Emedia: ISSN 1529-7306 Emedia/Online Inc

Trajectory Tracking Control for a Container Transferring Mobile Robot Based on an Improved NMPC

Xuehong Zhu[1*],Fei Liu[1],Lizhen Jia[2],Ning Fan[1]

1.Tianjin Sino-German University of Applied Sciences, School of Mechanical Engineering, 300350

2.Civil Aviation University of China, School of Transportation Science and Engineering, 300300

***Corresponding Author:Xuehong Zhu*

Email: zhuxuehong@tsguas.edu.cn

Abstract. Container Transferring Mobile Robots are widely used in container handling, narrow-aisle transfer, workstation delivery, and automated loading and unloading tasks. These robots usually operate in environments with dense shelving, narrow passages, and clearly defined boundaries. During practical operation, they also frequently perform start-stop motions, pickup tasks, and delivery tasks. These factors make trajectory tracking more challenging, especially when the robot is affected by motion-state switching, reference-path curvature variation, and limited aisle space. In such conditions, attitude fluctuations and control instability may occur, increasing the risk of collision between the robot body and the aisle boundaries.To address this problem, this paper proposes a constrained trajectory-tracking control method for a Container Transferring Mobile Robot based on a discretized model derived from Lagrangian dynamics. First, the dynamic model of the robot is established using the Lagrange method, and a prediction model is obtained through discretization. Then, considering the vehicle width, aisle width, and safety-margin requirements, a virtual safety corridor is constructed to describe the allowable motion range of the robot body. This corridor is introduced into the trajectory-tracking optimization problem as a geometric boundary constraint, so that the operating region of the robot can be limited during prediction and control.The hard-constraint formulation can prevent the robot from crossing the aisle boundary when the optimization problem remains feasible. However, in cases with large initial deviations, rapid changes in reference-path curvature, or further restricted aisle space, the feasible region of the optimization problem may shrink significantly. This may lead to optimization infeasibility and affect the continuous operation of the controller. To improve the applicability of the controller under such conditions, nonnegative slack variables are introduced to soften the geometric boundary constraints, and a linear penalty term is added to the cost function. In addition, an adaptive penalty mechanism based on the degree of constraint violation is designed to enhance constraint recovery near the boundary.Simulation and hardware experimental results show that the proposed method maintains optimization feasibility when the initial lateral deviation exceeds the prescribed corridor threshold. Compared with the standard NMPC and the fixed-weight soft-constrained NMPC, the proposed controller reduces the corridor-violation duration and improves boundary-recovery performance, while maintaining acceptable tracking

accuracy and real-time computational performance.

Keywords. differential-drive mobile robot; container transferring mobile robot; trajectory tracking control; model predictive control; constrained control.

Introduction

With rapid developments in intelligent manufacturing, container transferring mobile robots are increasingly used in scenarios such as material handling, high-density narrow-aisle transfer, and flexible production. Unlike navigation tasks in open environments, these robots usually work in dense and narrow industrial spaces with clear physical boundaries. They are required not only to track reference trajectories accurately, but also to avoid collisions with shelves, containers, and surrounding equipment. Klančar et al.[1]pointed out that mobile robots in industrial environments must satisfy both path-tracking and safe-operation requirements within constrained spaces. Achirei et al.[2]showed that trajectory control in logistics environments needs to consider environmental constraints and online control performance at the same time. Lucet et al.[3]studied autonomous forklift navigation in a cluttered logistics factory and emphasized the importance of navigation corridors and tracking-control constraints for safe operation in space-limited logistics environments. Therefore, accurate and safe trajectory tracking in confined spaces remains an important control problem for container transferring mobile robots.

For mobile robot trajectory tracking, controller design usually starts from either a kinematic model or a dynamic model. In scenarios with rapid input changes and non-negligible dynamic responses, a purely kinematic model is often insufficient to describe the actual motion evolution of the robot. Dynamic models based on physical mechanisms are therefore more suitable for high-precision trajectory-tracking control. Zhang et al.[4]established a robot chassis dynamic model using the Lagrange equations and considered factors such as the rotational inertia of the driving wheels, which improved the description of the robot's dynamic behavior. Moritz et al.[5]developed a differential-drive robot model that includes dynamic modeling, kinematic modeling, linearization, and stabilizing control design, providing a physical basis for constrained controller design. Liu et al.[6]pointed out that the control design of wheeled mobile robots should consider both kinematic and dynamic characteristics to handle nonlinearity, underactuation, and constraints.

Model predictive control has become an important method for mobile robot trajectory tracking because it can handle system dynamics and constraints within a receding-horizon framework. Köhler et al.[7]noted that MPC is well suited for tracking problems and constrained nonlinear systems. Wang et al.[8]applied nonlinear model predictive control to pose tracking of skid-steered mobile robots and verified its effectiveness for complex dynamic systems. Peng et al.[9]combined a nonlinear disturbance observer with MPC to improve the control performance of wheeled mobile robots under complex operating conditions. Tang et al.[10]pointed out that the kinematic and dynamic characteristics of mobile robots make accurate modeling and effective control a persistent challenge. Tang et al.[11]further proposed a geometric MPC method for wheeled mobile robot trajectory

tracking, showing that the description of robot configuration and tracking error has an important influence on MPC tracking performance.

However, in narrow-corridor container-handling scenarios, a dynamic model and a standard NMPC framework alone are still not sufficient. In many existing studies, spatial boundaries or safety conditions are directly imposed on the optimization problem as hard constraints. Fnadi et al.[12]integrated steering and sideslip constraints into a constrained model predictive control problem. Nam et al.[13]reinforced safety boundaries during path tracking through independent constraints. Jian et al.[14]combined dynamic control barrier functions with MPC for safety-critical obstacle avoidance of mobile robots, further showing the importance of introducing safety constraints into predictive control. These methods can strictly enforce boundary and physical constraints when the optimization problem is feasible. However, when the robot has a large initial deviation, the trajectory curvature changes rapidly, or the available corridor space is limited, the feasible set within the prediction horizon may shrink significantly. In severe cases, the optimization problem may become infeasible. For container transferring mobile robots, lateral deviations may occur during pickup, transfer, placement, and attitude adjustment. If the geometric boundary is always treated as a hard constraint, the controller can provide strong boundary restriction, but its feasibility may be weakened. Khan et al.[15]also pointed out that strict constraints and complex operating conditions can affect the online continuity and practical deployment of predictive control.

To reduce the infeasibility caused by hard constraints, soft-constrained MPC introduces slack variables into the constraints so that the feasible set of the optimization problem can be enlarged. Wabersich et al.[16]proposed a soft-constrained MPC formulation and showed that the control problem can remain tractable even when constraint-violating trajectories exist. Gracia et al.[17]further discussed the implementation of soft-constrained MPC for tracking problems and emphasized its role in maintaining optimization solvability. Liu et al.[18]developed an optimization-based local planner for nonholonomic autonomous mobile robots in semi-structured environments, where safety and feasibility were both considered in constrained navigation. These studies suggest that soft constraints can alleviate the limitations of hard-constrained methods, especially the problems of feasible-set shrinkage and optimization infeasibility.

Softened constraints have previously been introduced into MPC-based trajectory tracking of wheeled mobile robots. For example, Yang et al.[23] applied MPC with softening constraints to a nonholonomic wheeled mobile robot to avoid infeasibility caused by strict constraints. Stochastic MPC with soft probability constraints has also been investigated for wheeled mobile robot trajectory tracking[24]. However, the formulation of a body-size-dependent virtual corridor and the recovery behavior from initially infeasible corridor conditions have received comparatively limited attention in torque-level NMPC for container transferring mobile robots.

