MPC-based Unified Trajectory Planning and Tracking Control Approach for Automated Guided Vehicles*

Juncheng Li, Maopeng Ran, Han Wang, and Lihua Xie

***Abstract*— Autonomous navigation of Automated Guided Vehicles (AGVs) in manufacturing environment is an important part of industrial automation. This paper presents an MPC-based unified trajectory planning and tracking control approach for AGVs. Based on the model of the AGV, improved path planning and reference velocity planning techniques are developed. Then a model predictive controller is designed to track the generated trajectory. In addition, obstacle avoidance capability is also incorporated in the navigation framework. To evaluate the proposed method, simulations and experiments are conducted. The results show that the AGV can achieve high trajectory tracking accuracy with smooth movement.**

I. Introduction

In modern manufacturing, automated guided vehicles (AGVs) have been widely used for material handling tasks [1]. Traditionally, the navigation system for AGVs is based on magnetic tape. Nowadays material handling tasks and manufacturing environment are becoming more and more complex, and thus a more flexible and smarter navigation system is required. In most cases, LiDAR or vision based navigation approaches provide satisfactory solutions. However, manufacturing environments are very crowded with machines and human operators. In this case, for LiDAR or vision based navigation systems, motion planning and control become challenging [2].

For the motion planning and control problem, one approach is to solve an optimal control problem at each step and obtain optimal trajectory and corresponding control input simultaneously [3], [4]; see, for example, the dynamic window approach (DWA) [5] and the timed elastic band (TEB) approach [6]. However, due to the complexity of the problem, these approaches are often limited by heavy computation. Therefore, in real applications, the most common approach is to separate the problem into two parts: path planning part and path following part [7]. For the path planning part, various kinds of algorithms have been developed. For example, graph-based methods (such as A* and D* [8], [9]) and sampling-based methods (such as rapidly exploring random trees (RRT) [10]), etc. Employing any one of these methods, AGVs can obtain the planned path in real-time. For the path following part, a trajectory tracking controller is used to steer the vehicle. Many control approaches can be applied to AGVs, including PID control, sliding-mode control and fuzzy control, etc [11]. In this paper, our proposed method is based on model predictive control (MPC). MPC has been widely used to solve the path tracking problem of autonomous vehicles (e.g. [12]–[14]), since it can handle nonlinear kinematic models and constraints in a systematic way. By minimizing a designed objective function, an optimal sequence of future control inputs is generated and applied such that the robot states are optimally regulated according to the reference trajectory.

Generally, path planning part and path following part are completely independent. However, in this case, the planned path may not be a perfect path for an AGV to follow. The reasons are twofold. First, the planned path may be not smooth enough or even unrealistic for an AGV. Second, the kinematic model and the constraints of the vehicle are generally neglected in the path planning phase. As a result, in the path following phase, AGVs are not able to achieve high tracking performance. Therefore, it is important and meaningful to integrate path planning and tracking to improve the control performance.

In this paper, we focus on developing an MPC-based unified trajectory planning and tracking control strategy for AGVs to follow the reference path accurately and smoothly. The original path generated by path planner is improved using MPC-based approach. Then a trajectory is planned based on the new path using constrained reference velocity. The AGV can finally track this trajectory and achieve high control performance. Since the problems in both trajectory planning and tracing control part are solved using MPC, the proposed approach is named MPC-based unified planning and tracking approach. Besides, obstacle avoidance is also considered in the proposed framework. Based on an obstacle detection and path replanning mechanism, the AGV can avoid any static or dynamic obstacle which is in its way.

This paper is organized as follows. In Section II, the kinematic model of the AGV and some constraints are described. In Section III, our proposed MPC-based unified trajectory planning and tracking control approach is presented. Section IV shows the simulation and experiment settings and results. Finally, Section V gives the conclusion.

II. Modelling

In this study, a kinematic model is considered for AGVs. The configuration of the differential wheeled AGV platform is shown in Figure 1. Let $[x_c, y_c]^{\mathrm{T}}$ denote the central position of the robot, θ_c denote the orientation of the robot. Therefore, $z = [x_c, y_c, \theta_c]^{\mathrm{T}}$ forms the states of the robot. Let R_w and R_l denote the width and length of the AGV, $2l_w$ denote the distance between two wheels.

