziyad, T. Rabie, and M. Bettayeb, “Enhancing path quality of real-time path planning algorithms for mobile robots: A sequential linear paths approach,” *IEEE Access*, vol. 8, pp. 167090–167104, 2020, doi: 10.1109/ACCESS.2020.3016525.

- [3] C. Dai, C. Zong, and G. Chen, "Path tracking control based on model predictive control with adaptive preview characteristics and speed-assisted constraint," *IEEE Access*, vol. 8, pp. 184697–184709, 2020, doi: [10.1109/ACCESS.2020.3029635](https://doi.org/10.1109/ACCESS.2020.3029635).
- [4] M. G. B. Atia, H. El-Hussieny, and O. Salah, "A supervisory-based collaborative obstacle-guided path refinement algorithm for path planning in wide terrains," *IEEE Access*, vol. 8, pp. 214672–214684, 2020, doi: [10.1109/ACCESS.2020.3041802](https://doi.org/10.1109/ACCESS.2020.3041802).
- [5] Z. Chen, Y. Zhang, Y. Zhang, Y. Nie, J. Tang, and S. Zhu, "A hybrid path planning algorithm for unmanned surface vehicles in complex environment with dynamic obstacles," *IEEE Access*, vol. 7, pp. 126439–126449, 2019, doi: [10.1109/ACCESS.2019.2936689](https://doi.org/10.1109/ACCESS.2019.2936689).
- [6] L. Wang, Z. Chen, and W. Zhu, "An improved pure pursuit path tracking control method based on heading error rate," *Ind. Robot, Int. J. Robot. Res. Appl.*, vol. 49, no. 5, pp. 973–980, Jun. 2022, doi: [10.1108/IR-11-2021-0257](https://doi.org/10.1108/IR-11-2021-0257).
- [7] Y. Yang, Y. Li, X. Wen, G. Zhang, Q. Ma, S. Cheng, J. Qi, L. Xu, and L. Chen, "An optimal goal point determination algorithm for automatic navigation of agricultural machinery: Improving the tracking accuracy of the pure pursuit algorithm," *Comput. Electron. Agricult.*, vol. 194, Mar. 2022, Art. no. 106760, doi: [10.1016/j.compag.2022.106760](https://doi.org/10.1016/j.compag.2022.106760).
- [8] J. Ahn, S. Shin, M. Kim, and J. Park, "Accurate path tracking by adjusting look-ahead point in pure pursuit method," *Int. J. Automot. Technol.*, vol. 22, no. 1, pp. 119–129, Feb. 2021, doi: [10.1007/s12239-021-0013-7](https://doi.org/10.1007/s12239-021-0013-7).
- [9] N. H. Amer, K. Hudha, H. Zamzuri, V. R. Aparow, A. F. Z. Abidin, Z. A. Kadir, and M. Murrad, "Adaptive modified Stanley controller with fuzzy supervisory system for trajectory tracking of an autonomous armoured vehicle," *Robot. Auto. Syst.*, vol. 105, pp. 94–111, Jul. 2018, doi: [10.1016/j.robot.2018.03.006](https://doi.org/10.1016/j.robot.2018.03.006).
- [10] N. H. Amer, H. Zamzuri, K. Hudha, V. R. Aparow, Z. A. Kadir, and A. F. Z. Abidin, "Path tracking controller of an autonomous armoured vehicle using modified Stanley controller optimized with particle swarm optimization," *J. Brazilian Soc. Mech. Sci. Eng.*, vol. 40, no. 2, p. 104, Feb. 2018, doi: [10.1007/s40430-017-0945-z](https://doi.org/10.1007/s40430-017-0945-z).
- [11] A. AbdElmoniem, A. Osama, M. Abdelaziz, and S. A. Maged, "A path-tracking algorithm using predictive Stanley lateral controller," *Int. J. Adv. Robotic Syst.*, vol. 17, no. 6, 2020, doi: [10.1177/1729881420974852](https://doi.org/10.1177/1729881420974852).
- [12] S. H. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, and J. Diebel, "Stanley: The robot that won the DARPA grand challenge," *Field Robot.*, vol. 23, pp. 661–692, Jul. 2006.
- [13] Y. Chen, Y. Shan, L. Chen, K. Huang, and D. Cao, "Optimization of pure pursuit controller based on PID controller and low-pass filter," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Maui, HI, USA, Nov. 2018, pp. 3294–3299.
- [14] C. Piao, X. Liu, and C. Lu, "Lateral control using parameter self-tuning LQR on autonomous vehicle," in *Proc. Int. Conf. Intell. Comput., Autom. Syst. (ICICAS)*, Dec. 2019, pp. 913–917, doi: [10.1109/ICICAS48597.2019.00197](https://doi.org/10.1109/ICICAS48597.2019.00197).