Nevertheless, for the virtual-corridor recovery problem considered in this paper, a conventional fixed quadratic slack penalty may still have limitations. This design can reduce constraint violation, but when the robot moves near the boundary, the optimizer may still retain a small nonzero slack value to improve local tracking accuracy or input smoothness. As a result, slight boundary violations may remain during the steady-state phase. For container transferring mobile robots operating in narrow corridors, such small

but persistent violations weaken the practical meaning of geometric safety constraints. Therefore, it is necessary to further strengthen the compression of slack variables while preserving optimization feasibility, so that the controller can recover the original boundary constraint as much as possible after the robot returns to the feasible region. Sun et al.[19]proposed a multi-constraint MPC method for forklift trajectory control, indicating that stability and safety constraints need to be considered together in industrial vehicle applications. Magalhães and Pimenta[20]studied distributed MPC for safety-critical obstacle avoidance in multi-robot systems, further showing that online solvability and safety constraints remain important issues in complex constrained robotic systems.

This paper proposes an improved soft-constrained nonlinear model predictive control method for trajectory tracking of container transferring mobile robots in narrow-corridor environments. The method is based on a discretized Lagrangian dynamic model. First, the dynamic model of the container transferring mobile robot is established using the Lagrange method and then discretized to construct the prediction model for receding-horizon optimization. Next, the vehicle width, corridor width, and safety margin are used to convert the lateral tracking error into a virtual safety-corridor constraint, which is embedded into the NMPC optimization problem as a geometric safety boundary. Nonnegative slack variables are then introduced to soften the original geometric hard constraints, improving controller feasibility under large initial deviations and complex trajectory curvatures. To avoid insufficient slack compression near the boundary, a linear penalty term is added to the slack-variable cost function, and an adaptive penalty mechanism is designed according to the degree of constraint violation. In this way, the proposed method limits excessive slack-variable growth and helps the controller recover the boundary constraint after the robot re-enters the feasible region of the original hard-constrained problem. While retaining the feasibility advantage of soft constraints, the proposed method is intended to improve boundary-recovery performance and reduce corridor-violation exposure, rather than to provide an absolute geometric-safety guarantee.

1 Modeling and Analysis of a Differential-Drive Container Transferring Mobile Robot

1.1 Kinematic Modeling

The differential-drive container transferring mobile robot is simplified as a differential-drive mobile platform. As shown in Fig. 1.1, OXY is the global coordinate system, and $O_PX_PY_P$ is the body-fixed coordinate system attached to the robot chassis. Under the assumptions of pure rolling and no lateral slipping, the kinematic relationship of the robot can be established.

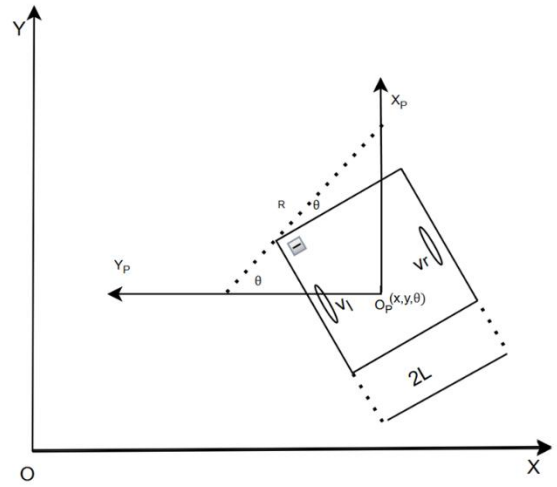


Fig. 1.1 Motion Model of the Container Transferring Mobile Robot

Let v_l and v_r be the linear velocities of the left and right driving wheels, respectively. The track width is defined as 2L, and the half-track is L. Based on rigid body kinematics, the linear velocity v and angular velocity v in the robot body coordinate system can be derived from the wheel velocities as:

$$v = \frac{v_l + v_r}{2} \tag{1.1}$$

$$\omega = \frac{v_r - v_l}{2L} \tag{1.2}$$

Then, the kinematic model in the global coordinate system can be written as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta/2 & \cos\theta/2 \\ \sin\theta/2 & \sin\theta/2 \\ \frac{1}{2L} & -\frac{1}{2L} \end{bmatrix} \begin{bmatrix} v_l \\ v_r \end{bmatrix} \tag{1.3}$$

1.2 Lagrangian dynamics modeling

Differential drive mobile robots belong to non-holonomic constraint systems. Based on the assumption that the 'lateral velocity of the vehicle body is zero', its non-holonomic constraint can be expressed as:

$$-\sin\theta\dot{x} + \cos\theta\dot{y} = 0 \tag{1.4}$$

Since the generalized coordinates of the robot are defined as $\mathrm{q} = [\mathrm{x}, \mathrm{y}, \theta]^{\mathrm{T}}$,the constraint can be written in Pfaffian form.

$$M(q)\dot{q} = 0, M(q) = [-sin\theta \quad cos\theta \quad 0] \tag{1.5}$$

The dynamic equation with Lagrange multipliers can be expressed as Equation (1.6).

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + M^T(q)\lambda = E\tau \tag{1.6}$$

Where $\mathrm{D(q)}$ is the inertia matrix, $\mathrm{C(q,\dot{q})\dot{q}}$ denotes the Coriolis and centrifugal terms,

$g(q)$ is the generalized gravitational force, λ is the Lagrange multiplier for the constraint forces, $E(q)$ is the input transformation matrix, and $\tau = [\tau_r, \tau_l]^T$ denotes the left and right wheel torques.

For low-speed trajectory tracking on a horizontal plane, the potential energy is independent of the generalized coordinates; therefore, the generalized gravitational force is zero $g(q) = 0$.

Furthermore, in low-speed trajectory tracking scenarios, the Coriolis and centrifugal terms, along with unmodeled dynamics such as rolling resistance and friction variations, are relatively small compared to the dominant terms and can be lumped into equivalent disturbances. Under these approximations, Equation (1.6) can be simplified to:

$$D(q)\ddot{q} + M^T(q)\lambda = E\tau \tag{1.7}$$

To eliminate the constraint force terms from the dynamic equations, a null-space basis matrix satisfying $M(q)B(q)=0$ is introduced Equation (1.8).

$$B(q) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}, \dot{q} = B(q)u, u = \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{1.8}$$

Vector $u = [v, \omega]^T$ and the time derivative of the posture $\dot{q}$.

Pre-multiplying both sides of Equation (1.7) by $B(q)^T$ yields:

$$B(q)^T D(q)\ddot{q} + B(q)^T M^T(q)\lambda = B(q)^T E\tau \tag{1.9}$$

By utilizing $M(q)B(q) = 0 \Rightarrow B(q)^T M(q)^T = 0$,the constraint force terms can be eliminated, yielding:

$$B(q)^T D(q)\ddot{q} = B(q)^T E\tau \tag{1.10}$$

From Equation (1.8), we obtain:

$$\ddot{q} = B(q)\dot{u} + \dot{B}(q)u \tag{1.11}$$

Substituting into Equation (1.10) yields:

$$B(q)^T D(q)\left(B(q)\dot{u} + \dot{B}(q)u\right) = B(q)^T E\tau \tag{1.12}$$

Where $\dot{B}(q)u$ represents the velocity product terms associated with $\dot{\theta}$. Considering the low-speed conditions and a small sampling period, these terms can be approximated and lumped into the equivalent disturbances, yielding Equation (1.13).

$$B(q)^T D(q)B(q)\dot{u} = B(q)^T E\tau \tag{1.13}$$

Defining the inertia matrix as $D(q) = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix}$ where m is the mass of the vehicle body and I is the moment of inertia. By mapping the driving torques to the longitudinal driving force and the yaw moment, the equivalent input mapping can be expressed as: $B(q)^T E(q) = \frac{1}{r}\begin{bmatrix} 1 & 1 \\ L & -L \end{bmatrix}$

Where r is the wheel radius, and L is the half-track width. Substituting these into Equation (1.13) yields the dynamic equation of the differential-drive mobile robot as

Equation (1.14)

$$\begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = \frac{1}{r}\begin{bmatrix} 1 & 1 \\ L & -L \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{1.14}$$

can be equivalently expressed as:

$$\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} \frac{1}{mr} & \frac{1}{mr} \\ \frac{L}{Ir} & -\frac{L}{Ir} \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{1.15}$$

Alternatively, the required driving torques for the mobile robot can be derived from Equation (1.15), yielding: $\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} = \frac{r}{2}\begin{bmatrix} m & \frac{I}{L} \\ m & -\frac{I}{L} \end{bmatrix}\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix}$(1.16)

1.3 Discretization of the Dynamic Model

For digital implementation, the continuous-time model in Eq. (1.15) is discretized with a sampling period T_s.

$$t_k = k \cdot T_s, k = 0,1,2, \ldots$$

At discrete time instant k (corresponding to continuous time $t = t_k$), let the system state vector and the control input vector be defined respectively as: $\mathrm{X}(k) = [x_k, y_k, \theta_k, v_k, \omega_k]^T$,$\tau(k) = [\tau_{r,k}, \tau_{l,k}]^T$.