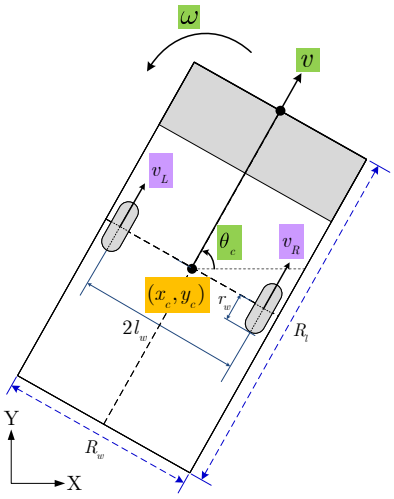


Fig. 1: Configuration of the AGV platform

The kinematic model of the AGV is described as:

$$\begin{cases} \dot{x}_c(t) = v(t)\cos(\theta_c(t)), \\ \dot{y}_c(t) = v(t)\sin(\theta_c(t)), \\ \dot{\theta}_c(t) = \omega(t), \end{cases} \tag{1}$$

where $v(t)$ is the linear velocity, $\omega(t)$ is the angular velocity. Let $u(t) = [v(t), w(t)]^{\mathrm{T}}$ denote the control input of the AGV, v^L and v^R denote the velocities of the left and right wheels of the robot, respectively. The velocities v^L and v^R are bounded by

$$\begin{cases} |v^L| \le v_{\max}, \\ |v^R| \le v_{\max}, \end{cases} \tag{2}$$

where $v_{\max}$ denotes the maximum linear velocity. The control inputs v and w can be formulated as the function of wheel velocities v^L and v^R as

$$\begin{cases} v = (v^L + v^R)/2, \\ w = (v^R - v^L)/2l_w. \end{cases} \tag{3}$$

According to Eq. (2) and (3), the control input is limited by

$$|v| + l_w |w| \le v_{\max}. \tag{4}$$

The inequality (4) defines a diamond shaped region [15]. The velocity of the robot should be within this region.

In real application, the kinematic model should be rewritten as the discrete-time form,

$$\begin{bmatrix} x_c(k+1) \\ y_c(k+1) \\ \theta_c(k+1) \end{bmatrix} = \begin{bmatrix} x_c(k) \\ y_c(k) \\ \theta_c(k) \end{bmatrix} + \begin{bmatrix} \cos(\theta_c(k)) & 0 \\ \sin(\theta_c(k)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(k) \\ w(k) \end{bmatrix} \Delta T. \tag{5}$$

The control input constraint needs to be satisfied at each time step, hence it is rewritten as:

$$|v(k)| + l_w |w(k)| \le v_{\max}. \tag{6}$$

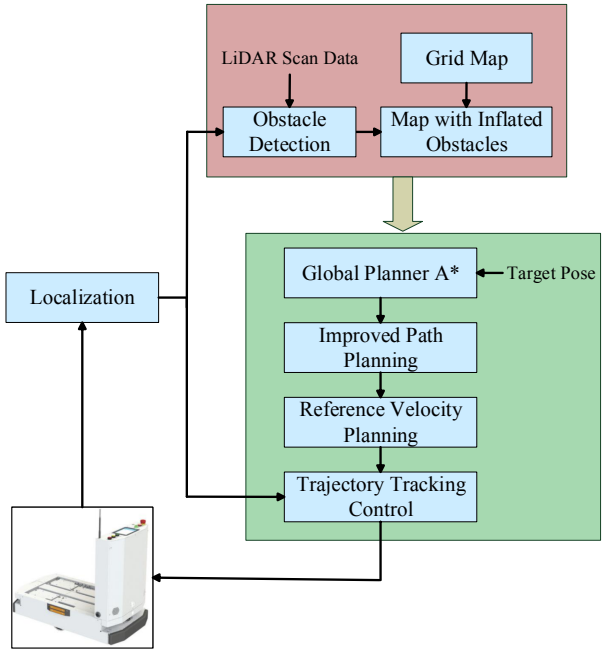


Fig. 2: The proposed navigation framework for an AGV

III. Unified Trajectory Planning and Tracking Control Algorithms

A. Navigation Framework

The whole framework of the proposed approach is shown in Figure 2. Firstly, based on the prebuilt grid map, a collision-free reference path r to the target position is generated by global planner. Then, an optimal path is obtained by considering the kinematics of the AGV. After that, reference velocity is planned by utilizing the control input constraint. Finally, a trajectory is generated and tracked by an MPC tracking controller. At each time step, the tracking error is defined as $\tilde{z}(k) = \|z(k) - r(k)\|$, where $z = [x_c, y_c, \theta_c]$ denotes the robot actual trajectory, $r = [x_r, y_r, \theta_r]$ denotes the reference trajectory . The controller generates the smooth control signal $u = [v, w]^{\mathrm{T}}$ which minimizes the tracking error $\tilde{z}$.

In the proposed navigation framework, the dynamic obstacle avoidance is also included. If any future collision is detected, the AGV can reactivate the trajectory planning and control method based on the new inflated map. In the following, the proposed unified trajectory planning and tracking control algorithm will be described in detail.