- [15] E. Alcalá, V. Puig, J. Quevedo, T. Escobet, and R. Comasolivas, "Autonomous vehicle control using a kinematic Lyapunov-based technique with LQR-LMI tuning," *Control Eng. Pract.*, vol. 73, pp. 1–12, Apr. 2018, doi: [10.1016/j.conengprac.2017.12.004](https://doi.org/10.1016/j.conengprac.2017.12.004).
- [16] C. Sun, X. Zhang, Q. Zhou, and Y. Tian, "A model predictive controller with switched tracking error for autonomous vehicle path tracking," *IEEE Access*, vol. 7, pp. 53103–53114, 2019, doi: [10.1109/ACCESS.2019.2912094](https://doi.org/10.1109/ACCESS.2019.2912094).
- [17] B. Zhang, C. Zong, G. Chen, and B. Zhang, "Electrical vehicle path tracking based model predictive control with a Laguerre function and exponential weight," *IEEE Access*, vol. 7, pp. 17082–17097, 2019, doi: [10.1109/ACCESS.2019.2892746](https://doi.org/10.1109/ACCESS.2019.2892746).
- [18] H. Peng, W. Wang, Q. An, C. Xiang, and L. Li, "Path tracking and direct yaw moment coordinated control based on robust MPC with the finite time horizon for autonomous independent-drive vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6053–6066, Jun. 2020, doi: [10.1109/TVT.2020.2981619](https://doi.org/10.1109/TVT.2020.2981619).
- [19] M. Elbanhawi, M. Simic, and R. Jazar, "Receding horizon lateral vehicle control for pure pursuit path tracking," *J. Vibrat. Control*, vol. 24, no. 3, pp. 619–642, Feb. 2018, doi: [10.1177/1077546316646906](https://doi.org/10.1177/1077546316646906).
- [20] S. Xu and H. Peng, "Design, analysis, and experiments of preview path tracking control for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 48–58, Jan. 2020, doi: [10.1109/TITS.2019.2892926](https://doi.org/10.1109/TITS.2019.2892926).
- [21] H. Li, J. Luo, S. Yan, M. Zhu, Q. Hu, and Z. Liu, "Research on parking control of bus based on improved pure pursuit algorithms," in *Proc. 18th Int. Symp. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES)*, Nov. 2019, pp. 21–26, doi: [10.1109/DCABES48411.2019.00013](https://doi.org/10.1109/DCABES48411.2019.00013).
- [22] H. Ohta, N. Akai, E. Takeuchi, S. Kato, and M. Edahiro, "Pure pursuit revisited: Field testing of autonomous vehicles in urban areas," in *Proc. IEEE 4th Int. Conf. Cyber-Phys. Syst., Netw., Appl. (CPSNA)*, Oct. 2016, pp. 7–12, doi: [10.1109/CPSNA.2016.10](https://doi.org/10.1109/CPSNA.2016.10).
- [23] M.-W. Park, S.-W. Lee, and W.-Y. Han, "Development of lateral control system for autonomous vehicle based on adaptive pure pursuit algorithm," in *Proc. 14th Int. Conf. Control, Autom. Syst. (ICCAS)*, Oct. 2014, pp. 1443–1447, doi: [10.1109/ICCAS.2014.6987787](https://doi.org/10.1109/ICCAS.2014.6987787).
- [24] L. Yu, X. Yan, Z. Kuang, B. Chen, and Y. Zhao, "Driverless bus path tracking based on fuzzy pure pursuit control with a front axle reference," *Appl. Sci.*, vol. 10, no. 1, p. 230, Dec. 2019, doi: [10.3390/app10010230](https://doi.org/10.3390/app10010230).
- [25] C. G. Serna, A. Lombard, Y. Ruichek, and A. Abbas-Turki, "GPS-based curve estimation for an adaptive pure pursuit algorithm," in *Proc. MICAI*, 2016, pp. 497–511.
- [26] W. Zhang, J. Gai, Z. Zhang, L. Tang, Q. Liao, and Y. Ding, "Double-DQN based path smoothing and tracking control method for robotic vehicle navigation," *Comput. Electron. Agricult.*, vol. 166, Nov. 2019, Art. no. 104985, doi: [10.1016/j.compag.2019.104985](https://doi.org/10.1016/j.compag.2019.104985).
- [27] J. Chen, H. Zhu, L. Zhang, and Y. Sun, "Research on fuzzy control of path tracking for underwater vehicle based on genetic algorithm optimization," *Ocean Eng.*, vol. 156, pp. 217–223, May 2018, doi: [10.1016/j.oceaneng.2018.03.010](https://doi.org/10.1016/j.oceaneng.2018.03.010).