Where (x_k, y_k, θ_k) denotes the posture of the robot in the world coordinate frame. v_k and ω_k are the linear and angular velocities of the vehicle body, respectively, and $\tau_{r,k}, \tau_{l,k}$ represent the output driving torques of the right and left motors."

Given the high sampling frequency of the system, it is assumed that the rate of change of the state remains approximately constant within a single sampling interval. Consequently, this paper employs the first-order Forward Euler method to approximately discretize the continuous-time dynamic model. Based on the finite-difference definition of the derivative, the time derivative of the state variables can be expressed as Equation (1.17)

$$\dot{\mathrm{X}}(t) \approx \frac{\mathrm{X}(k+1)-\mathrm{X}(k)}{T_s} \tag{1.17}$$

Substituting the continuous-time nonlinear state equation $\dot{\mathrm{X}} = f(\mathrm{X}, \tau)$ into Equation (1.17) and rearranging the terms yields the discrete-time recursive state model:

$$\mathrm{X}(k+1) = \mathrm{X}(k) + T_s \cdot f\big(\mathrm{X}(k), \tau(k)\big) \triangleq f_d(X_k, \tau_k) \tag{1.18}$$

By substituting the kinematic model in Equation (1.3) and the dynamic model in Equation (1.15) into Equation (1.18) and expanding the formulation, the system of discrete-time nonlinear state transition equations for the differential-drive mobile robot is obtained as follows:

$$\begin{cases} x_{k+1} = x_k + T_s \cdot v_k \cos\theta_k \\ y_{k+1} = y_k + T_s \cdot v_k \sin\theta_k \\ \theta_{k+1} = \theta_k + T_s \cdot \omega_k \\ v_{k+1} = v_k + T_s \cdot \left[\frac{1}{mr}(\tau_{r,k} + \tau_{l,k})\right] \\ \omega_{k+1} = \omega_k + T_s \cdot \left[\frac{L}{Ir}(\tau_{r,k} - \tau_{l,k})\right] \end{cases} \tag{1.19}$$

2 Controller Design

2.1 Nonlinear Model Predictive Controller with Hard Constraints

The discrete model in Eq. (1.19) is used for prediction.

$$\mathrm{X}_{k+1} = \begin{bmatrix} x_k + v_k \cos\theta_k T_s \\ y_k + v_k \sin\theta_k T_s \\ \theta_k + \omega_k T_s \\ v_k + \frac{1}{mr}(\tau_{r,k} + \tau_{l,k})T_s \\ \omega_k + \frac{L}{Ir}(\tau_{r,k} - \tau_{l,k})T_s \end{bmatrix} \tag{2.1}$$

where $X_{k+i|k}$ denotes the predicted state at step $k+i$ made at time k based on the current measurement $X_{\text{measured}}(k)$, and $\tau_{k+i|k}$ denotes the corresponding predicted control input. The initial-state constraint is given by

$$\mathrm{X}_{k|k} = \mathrm{X}_{measured}(k) \tag{2.2}$$

Let N_p denote the prediction horizon, $\left\{X_{k+i}^{ref}\right\}_{i=0}^{N_p}$ the reference trajectory, and $\left\{\tau_{k+i}^{ref}\right\}_{i=0}^{N_p-1}$ the reference input sequence. The standard NMPC employs a quadratic cost function with fixed weighting, denoted by J_{hard}, to minimize the state-tracking error and the control-input deviation:

$$\begin{aligned} J_{hard}(X,\tau) = \sum_{i=0}^{N_p-1} & (\|X_{k+i|k} - X_{k+i|k}^{ref}\|_Q^2 + \|\tau_{k+i|k} - \tau_{k+i|k}^{ref}\|_R^2) \\ & + \|X_{k+N_p|k} - X_{k+N_p|k}^{ref}\|_{Q_T}^2 \end{aligned} \tag{2.3}$$

Here $\|e\|_Q^2 = e^T Q e$ represents the weighted Euclidean norm. Q denotes the state weighting matrix, which is used to regulate trajectory tracking accuracy. Q_T denotes the terminal weighting matrix used to penalize the terminal tracking error. Under the standard terminal conditions stated in Section 2.3, it also contributes to the conditional closed-loop stability analysis. τ^{ref} is the feedforward torque obtained from inverse dynamics. The term $\|\tau - \tau^{ref}\|_R^2$ penalizes the deviation of the actual torque from the reference torque, where R is the control weighting matrix.

First, the physical constraints of the actuators must be considered. Limited by the rated power of the brushless DC motors and their drivers, the output torque of the right and left driving wheels, $\tau_k = [\tau_{r,k}, \tau_{l,k}]^T$,must be bounded within the saturation limit τ_{max} to prevent overcurrent damage to the hardware. The mathematical expression is given in Eq. (2.4):

$$\tau_{min} \leq \tau_{r,k+i|k} \leq \tau_{max} \ \ \tau_{min} \leq \tau_{l,k+i|k} \leq \tau_{max}, i = 0,1 \ldots, N_p - 1 \tag{2.4}$$

In addition to actuator constraints, geometric constraints are imposed to define the admissible operating corridor of the robot under complex operating conditions. To this end, this paper proposes a dynamic safety-envelope construction method based on the reference trajectory: in the global trajectory-tracking task, a virtual safety corridor with a total width of W_{aisle} is dynamically generated along the normal direction of the predefined reference trajectory. By combining the physical width of the robot body, W_{robot}, with the unilateral safety margin d_{margin} reserved to account for unmodeled errors, the maximum allowable lateral deviation of the geometric center of the robot from the reference trajectory, namely the half-width of the safety corridor, d_{safe}, can be strictly defined as:

$$d_{safe} = W_{aisle}/2 - W_{robot}/2 - d_{margin} \tag{2.5}$$

The resulting virtual corridor is illustrated in Fig. 2.1.

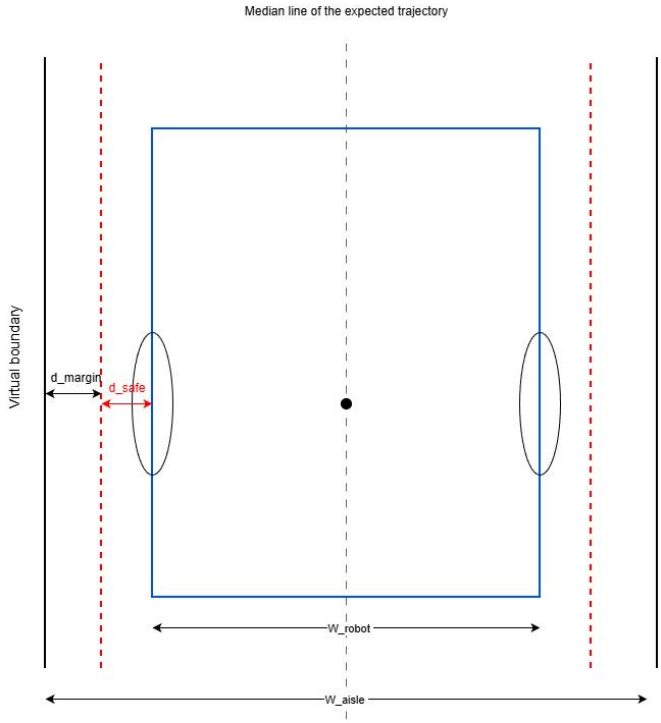


Fig. 2.1. Schematic illustration of the virtual boundary

To prevent collisions between the robot body and the shelves, the lateral tracking error of the robot, e_{lat}, is strictly constrained within the half-width of the safety corridor, d_{safe}. By approximating the reference heading angle as the tangential angle of the reference trajectory, we have:

$$\theta_{k+i}^{ref} = \text{atan2}\left(y_{k+i+1}^{ref} - y_{k+i}^{ref}, x_{k+i+1}^{ref} - x_{k+i}^{ref}\right) \tag{2.6}$$