B. Improved Path Planning

In this study, a navigation map is prebuilt using simultaneous localization and mapping (SLAM). This offline map is a 2-D occupancy grid map. The obstacles on the map get inflated for collision avoidance according to the size of the robot and the localization inaccuracies. The static map with inflated layer is used for collision-free path planning.

A* is one of the most popular algorithms for global path planning. Based on the map, it can be utilized to generate a

shortest path from the current position to the goal position. The path consists of a sequence of waypoints. Each waypoint can be represented by $r = [x_r, y_r]^{\mathrm{T}}$. As mentioned before, this global path is not good enough. The smoothness of the path and nonholonomic motion constraint should be considered to improve the path.

Our proposed solution is an MPC-based path improvement. The key idea is that by simulating a simplified trajectory tracking process using MPC, an improved path can be extracted from the simulation results. Next, the details of the approach are presented.

At the beginning, from the planned path obtained by A* algorithm, a trajectory is created. The time interval between the reference waypoints is calculated as follows:

$$\begin{aligned}\Delta t(k) &= \frac{\|r(k) - r(k-1)\|}{v_{\max}} \\ &= \frac{\sqrt{\left(x_r(k) - x_r(k-1)\right)^2 + \left(y_r(k) - y_r(k-1)\right)^2}}{v_{\max}}\end{aligned} \tag{7}$$

In this trajectory, the linear velocity of the robot v is set to a constant value $v_{\max}$. Based on the time interval $\Delta t(k)$, the corresponding reference time of each waypoint $r(k)$ can be expressed as:

$$T(k) = \sum_{i=1}^{k} \Delta t(i) \tag{8}$$

In summary, the whole trajectory which consists of reference waypoints, time intervals and reference time, is shown in Table I.

TABLE I: Reference trajectory

Number k	1	2	...	N_r
Waypoints	$r(1)$	$r(2)$	...	$r(N_r)$
Time intervals	$\Delta t(1)$	$\Delta t(2)$	...	$\Delta t(N_r)$
Reference time	$T(1)$	$T(2)$	...	$T(N_r)$

Based on this trajectory, a simplified trajectory tracking process is simulated. In this part, MPC approach is used. A standard MPC trajectory tracking control problem at time step k is formulated as:

$$\min_{u_0, u_1, \ldots, u_{H-1}} \sum_{i=0}^{H-1} C(z_i, z_H, u_i, r) \tag{9a}$$

$$\text{s.t.} \quad z_0 = z(k), \tag{9b}$$

$$z_{i+1} = f(z_i, u_i), \qquad i \in [0, H-1] \tag{9c}$$

$$z_i \in \mathbb{Z}, u_i \in \mathbb{U}, \qquad \forall i \geq 0 \tag{9d}$$

where $C(\cdot)$ is the cost function which needs to be minimized within the finite horizon H. This optimization problem is solved under several constraints. Eq. (9b) represents the initial state constraint. At time step k, the initial state z_0 is given by $z(k)$. Eq. (9c) represents the kinematic constraint, where $f(z, u)$ is the model of the AGV which is described in (5). Eq. (9d) represents the state and control input constraints, where $\mathbb{X}$ and $\mathbb{U}$ are feasible sets for state z and control input u, respectively.

In the MPC framework, a cost function is defined to achieve good control performance, which considers both high tracking accuracy and movement smoothness. The cost function is formulated as:

$$\min_{u_0, \ldots, u_{H-1}} \sum_{i=1}^{H} \widetilde{z_i}^T Q \widetilde{z_i} + \sum_{i=0}^{H-1} (\widetilde{u_i}^T R \widetilde{u_i} + \Delta u_i{}^T S \Delta u_i) \tag{10}$$

where

$$\widetilde{z_i} = r(k+i) - z_i, \tag{11a}$$

$$\widetilde{u_i} = u_i^{ref} - u_i, \tag{11b}$$

$$\Delta u_i = u_{i+1} - u_i. \tag{11c}$$

The cost consists of three parts of penalty: $\widetilde{z_i}$ represents the deviation from the reference trajectory, $\widetilde{u_i}$ represents the difference between control inputs and reference velocity, Δu_i represents the variation of inputs. The matrices $Q \in \mathbb{R}^{3\times3}$, $R \in \mathbb{R}^{2\times2}$, $S \in \mathbb{R}^{2\times2}$ are positive definite weighting matrices of the three parts, respectively.