- [28] Á. Domina and V. Tihanyi, "Path following controller for autonomous vehicles," in *Proc. IEEE Int. Conf. Connected Vehicles Expo (ICCVE)*, Nov. 2019, pp. 1–5, doi: [10.1109/ICCVE45908.2019.8964960](https://doi.org/10.1109/ICCVE45908.2019.8964960).
- [29] M. Cibooglu, U. Karapinar, and M. T. Söylemez, "Hybrid controller approach for an autonomous ground vehicle path tracking problem," in *Proc. 25th Medit. Conf. Control Autom. (MED)*, Jul. 2017, pp. 583–588, doi: [10.1109/MED.2017.7984180](https://doi.org/10.1109/MED.2017.7984180).
- [30] R. Wang, Y. Li, J. Fan, T. Wang, and X. Chen, "A novel pure pursuit algorithm for autonomous vehicles based on salp swarm algorithm and velocity controller," *IEEE Access*, vol. 8, pp. 166525–166540, 2020, doi: [10.1109/ACCESS.2020.3023071](https://doi.org/10.1109/ACCESS.2020.3023071).
- [31] S. Qinpeng, W. Zhonghua, L. Meng, L. Bin, C. Jin, and T. Jiaxiang, "Path tracking control of wheeled mobile robot based on improved pure pursuit algorithm," in *Proc. Chin. Autom. Congr. (CAC)*, Nov. 2019, pp. 4239–4244, doi: [10.1109/CAC48633.2019.8997258](https://doi.org/10.1109/CAC48633.2019.8997258).
- [32] L. Xu, Y. Yang, Q. Chen, F. Fu, B. Yang, and L. Yao, "Path tracking of a 4WIS-4WID agricultural machinery based on variable look-ahead distance," *Appl. Sci.*, vol. 12, no. 17, p. 8651, Aug. 2022, doi: [10.3390/app12178651](https://doi.org/10.3390/app12178651).
- [33] P. Huang, Z. Zhang, X. Luo, J. Zhang, and P. Huang, "Path tracking control of a differential-drive tracked robot based on look-ahead distance," *IFAC-PapersOnLine*, vol. 51, no. 17, pp. 112–117, 2018, doi: [10.1016/j.ifacol.2018.08.072](https://doi.org/10.1016/j.ifacol.2018.08.072).
- [34] H. Shin, M.-J. Kim, S. Baek, C. D. Crane, and J. Kim, "Perpendicular parking path generation and optimal path tracking algorithm for auto-parking of trailers," *Int. J. Control, Autom. Syst.*, vol. 20, no. 9, pp. 3006–3018, Sep. 2022, doi: [10.1007/s12555-021-0268-9](https://doi.org/10.1007/s12555-021-0268-9).
- [35] A. E. B. Ibrahim, "Self-tuning look-ahead distance of pure-pursuit path-following control for autonomous vehicles using an automated curve information extraction method," *Int. J. Intell. Transp. Syst. Res.*, vol. 20, no. 3, pp. 709–719, Dec. 2022, doi: [10.1007/s13177-022-00319-z](https://doi.org/10.1007/s13177-022-00319-z).
- [36] C. Fruzzetti, S. Donnarumma, and M. Martelli, "Dynamic target chasing: Parameters and performance indicators assessment," *J. Mar. Sci. Technol.*, vol. 27, no. 1, pp. 712–729, Mar. 2022, doi: [10.1007/s00773-021-00865-3](https://doi.org/10.1007/s00773-021-00865-3).
- [37] A. Gautam, M. Singh, P. B. Sujit, and S. Saripalli, "Autonomous quadcopter landing on a moving target," *Sensors*, vol. 22, no. 3, p. 1116, Feb. 2022, doi: [10.3390/s22031116](https://doi.org/10.3390/s22031116).
- [38] J. M. Snider, "Automatic steering methods for autonomous automobile path tracking," *Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-09-08*, 2009.
- [39] S. Macenski, S. Singh, F. Martín, and J. Ginés, "Regulated pure pursuit for robot path tracking," *Auto. Robots*, vol. 47, no. 6, pp. 685–694, Aug. 2023, doi: [10.1007/s10514-023-10097-6](https://doi.org/10.1007/s10514-023-10097-6).
- [40] J. Wu, H. Ren, T. Lin, Y. Yao, Z. Fang, and C. Liu, "Autonomous path finding and obstacle avoidance method for unmanned construction machinery," *Electronics*, vol. 12, no. 9, p. 1998, Apr. 2023, doi: [10.3390/electronics12091998](https://doi.org/10.3390/electronics12091998).