The lateral error is defined as:

$$e_{lat,k+i|k} = -\left(x_{k+i|k} - x_{k+i}^{ref}\right) sin\,\theta_{k+i}^{ref} + \left(y_{k+i|k} - y_{k+i}^{ref}\right) cos\,\theta_{k+i}^{ref} \tag{2.7}$$

and, on this basis, the following hard constraint is imposed:

$$\left|e_{lat,k+i|k}\right| \le d_{safe}, i = 0,1 \ldots, N_p - 1 \tag{2.8}$$

In summary, the hard-constrained NMPC trajectory-tracking problem at time step k can be formulated as the following nonlinear programming problem:

$$\begin{aligned} &\min_{\{\tau_{k+i|k}\}_{i=0}^{N_p-1}} J_{hard}(X,\tau) \\ &s.t. X_{k|k} = X_{measured}(k), \\ &X_{k+i+1|k} = f_d(X_{k+i|k}, \tau_{k+i|k}), i = 0, \ldots, N_p - 1, \\ &\tau_{min} \le \tau_{r,k+i|k} \le \tau_{max}, i = 0, \ldots, N_p - 1, \\ &\tau_{min} \le \tau_{l,k+i|k} \le \tau_{max}, i = 0, \ldots, N_p - 1, \\ &|e_{lat,k+i|k}| \le d_{safe}, i = 0, \ldots, N_p - 1. \end{aligned} \tag{2.9}$$

By solving this optimization problem, the controller obtains the optimal control sequence $\tau^* = \tau_{k|k}^*, \ldots, \tau_{k+N_p-1|k}^*$ Subsequently, only the first control input is applied to the system, thereby implementing receding-horizon optimal control.

2.2 Nonlinear Model Predictive Controller with Soft Constraints

During narrow-corridor trajectory tracking, the robot may temporarily lie outside the virtual corridor because of initial localization errors, path switching, or transient disturbances. If Eq. (2.8) is retained as an unrelaxed hard constraint, the optimization problem in Eq. (2.9) may have no feasible solution. To prevent the geometric corridor constraint from directly causing optimization infeasibility, only the corridor constraint is softened, whereas the actuator torque constraints remain hard. The resulting control procedure is shown in Fig. 2.2.

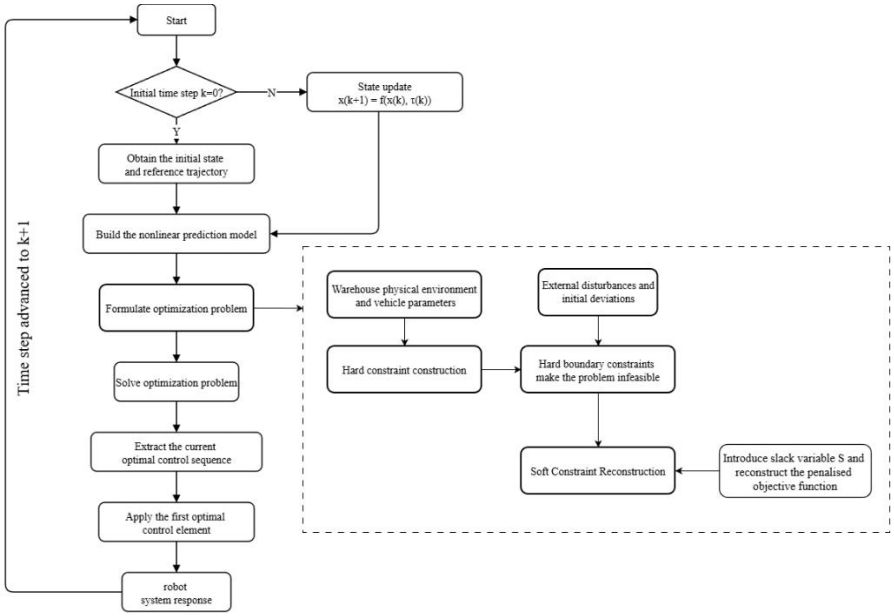


Fig. 2.2. Flowchart of the controller

A nonnegative slack-variable sequence $S = [S_0, S_1, ..., S_{N_p-1}]^T (S_k \geq 0)$ is introduced to soften the geometric hard constraint in Eq. (2.8):

$$\left|e_{lat,k+i|k}\right| \leq d_{safe} + S_{k+i} \tag{2.10}$$

Equation (2.10) allows the optimizer to maintain feasibility by selecting $S_{k+i} > 0$ when the original hard constraint cannot be temporarily satisfied. When $S_{k+i} > 0$, the predicted lateral error is allowed to exceed the original virtual-corridor boundary. Therefore, the soft constraint improves optimization feasibility but does not strictly guarantee that the robot always remains within d_{safe}. In this paper, the virtual corridor is treated as a geometric constraint within the NMPC formulation rather than as an independent strict safety guarantee.

$$\rho(S_{k+i}) = \rho_{base}(1 + \gamma S_{k+i}) \tag{2.11}$$

In this paper, "adaptive" indicates that the penalty weight varies explicitly with the instantaneous slack variable. It does not refer to an independent parameter-estimation or learning law. When the slack variable increases, the corresponding penalty weight increases automatically, thereby imposing a stronger penalty on large corridor violations.

In Eq. (2.11), ρ_{base} denotes the base quadratic penalty coefficient, and γ denotes the adaptive barrier factor.

Combined with an L_1-norm linear regularization term, the improved total penalty cost function for the soft constraints, J_{soft}, is defined as:

$$J_{soft} = \sum_{i=0}^{N_p-1} \left(\lambda_{lin} S_{k+i} + \rho(S_{k+i}) S_{k+i}^2\right) \tag{2.12}$$

By substituting Eq. (2.11) into the above expression and expanding it, the mathematical essence of this adaptive mechanism can be clearly revealed:

$$J_{soft} = \sum_{i=0}^{N_p-1} \left(\lambda_{lin} S_{k+i} + \rho_{base} S_{k+i}^2 + \rho_{base} \gamma S_{k+i}^3\right) \quad (2.13)$$

Equation (2.13) shows that the slack-variable penalty consists of a linear term, a fixed quadratic term, and a cubic term induced by the adaptive weight. For the conventional fixed quadratic penalty, the gradient with respect to S_{k+i} is $2\rho_{\text{base}} S_{k+i}$. This gradient approaches zero as $S_{k+i} \to 0$, and therefore provides relatively limited suppression of small nonzero slack values. In contrast, the gradient of the proposed penalty function is given by

$$\lambda_{\text{lin}} + 2\rho_{\text{base}} S_{k+i} + 3\rho_{\text{base}} \gamma S_{k+i}^2.$$

The linear term maintains a nonzero marginal penalty near $S_{k+i} = 0$, thereby helping to suppress small but persistent slack values. As S_{k+i} increases, the higher-order term associated with γ causes the penalty gradient to increase nonlinearly, strengthening the suppression of large constraint relaxations.