The formulated nonlinear optimal control problem is solved by using the interior point method iteratively [16]. The optimal predicted states z^* and control inputs u^* sequence are obtained:

$$\begin{aligned} z^* &= \left[\ z_1^*, \quad z_2^*, \quad \ldots, \quad z_H^*\ \right], \\ u^* &= \left[\ u_0^*, \quad u_1^*, \quad \ldots, \quad u_{H-1}^*\ \right]. \end{aligned} \tag{12}$$

Using the predictive states z^*, we can generate a new path to replace the original path. In this case, if the prediction horizon H is set to the length of the whole path N_r, the number of variables in the optimization problem will be huge and more computing time is needed. Therefore, we proposed a piecewise path generation approach. Firstly, a proper horizon length H is chosen. Based on the optimization result, half of the z^* is extracted to the new path, and the state $z^*_{\lfloor \frac{H}{2} \rfloor}$ is used to update the initial constraint (9b) which generates a new optimization problem. This process is repeated until the whole original path is replaced by the MPC predicted states. The proposed MPC-based improved path planning approach is summarized in Algorithm 1.

Algorithm 1 MPC-based improved path planning

Require: global planned path $r = [x_r, y_r]$

function SIMULATED PATH IMPROVEMENT (r)
  Generate a feasible trajectory $\{r, T\}$
  Initialize the state z_0
  Time step $k \leftarrow 0$
  while $k < N_r$ **do**
    Solve MPC trajectory tracking problem (9)
    Obtain z^* and u^*
    $\{r^*(k), \ldots, r^*(k + \lfloor \frac{H}{2} \rfloor - 1)\} \leftarrow \{z_1^*, \ldots z^*_{\lfloor \frac{H}{2} \rfloor}\}$
    $k \leftarrow k + \lfloor \frac{H}{2} \rfloor$
    $z_0 \leftarrow z^*_{\lfloor \frac{H}{2} \rfloor}$
  return $r^* = [x_r, y_r, \theta_r]$

C. Reference Velocity Planning

Given the planned path, the AGV can follow it with a constant speed. However, this is not a smart choice. For example, when an AGV runs into a corner or follows a curved line, it should slow down to guarantee safety. In this case, a reasonable reference velocity planning is necessary.

In the proposed approach, the control input constraint is utilized to conduct the velocity planning. According to Eq. (6), the linear velocity $v(k)$ can be derived based on the angular velocity $w(k)$. However, in the planning phase angular velocity $w(k)$ is not available, so the velocity cannot be planned directly. To overcome this problem, we use the following approach to estimate the reference velocity $v_{ref}(k)$. Firstly, θ_r can be extracted from the improved path $r = [x_r, y_r, \theta_r]$. Then a cubic polynomial fitting of $\theta_r(k)$ with respect to the index k is given by

$$\theta_r(k) = c_3k^3 + c_2k^2 + c_1k + c_0. \tag{13}$$

Now θ_r becomes a continuous and smooth function of k. However, since the length of path is mostly quite large, fitting a single polynomial curve will not represent the data precisely. Therefore, a piecewise polynomial fitting is implemented to solve the problem. Based on that piecewise polynomial function, the derivate of θ can be written as:

$$\frac{d}{dk}\theta_r(k) = 3c_3k^2 + 2c_2k + c_1. \tag{14}$$

In fact, $\frac{d}{dk}\theta_r(k)$ has a linear relationship with $w(k)$. That means we can estimate the $w(k)$ as:

$$\hat{w}(k) = c_v\frac{d}{dk}\theta_r(k), \tag{15}$$

where c_v is a constant. Now, the estimation of the reference velocity can be calculated using Eq. (6) and (15):

$$v_{ref}(k) = v_{\max} - l_w c_v \left|\frac{d}{dk}\theta_r(k)\right| \tag{16}$$

D. Tracking Control

Since the improved path r^* and reference velocity v_{ref} have been determined, a new trajectory can be generated. The method is presented in Eq. (7) and (8), but instead of using a constant velocity $v_{\max}$, now the calculated reference velocity is applied. The trajectory generation can be expressed as:

$$T^*(k) = \sum_{i=1}^{k} \frac{\|r^*(k) - r^*(k-1)\|}{v_{ref}(k)} \tag{17}$$

Base on the new trajectory, a new MPC trajectory tracking problem is built. It is similar to the problem described by Eq. (9) and (10). However, the reference trajectory r^*, T^*, reference velocity v_{ref} and weighting matrix Q, R, S in cost function are changed in the tracking control phase. Besides, the optimal control problem need to be solved at each time step. When the state of the AGV is updated by localization, the initial constraint of the MPC problem Eq. (9b) is updated. By solving (9), the optimal predictive states z^* and inputs u^* are obtained, but only the first element of the vector u^* is applied to the AGV as current control input:

$$u^* = \left[\ u_0^*,\quad u_1^*,\quad ...,\quad u_{H-1}^*\ \right], \tag{18}$$

$$u = u_0^*. \tag{19}$$

The optimization process is repeated to obtain the control input u until the target position is reached.