YIBO CAO was born in Hengyang, Hunan, China, in 1971. He received the B.S. and M.S. degrees from the Department of Electrical and Mechanical Engineering, Xi'an University of Electronic Science and Technology, in 1994 and 1997, respectively, the Ph.D. degree from the School of Mechanical Engineering, South China University of Technology, Guangzhou, China, in 2007, and the Ph.D. degree from the Department of Precision Instruments, Tsinghua University, Beijing,

China, in 2009.

He joined the Software College, South China Normal University, Guangzhou, in 2009, where he is currently an Associate Professor. His research interests include AI artificial intelligence technology research, optical electromechanical integration technology, the Internet of Things technology, pattern recognition, and simultaneous localization and mapping (SLAM).



ZHENG DONG YANG was born in Nanchong, Sichuan, China, in 1999. He is currently pursuing the Graduate degree with the School of Software, South China Normal University. His current research interests include path planning, mobile robot, and autonomous robot navigation.



JIAHENG ZHAO was born in Xuchang, Henan, China, in 1998. He is currently pursuing the Graduate degree with the School of Software, South China Normal University. His current research interests include path tracking, mobile robot, and autonomous robot navigation.



HAIWEN ZHU received the B.E. degree from the College of Earth Science and Technology, Southwest Petroleum University, Chengdu, China, in 2020. He is currently pursuing the master's degree with the School of Software, South China Normal University. His research interests include autonomous robot navigation and environmental mapping.



JINGWEN FAN was born in Ganzhou, Jiangxi, China, in 1998. He is currently pursuing the Graduate degree with the School of Software, South China Normal University. His current research interests include computer vision, vision simultaneous localization, mapping, and indoor positioning.

...