To ensure a fair comparison with the conventional fixed quadratic soft-constraint formulation, all soft-constrained controllers employ the same prediction model, state and input weighting matrices, prediction horizon, actuator constraints, base quadratic penalty coefficient, and solver settings. Only the penalty components already defined in Eq. (2.13) are enabled or disabled. When $\gamma = 0$ and $\lambda_{\text{lin}} = 0$, Eq. (2.13) reduces to the conventional fixed quadratic penalty, corresponding to soft_fixed. When $\gamma = 0$ and $\lambda_{\text{lin}} > 0$, the linear penalty is retained while the adaptive weighting is disabled, corresponding to soft_no_adaptive. When $\gamma > 0$ and $\lambda_{\text{lin}} > 0$, both the linear penalty and the adaptive weighting are enabled, corresponding to the complete improved_NMPC. Therefore, the comparison between soft_fixed and soft_no_adaptive isolates the contribution of the linear penalty term, whereas the comparison between soft_no_adaptive and improved_NMPC isolates the contribution of the adaptive weighting mechanism.

Taking the above soft-constraint mechanism into account, the overall cost function of the improved NMPC, denoted by J_{imp}, is defined as:

$$J_{\text{imp}} = \sum_{i=0}^{N_p-1} \left(\|X_{k+i|k} - X_{k+i}^{ref}\|_Q^2 + \|\tau_{k+i|k} - \tau_{k+i}^{ref}\|_R^2\right) + \|X_{k+N_p|k} - X_{k+N_p}^{ref}\|_{Q_T}^2 + J_{\text{soft}} \quad (2.14)$$

The corresponding finite-horizon optimal control problem is formulated as follows:

$$\begin{gathered}\min_{\left\{\tau_{k+i|k}\right\}_{i=0}^{N_p-1},S} J_{\text{imp}}(X,\tau,S)\\ \text{s.t.} X_{k|k}=X_{\text{measured}}(k),\\ X_{k+i+1|k}=f_d(X_{k+i|k},\tau_{k+i|k}),i=0,...,N_p-1,\\ \tau_{min}\le\tau_{r,k+i|k}\le\tau_{max},i=0,...,N_p-1,\\ \tau_{min}\le\tau_{l,k+i|k}\le\tau_{max},i=0,...,N_p-1,\\ |e_{lat,k+i|k}|\le d_{safe}+S_{k+i},i=0,...,N_p-1,\\ S_{k+i}\ge 0,i=0,...,N_p-1.\end{gathered} \tag{2.15}$$

By solving the above nonlinear programming problem, the controller obtains the optimal control sequence $\tau^*=\tau^*_{k|k},...,\tau^*_{k+N_p-1|k}$. The first element, $\tau^*_{k|k}$, is then applied to the low-level drive system, and the same procedure is repeated at each sampling instant to implement receding-horizon optimal control.

2.3 Stability Analysis

With the introduction of the soft-constraint mechanism in Eq. (2.10), the controller relaxes the original geometric safety constraints by means of the slack variable S_{k+i}, which substantially enhances the feasibility of the optimization problem under extreme operating conditions. Accordingly, this section theoretically investigates the improved soft-constrained NMPC from two aspects: recursive feasibility and closed-loop behavior.

It is worth noting that the stability results presented below are derived under the standard terminal-condition assumptions in NMPC, i.e., the existence of a terminal local control law and a terminal invariant set, under which the optimal value function can be employed as a candidate Lyapunov function[21]. Hence, the stability analysis in this section should be regarded as a conditional theoretical justification. By contrast, the recursive feasibility result follows directly from the intrinsic structure of the soft-constrained problem formulated in this paper.

In the hard-constrained NMPC formulation presented in Section 2.1, an excessively large initial state error of the robot may lead to the absence of any admissible control sequence within the prediction horizon that satisfies $|e_{\text{lat},k+i|k}|\le d_{\text{safe}}$. Consequently, the feasible set becomes empty, i.e., $X_{\text{feasible}}=\emptyset$, and the optimization solver may become stalled due to infeasibility. To address this issue, a nonnegative slack variable $S_{k+i}\ge 0$is introduced in this paper, and the geometric constraint is reformulated into the soft-constraint form shown in Eq. (2.10).

Theorem 2.1: Under the condition that only the input saturation constraints, the system dynamic equality constraints, and the softened geometric constraints are taken into account, with no additional hard state constraints or hard terminal constraints imposed, the improved soft-constrained optimization problem defined in Eq. (2.15) is always feasible for any initial state X_k.

Proof:

For any given initial state X_k, as long as the control inputs satisfy the actuator physical saturation constraints, the corresponding predicted state sequence

$\{X_{k+i|k}\}_{i=0}^{N_p}$ can be uniquely generated from the discrete-time dynamic equations. Accordingly, the values of the lateral error $e_{\mathrm{lat},k+i|k}$ are determined. Since the slack variable S_{k+i} is only subject to the lower bound 0 and has no upper bound, it is always possible to choose

$$S_{k+i} \geq max(0, |e_{lat,k+i|k}| - d_{safe}) \tag{2.16}$$

such that the soft constraint inequality is satisfied at all prediction steps. Therefore, for any given initial state X_k, one can always construct a feasible decision sequence (X, τ, S) that satisfies all the constraints. Hence, the feasible set of the optimization problem (2.15) is always nonempty, which implies that the problem remains feasible for any initial state and therefore possesses recursive feasibility. **Q.E.D.**

From the foregoing proof, it can be seen that the introduction of slack variables expands the originally finite feasible region truncated by hard constraints into a soft feasible region over the entire state space, thereby preventing the softened corridor constraint itself from causing infeasibility under the conditions of Theorem 2.1. Numerical solver failure may still occur because of iteration limits, numerical conditioning, inappropriate initialization, or implementation errors.

Under the assumptions of Theorem 2.1, the softened corridor constraint does not make the optimization problem infeasible. However, feasibility does not automatically imply closed-loop stability. To address this, the standard optimal value function analysis framework commonly used in NMPC is introduced[22].

Assumption 2.1: There exist a terminal local control law $\tau_f(X)$ and a terminal invariant set X_f such that: (1) for all $X \in X_f$, the input constraints are satisfied under $\tau_f(X)$; (2) X_f is positively invariant for the closed-loop system; (3) X_f is contained in the original hard-constrained feasible region, so that the terminal slack variable can be selected as zero; and (4) the terminal weighting matrix Q_T satisfies the following local descent condition. For the conditional analysis below, it is additionally assumed that the predicted terminal state satisfies $X^*_{k+N_p|k} \in X_f$.

$$\|X_{k+1} - X_{k+1}^{ref}\|_{Q_T}^2 - \|X_k - X_k^{ref}\|_{Q_T}^2 \leq -(\|X_k - X_k^{ref}\|_Q^2 + \|\tau_f(X_k) - \tau_k^{ref}\|_R^2) \tag{2.17}$$

Let the optimal control sequence and the optimal slack sequence obtained at time k be denoted by τ_k^* and S_k^*, respectively, and let the corresponding optimal cost be denoted by $V(X_k) = J_{\mathrm{imp}}^*(X_k)$.

Theorem 2.2: Suppose that Assumption 2.1 holds. Then, the optimal value function $V(X_k)$ is non-increasing along the closed-loop trajectory. Furthermore, when the original hard-constrained problem is feasible in the current neighborhood and the exact penalty condition is satisfied, the closed-loop error system converges asymptotically, and the optimal slack variables converge to zero.

Proof: After applying the optimal control input $\tau^*_{(k|k)}$ at time k, the system state evolves to X_{k+1}. At time $k+1$, a shifted sequence is constructed as a suboptimal candidate solution according to (2.18) and (2.19).

$$\tilde{\tau}_{k+1} = \{\tau^*_{k+1|k}, ..., \tau^*_{k+N_p-1|k}, \tau_f(X_{k+N_p|k})\} \tag{2.18}$$

$$\tilde{S}_{k+1} = \{S^*_{k+1|k}, ..., S^*_{k+N_p-1|k}, 0\} \tag{2.19}$$

Note: Since the terminal state has entered the terminal invariant set, the slack variable is set to zero.