E. Dynamic Obstacle Avoidance

For AGVs in factories and warehouses, the dynamic obstacle avoidance capability in navigation system is also one of the most essential parts. Because the speed of AGVs may be very high, a fast and robust dynamic obstacle detection and avoidance algorithm is needed. In this work, the dynamic obstacle avoidance behavior is achieved by using LiDAR data. If a dynamic obstacle, for example, a person wants to pass in front of the AGV. Navigation system needs to judge whether it will intersect with the following trajectory of the robot. Let η denote the amount of laser scanned data, $l(i)$ denote the ith distance measurement, $p(i)$ denote the position of the ith reflection point in the global frame. Each time when laser scanned data comes in, it will be compared with the following N_p trajectory waypoints. If the minimum distance between laser scanned data and following N_p waypoints is less than half the width of the AGV, then the replanning phase will be trigged. The whole process of the dynamic obstacle detection is given by Algorithm 2.

Algorithm 2 Dynamic obstacle detection

Require: laser scan l, p, trajectory r and current time k

```
function OBSTACLE DETECTION(l, p, r, k)
    for i = 0 → η do
        if l(i) < 2 then
            for j = 1 → N_p do
                d = ‖p(i) − r(k + j)‖
                if d < R_w/2 then
                    return True
    return False
```

When sensor data comes in, the map of the environment is also updated. Therefore, detected obstacles will be added into the map in real time. When the replanning phase is trigged, based on the new map, the whole trajectory planning and tracking control algorithm will be reactivated. The whole framework is shown in Figure 2.

IV. Simulation and Experimental Results

A. Simulation

We implement the proposed unified trajectory planning and tracking control algorithm based on *Robot Operating System (ROS)*. The simulation environment is built using *stage* simulator. Map, sensors, robots, obstacles are all included in the simulation, so that the implementation can be completely tested and evaluated. An efficient nonlinear optimization solver named Interior Point OPTimizer (IPOPT) [16] is employed to solve the MPC problem, which makes

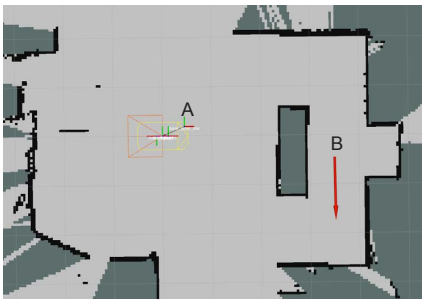


Fig. 3: Material handling task

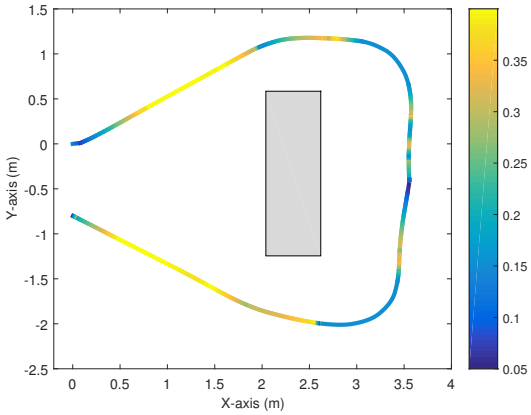
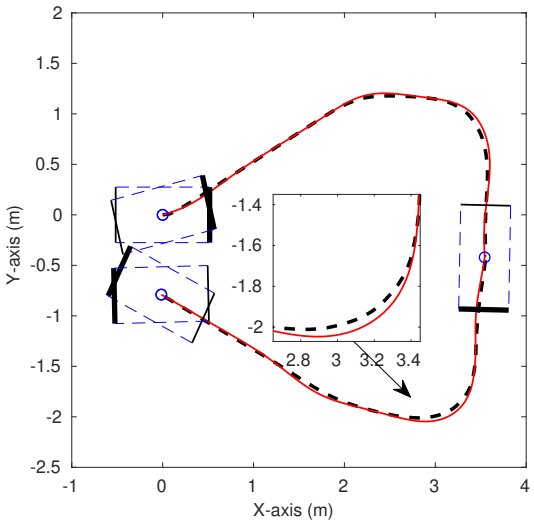