Substituting this suboptimal sequence into the cost function at time $k+1$, and comparing it with the optimal cost at time k, it follows from the terminal decrease condition $\Delta_T \leq 0$in Assumption 2.1 that:

$$\begin{aligned}\tilde{J}_{imp}(X_{k+1}) - J^*_{imp}(X_k) \leq & -(\|X_k - X_k^{ref}\|_Q^2 \\ & + \|\tau^*_{k|k} - \tau_k^{ref}\|_R^2 + J_{soft}(S^*_{k|k}))\end{aligned} \tag{2.20}$$

Since the true optimal cost satisfies $J^*_{\text{imp}}(X_{k+1}) \leq \tilde{J}_{\text{imp}}(X_{k+1})$, the Lyapunov decrease condition is obtained as follows

$$\begin{aligned}V(X_{k+1}) - V(X_k) \leq & -(\|X_k - X_k^{ref}\|_Q^2 \\ & + \|\tau^*_{k|k} - \tau_k^{ref}\|_R^2 + J_{soft}(S^*_{k|k})) \leq 0\end{aligned} \tag{2.21}$$

Thus, the optimal value function is non-increasing. Under Assumption 2.1, boundedness of the closed-loop trajectory, and the stated exact-penalty condition, the tracking error and control-input deviation converge to zero, and the optimal slack converges to zero once the original hard-constrained region becomes locally feasible. Because no explicit terminal set is implemented in the numerical or experimental controller, this result is a conditional stability property rather than an unconditional guarantee.

3 Simulation Verification

The simulations comprised nominal feasible tracking tests, initial corridor-violation tests, and multi-initial-condition recovery tests. The nominal tests compared the standard and hard-constrained NMPC controllers, whereas the recovery tests compared the standard NMPC, soft_fixed, soft_no_adaptive, and improved_NMPC.

All simulations were performed in MATLAB R2022b using CasADi and IPOPT with the MUMPS linear solver on a Windows 10 computer equipped with an Intel Core i5-9300H CPU and 16 GB of RAM. The sampling period and prediction horizon were $T_s = 0.05\,\text{s}$ and $N_p = 20$, respectively. The IPOPT iteration limit and convergence tolerance were set to 500 and 10^{-6}, with warm-start initialization used for all controllers.

For the straight-line tests, the nominal feasible scenario used an initial lateral deviation of 0.04 m and a longitudinal velocity of 1.5 m/s, whereas the boundary-recovery scenario used 0.10 m and 0.5 m/s, respectively. Comparisons were performed only under identical conditions within each scenario. The circular reference velocity was determined separately from its radius and angular velocity.

The parameters of the robot and the experimental setup are given in Table 3.1.

Table 3.1 Robot Parameters

Parameter	Symbol	Value	Unit
Vehicle width	W_{robot}	1	m
Half track width	L	0.4	m
Wheel radius	r	0.1	m
Vehicle mass	m	200	kg
Moment of inertia	I	59.33	kg·m²
Virtual corridor width	W_{aisle}	1.2	m
Single-side safety threshold	d_{margin}	0.05	m

Based on Eq. (2.5) and the parameters in Table 3.1, a one-sided safety margin of 0.05 m yields an effective corridor half-width of

$$d_{\text{safe}} = \frac{1.20}{2} - \frac{1.00}{2} - 0.05 = 0.05\text{m}.$$

For the nominal feasible straight-line test, the initial lateral deviation was set to 0.04 m, which lies within the effective corridor. The standard and hard-constrained NMPC controllers were compared in terms of tracking accuracy and computational efficiency.

The sampling period of the model predictive controller is set to $T_s = 0.05$ s, and the prediction horizon is chosen as $N_p = 20$. The weighting matrices in the objective function are selected as follows:

$$\begin{aligned} Q &= diag(150,200,150,8,2) \\ R &= diag(0.01,0.01) \\ \mathrm{Q}_T &= \mathrm{diag}(200,250,200,10,3) \end{aligned} \tag{3.1}$$

To comprehensively assess the tracking performance and robustness of the controller under different curvature conditions, two representative reference trajectories are considered in this study: a straight line trajectory, a circular trajectory.

In trajectory tracking control of nonholonomic mobile robots, the reference trajectory must satisfy the kinematic feasibility of the system. To this end, a virtual reference vehicle model subject to the same nonholonomic constraints is introduced to generate the desired states, and its state differential equations are defined as follows:

$$\dot{q}^{ref}(t) = \begin{bmatrix} \dot{x}^{ref}(t) \\ \dot{y}^{ref}(t) \\ \dot{\theta}^{ref}(t) \end{bmatrix} = \begin{bmatrix} v^{ref}(t)\cos\theta^{ref}(t) \\ v^{ref}(t)\sin\theta^{ref}(t) \\ \omega^{ref}(t) \end{bmatrix} \tag{3.2}$$

It follows from the geometric relationships that, for any given planar explicitly time-parameterized path $\left(x^{ref}(t), y^{ref}(t)\right)$, the reference heading angle $\theta^{ref}(t)$ and the reference control input $u^{ref}(t) = [v^{ref}(t), \omega^{ref}(t)]^{\mathrm{T}}$ can be obtained in a unified manner by differentiation as follows:

$$\begin{cases} v^{ref}(t) = \sqrt{(\dot{x}^{ref}(t))^2 + (\dot{y}^{ref}(t))^2} \\ \theta^{ref}(t) = \arctan2\left(\dot{y}^{ref}(t), \dot{x}^{ref}(t)\right) \\ \omega^{ref}(t) = \dot{\theta}^{ref}(t) \end{cases} \tag{3.3}$$

Performance metrics

To quantitatively evaluate the controller performance, let N_s denote the total number of simulation samples. The eight performance metrics used in the following comparisons are formally defined as follows:

$$\text{latP95} = Q_{0.95}\left(\{|e_{\text{lat},k}|\}_{k=1}^{N_s}\right), \text{latMax} = \max_{1\le k\le N_s} |e_{\text{lat},k}|$$

$$\text{lonP95} = Q_{0.95}\left(\{|e_{\text{lon},k}|\}_{k=1}^{N_s}\right), \text{lonMax} = \max_{1\le k\le N_s} |e_{\text{lon},k}|$$

$$\text{psiP95} = Q_{0.95}\left(\{|e_{\psi,k}|\}_{k=1}^{N_s}\right), \text{psiMax} = \max_{1\le k\le N_s} |e_{\psi,k}|$$

$$\text{viol}\% = \frac{100}{N_s}\sum_{k=1}^{N_s} \mathbb{I}\left(|e_{\text{lat},k}| > d_{\text{safe}}\right)$$

$$t_{\text{ms}} = \frac{1000}{N_s}\sum_{k=1}^{N_s} t_{\text{solve},k}$$

Here, $Q_{0.95}(\cdot)$ denotes the empirical 95th percentile, $\mathbb{I}(\cdot)$ is the indicator function that equals 1 when its condition is satisfied and 0 otherwise, and $t_{\text{solve},k}$ denotes the optimization time at the k-th sampling instant in seconds. The lateral and longitudinal errors are expressed in meters, the heading-angle errors in radians, the boundary-violation rate in percent, and the average solution time in milliseconds.

(1) Straight-line trajectory

Based on the above framework, the straight-line reference trajectory with inclination angle α is defined in this paper as follows:

$$\begin{cases} x^{ref}(t) = v_0 \cos(\alpha) t \\ y^{ref}(t) = v_0 \sin(\alpha) t \end{cases} \tag{3.4}$$

where v_0 denotes the constant reference linear velocity specified previously, and α is the inclination angle of the straight-line trajectory with respect to the positive X-axis. In this paper, $\alpha = \pi/4$ is chosen, corresponding to a straight line at $45°$. Taking the time derivative yields the components of the reference velocity as follows:

$$\begin{cases} \dot{x}^{ref}(t) = v_0 \cos(\alpha) \\ \dot{y}^{ref}(t) = v_0 \sin(\alpha) \end{cases} \tag{3.5}$$

As shown in Figs. 3.1 and 3.2, in the nominal straight-line tracking scenario, the initial lateral deviation is set to 0.04 m, and the initial and reference longitudinal velocities are both set to 1.5 m/s, the simulated trajectories of the standard NMPC and hard-constrained NMPC nearly overlap. This result indicates that the hard geometric constraint does not noticeably affect tracking accuracy when the initial state remains

inside the admissible corridor. Under this nominal feasible condition, both controllers achieve comparable tracking performance, while the hard-constrained NMPC maintains the lateral deviation within the prescribed corridor boundary.