Fig. 4: Reference velocity planning result

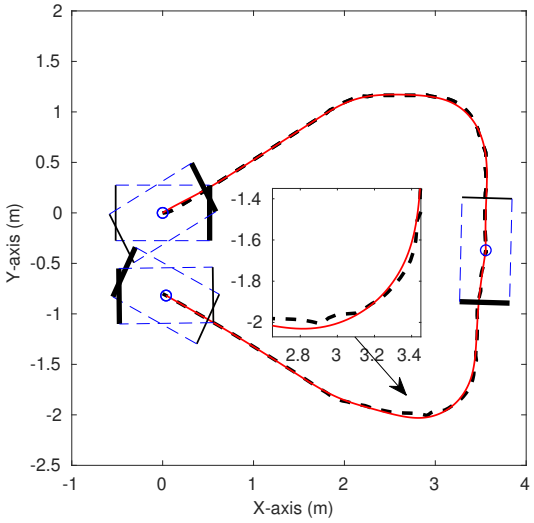
the control frequency of the AGV reach 20Hz. In the MPC-based design, parameter tuning is always a problem. In our approach, the simulation environment is used to tune the control parameters. More specifically, the cost function parameters, prediction horizon, solver settings and some other parameters in the path and reference velocity planning phases all need to be tuned in order to get perfect control performance. Some key parameters are selected as $H = 20$, $Q = \text{diag}\{1, 1, 0.01\}$, $R = \text{diag}\{0.5, 0.023\}$, $S = \text{diag}\{0.1, 0.05\}$, $l_w c_v = 4.5$, $N_p = 40$.

We consider the situation which is shown in Figure 3. The task for the AGV is to transport materials from A to B, and then come back. Figure 4 shows the planned reference velocity v_{ref} in simulation. According to the colorbar, lower speed is shown in blue and higher speed is shown in yellow. It is clear that at the beginning and the end of the path, or at the turning point, the velocity is planned to be slow. Besides, along straight lines, the velocity is planned close to maximum speed 0.4m/s.

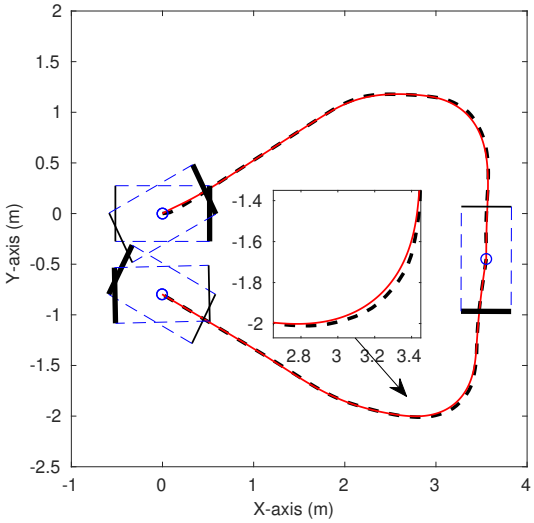
To evaluate the performance of the unified MPC-based trajectory planning and control approach, three different approaches are compared: (i) MPC-based improved trajectory planning + PID control, (ii) global plan (A*) + MPC and (iii) unified MPC planning + tracking. Figure 5 shows the trajectory tracking results from the three simulations. In Figure 5a, improved path planning and reference velocity planning are used to generate the trajectory, then a PID controller



(a) Improved trajectory planning + PID

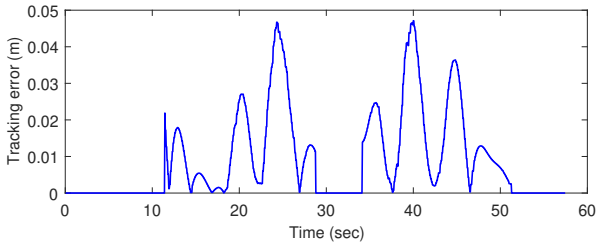


(b) Global plan (A*) + MPC

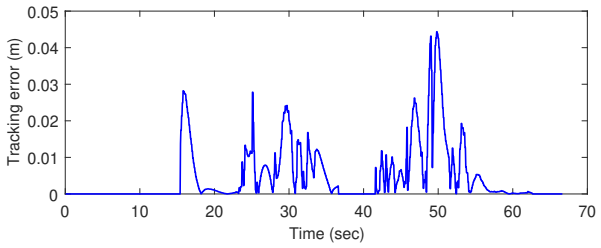


(c) Unified MPC planning + tracking

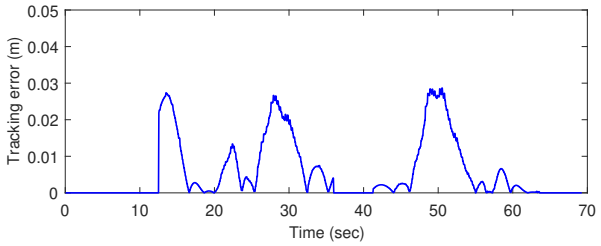
Fig. 5: Trajectory tracking (simulation)



(a) Improved trajectory planning + PID



(b) Global plan (A*) + MPC



(c) Unified MPC planning + tracking

Fig. 6: Trajectory tracking errors (simulation)

is designed to track it. In Figure 5b, the path is generated by A* algorithm, the reference velocity is set to the constant average velocity 0.25m/s, and the tracking controller is MPC. In Figure 5c, the proposed unified trajectory planning and tracking control approach is entirely applied.

In the three simulations, the task is completed successfully. For the planning part, compared with A* planned path, the MPC-based improved trajectory planning can smooth the path and provide reference velocity, see Figure 5b and 5c. For the trajectory tracking control part, compared with PID controller, the MPC controller performs better, see Figure 5a and 5c. The trajectory tracking errors are shown in Figure 6. It can be observed that larger error occurs when the tracking starts and when AGV goes around a corner. The maximum tracking error of the overall trajectory is 0.047m in Figure 6a, 0.044m in Figure 6b and 0.028m in Figure 6c. From the results, it can be concluded that the MPC-based unified trajectory planning and tracking control approach outperforms the other two approaches.

B. Experimental Results

We also run experiments in a real environment. The AGV is equipped with a 2D LiDAR for localization and obstacle detection, and a mini PC Intel® NUC. Adaptive Monte-Carlo Localization (AMCL) algorithm [17] is used to localize the AGV. Based on the localization result, our proposed

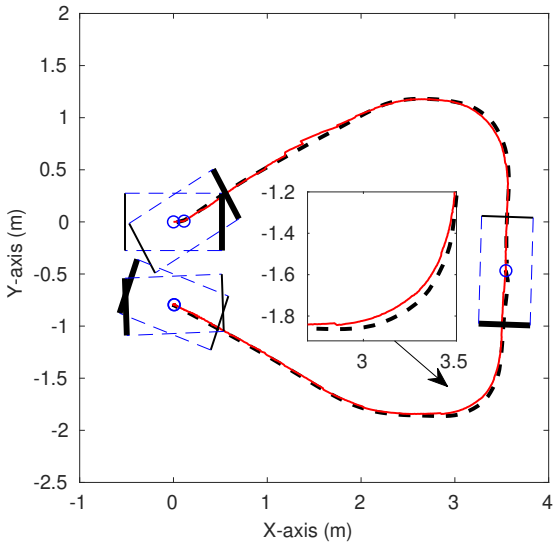


Fig. 7: Trajectory tracking (experiment)

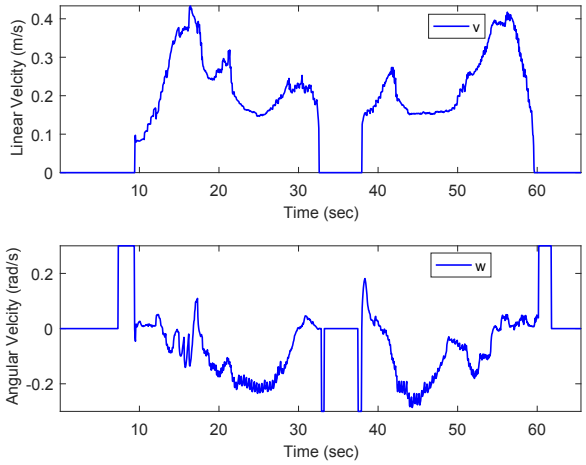


Fig. 8: Trajectory tracking control input (experiment)

unified trajectory planning and tracking control approach is fully implemented on the AGV. Considering the same scenario in simulation and experiment, we use the same system settings and the control parameters are selected as tuned in simulation.

In the experiment, only the proposed MPC-based unified trajectory planning and tracking approach is demonstrated. The reference velocity planning result is similar to the one shown in Figure 4. Figure 7 shows trajectory tracking results in the experiment. As expected, the robot can track the trajectory accurately. As previously mentioned, the robot needs to achieve smooth movement, that indicates the control signal should not vary too fast. The control input along the whole trajectory is shown in Figure 8. Both linear velocity v and angular velocity w vary at acceptable rates. The moving of AGV is acutally smooth enough in the experiment. Figure 9 shows the trajectory error during the experiment. Similar to the simulation, tracking error is larger at the beginning