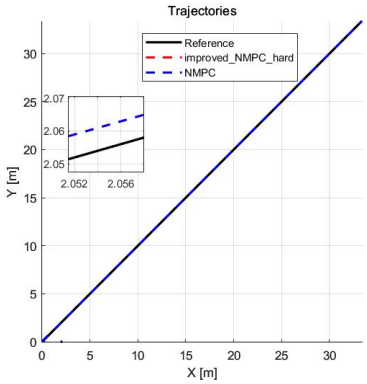


Fig.3.1Straight-Line Trajectory Tracking

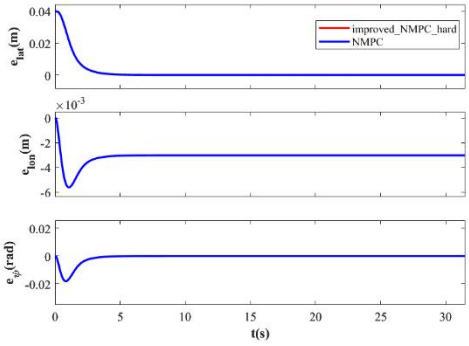


Fig. 3.2 Straight-Line Tracking Error

As shown by the data in Table 3.2, the 95th-percentile error (P95) and the maximum absolute error (Max) of the two controllers in the lateral, longitudinal, and heading dimensions are exactly identical, and the boundary violation rate (viol%) is zero for both.

The average solution times of the standard NMPC and the hard-constrained NMPC are 6.93 ms and 7.50 ms, respectively. Although the hard-constrained formulation introduces a slight increase in computational time, both values remain well below the sampling period of 50 ms, indicating that the additional corridor constraint does not compromise real-time feasibility under the nominal condition.

Table 3.2 Comparison of Comprehensive Performance Metrics for Straight-Line Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
improved_ NMPC_hard	0.012	0.04	0.004	0.006	0.006	0.018	0	7.5
NMPC	0.012	0.04	0.004	0.006	0.006	0.018	0	6.93

(2) Circular trajectory

The circular reference trajectory can be derived from (3.3) as follows:

$$\begin{cases} x^{ref}(t) = R\cos(\omega t) \\ y^{ref}(t) = R\sin(\omega t) \end{cases} \tag{3.6}$$

The velocity settings of 1.5 m/s and 0.5 m/s used in the preceding straight-line experiments do not apply to the circular trajectory. For the circular reference

trajectory, the reference linear velocity is determined independently by $v^{ref} = R\omega$. With $R = 2\,m$ and $\omega = 0.2\,rad/s$, the corresponding reference linear velocity is 0.4 m/s.

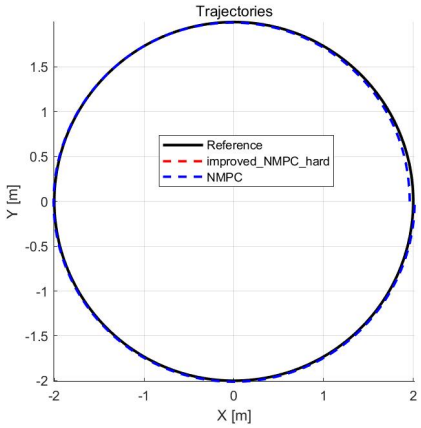


Fig. 3.3 Circular Trajectory

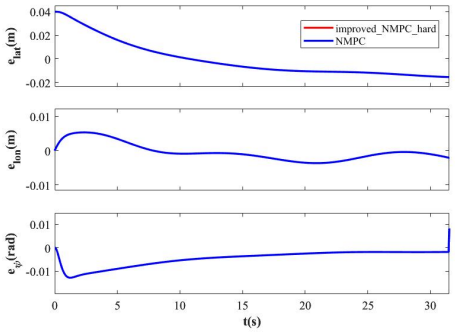


Fig. 3.4 Circular Trajectory Error

It can likewise be observed from the figures and the table that, as long as no boundary violation occurs, the responses of the two controllers remain highly consistent. Therefore, no further detailed discussion is provided here.

Table 3.3 Comparison of Comprehensive Performance Metrics for Circular Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
improved_ NMPC_hard	0.034	0.04	0.005	0.005	0.011	0.013	0	8.84
NMPC	0.034	0.04	0.005	0.005	0.011	0.013	0	8.07

Soft-constraint formulation:

To evaluate feasibility and boundary recovery under an initial corridor violation, a low-speed straight-line scenario was considered with an initial lateral deviation of 0.10 m, exceeding the corridor half-width of 0.05 m. The initial and reference longitudinal velocities were both set to 0.5 m/s. Consequently, the hard-constrained NMPC was infeasible at the initial sampling instant and was excluded from the closed-loop comparisons.

According to Eq. (2.13), the base quadratic, adaptive, and linear penalty coefficients were set to 50,000, 500, and 200, respectively. The standard NMPC, soft_fixed, soft_no_adaptive, and improved_NMPC were evaluated using identical models, trajectories, controller settings, and actuator constraints. As shown in Figs. 3.5–3.8,

all soft-constrained controllers remained feasible, with improved_NMPC providing the fastest lateral-error recovery.

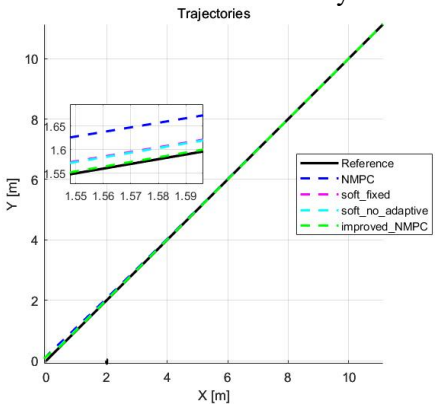


Fig. 3.5 Soft-Constrained Straight-Line Trajectory

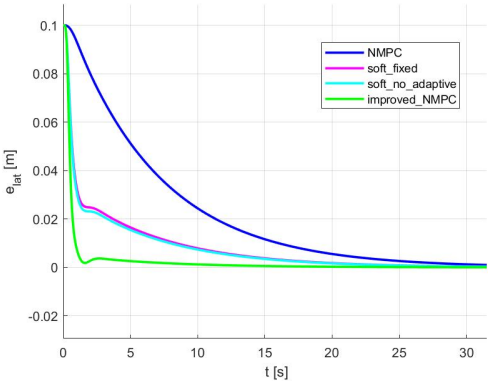


Fig. 3.6 Lateral Error under Constrained Straight-Line Tracking

As shown in Figs. 3.5–3.8, the standard NMPC produces the slowest lateral-error reduction because it does not explicitly account for the virtual corridor boundary. The fixed-weight quadratic soft NMPC and the non-adaptive linear-quadratic soft NMPC reduce the lateral deviation more rapidly, while their responses remain highly similar. The proposed adaptive soft NMPC produces the fastest boundary-recovery response and returns the lateral deviation to the effective corridor in approximately 1 s. However, this faster lateral correction is accompanied by larger transient longitudinal and heading-angle errors. In particular, the proposed controller reaches a maximum longitudinal error of 0.217 m and a maximum heading-angle error of 0.215 rad. Therefore, the adaptive mechanism improves boundary-recovery performance at the cost of increased transient motion adjustment, rather than improving all tracking-error indices simultaneously