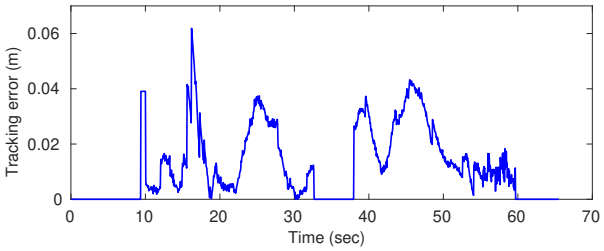
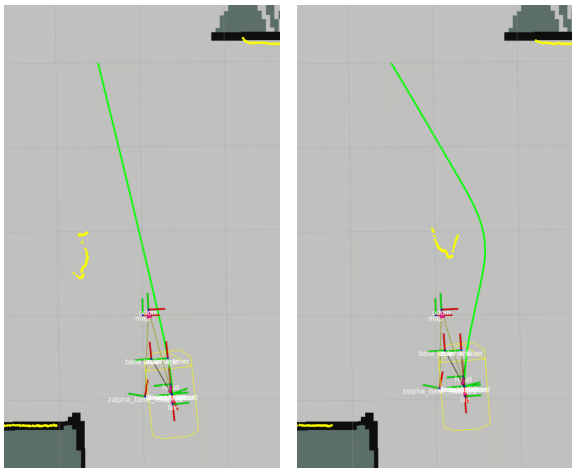


Fig. 9: Trajectory tracking error (experiment)



(a) Original path (b) New path

Fig. 10: Obstacle avoidance (experiment)

and the two corners. Besides, on the straight line there is another obvious tracking error. This is mainly caused by localization uncertainty. The maximum tracking error of the overall trajectory in the experiment is 0.062m.

Another two experiments are conducted to verify the collision avoidance ability. One is in static obstacle case and the other is in dynamic obstacle case. Figure 10 shows a simple scenario where a person passes in front of the AGV. At the beginning, the AGV follows a straight-line path, see Figure 10a. When an object is detected in its way, a new path is generated in real time, see Figure 10b. In the experiments, the AGV can not only track the trajectory accurately, but also avoid the obstacles in its way quickly. A video which shows the performance of the proposed approach in the test environment is available at https://youtu.be/vfNQ8kiD4I4.

V. Conclusion

This study has developed an MPC-based unified trajectory planning and tracking control strategy for an AGV to navigate itself with obstacle avoidance. Three comparative simulations were conducted in *ROS*, and the proposed approach was also tested in manufacturing environment. The results showed that the MPC-based unified trajectory planning and tracking control has advantages in improving the tracking accuracy and guaranteeing the movement smoothness. We believe that the proposed approach provides a novel and practical solution for the navigation of AGVs in manufacturing environment. Since the performance of MPC in general depends on the quality of the model, using machine learning techniques to obtain a more accurate model of the robot is considered for future research.

References

[1] L. Sabattini, M. Aikio, P. Beinschob, M. Boehning, E. Cardarelli, V. Digani, A. Krengel, M. Magnani, S. Mandici, F. Oleari *et al.*, "The pan-robots project: Advanced automated guided vehicle systems for industrial logistics," *IEEE Robotics & Automation Magazine*, vol. 25, no. 1, pp. 55–64, 2018.

[2] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, "Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 225–254, 2017.

[3] T. Mercy, R. Van Parys, and G. Pipeleers, "Spline-based motion planning for autonomous guided vehicles in a dynamic environment," *IEEE Transactions on Control Systems Technology*, 2017.

[4] H. Febbo, J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "Moving obstacle avoidance for large, high-speed autonomous ground vehicles," in *American Control Conference (ACC)*, 2017, pp. 5568–5573.

[5] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 1986–1991.

[6] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5681–5686.

[7] F. Debrouwere, "Optimal robot path following fast solution methods for practical non-convex applications," Ph.D. dissertation, KU LEUVEN, 2015.

[8] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[9] A. Stentz *et al.*, "The focussed d^* algorithm for real-time replanning," in *IJCAI*, 1995, pp. 1652–1659.

[10] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[11] J. Villagra and D. Herrero-Pérez, "A comparison of control techniques for robust docking maneuvers of an agv," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 4, pp. 1116–1123, 2012.

[12] X. Li, Z. Sun, D. Cao, D. Liu, and H. He, "Development of a new integrated local trajectory planning and tracking control framework for autonomous ground vehicles," *Mechanical Systems and Signal Processing*, vol. 87, pp. 118–137, 2017.

[13] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[14] S. J. Jeon, C. M. Kang, S.-H. Lee, and C. C. Chung, "Gps waypoint fitting and tracking using model predictive control," in *Intelligent Vehicles Symposium (IV)*, 2015, pp. 298–303.

[15] Z. Sun and Y. Xia, "Receding horizon tracking control of unicycle-type robots based on virtual structure," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 17, pp. 3900–3918, 2016.

[16] A. Wachter, "An interior point algorithm for large-scale nonlinear optimization with applications in process engineering." Ph.D. dissertation, Carnegie Mellon University, 2003.

[17] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.