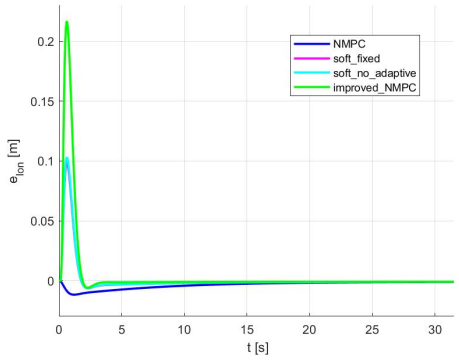


Fig. 3.7 Longitudinal Error under Soft-Constrained Straight-Line Tracking

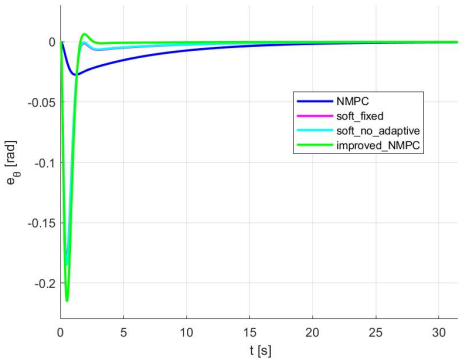


Fig.3.8 Heading-Angle Error under Soft-Constrained Straight-Line Tracking

As in Table 3.4, the hard-constrained NMPC was infeasible at the initial sampling instant. Compared with the standard NMPC, soft_fixed reduced latP95 from 0.086 to 0.025 m and the violation rate from 16.48% to 2.22%. Soft_no_adaptive provided a

modest further reduction to 0.024 m and 2.06%, respectively, whereas improved_NMPC achieved the lowest values of 0.004 m and 1.58%. However, improved_NMPC produced the largest lonMax and psiMax values of 0.217 m and 0.215 rad. Its solution time of 11.59 ms remained below the 50 ms sampling period.

Table 3.4 Comparison of Comprehensive Performance Metrics for Soft-Constrained Straight-Line Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
Improved _NMPC_hard	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
NMPC	0.086	0.10	0.010	0.012	0.023	0.027	16.48	6.77
improved_NMPC	0.004	0.1	0.007	0.217	0.006	0.215	1.58	11.59
soft_fixed	0.025	0.100	0.006	0.099	0.007	0.181	2.22	9.76
soft_no_adaptive	0.024	0.100	0.006	0.103	0.006	0.185	2.06	11.59

In the circular-trajectory experiment, the hard-constrained NMPC is again infeasible at the initial sampling instant and is therefore excluded from the closed-loop curve comparison. As shown in Figs. 3.9–3.12, the standard NMPC exhibits the slowest lateral-error reduction. Both soft_fixed and soft_no_adaptive substantially accelerate boundary recovery, whereas their lateral, longitudinal, and heading-angle responses remain close to each other. The proposed improved_NMPC produces the fastest lateral-error reduction and the lowest boundary-violation exposure, although it generates larger transient longitudinal and heading-angle errors during the initial recovery stage.

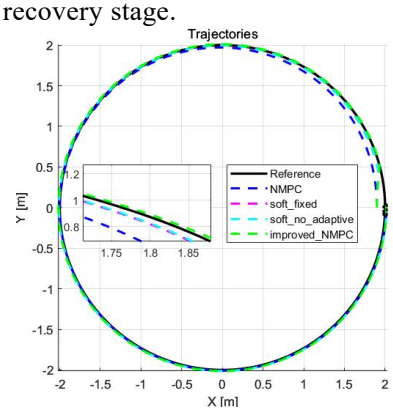


Fig. 3.9 Soft-Constrained Circular Trajectory

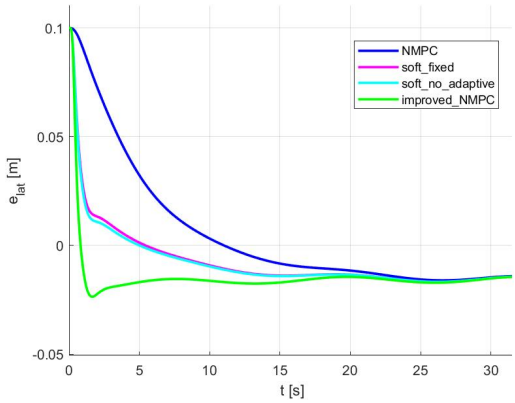


Fig. 3.10 Lateral Error in Circular Trajectory Tracking

It can also be observed from Figs. 3.11 and 3.12 that, in order to correct the initial lateral deviation, the soft-constrained NMPC produces relatively large longitudinal and heading-angle errors during roughly the first 1 s. The larger transient longitudinal and heading-angle errors reflect the control effort required for rapid lateral recovery. These transient responses remain bounded in the present numerical simulations, but their practical acceptability should be further evaluated through platform experiments.

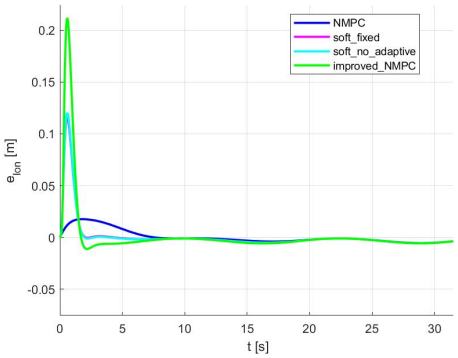


Fig. 3.11 Longitudinal Error under Soft-Constrained Circular Tracking

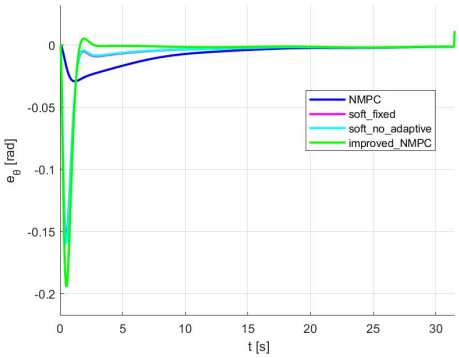


Fig. 3.12 Heading-Angle Error under Soft-Constrained Circular Tracking

As shown in Table 3.5, soft_fixed reduced latP95 and the violation rate from 0.087 m and 15.85% for the standard NMPC to 0.025 m and 2.22%, respectively. Improved_NMPC achieved the lowest values of 0.017 m and 1.58%, but produced the largest transient longitudinal and heading errors.

Table 3.5 Comparison of Comprehensive Performance Metrics for Soft-Constrained Circular Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
improved_ NMPC_hard	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
NMPC	0.087	0.100	0.016	0.016	0.021	0.024	15.85	8.63
improved_NMPC	0.017	0.100	0.008	0.227	0.003	0.200	1.58	11.70
soft_fixed	0.025	0.100	0.008	0.129	0.010	0.165	2.22	10.96
soft_no_adaptive	0.022	0.100	0.008	0.133	0.009	0.170	2.22	11.85

Multi-Initial-Condition Boundary-Recovery Tests

The preceding boundary-recovery tests used fixed initial lateral and heading errors. To examine whether the observed recovery performance depends on a single initial state, additional simulations were conducted under multiple initial lateral and heading errors.

Ten initial conditions were considered for each of the straight-line and circular trajectories. Let $(e_{\mathrm{lat},0}, e_{\psi,0})$ denote the initial lateral error and initial heading error, respectively. The selected initial conditions were:

$$\{(-0.10, -0.05), (-0.10,0.05), (-0.08,0),$$
$$(-0.06, -0.05), (-0.06,0.05), (0.06, -0.05),$$
$$(0.06,0.05), (0.08,0), (0.10, -0.05), (0.10,0.05)\}$$

Here, $e_{\mathrm{lat},0}$ and $e_{\psi,0}$ are expressed in meters and radians, respectively. The selected conditions cover both sides of the reference trajectory and positive and negative initial

heading errors. Because $|e_{\mathrm{lat},0}| > d_{\mathrm{safe}} = 0.05\,\mathrm{m}$ in all tests, the hard-constrained NMPC was initially infeasible and is therefore reported as N/A.

The standard NMPC, soft_fixed, soft_no_adaptive, and improved_NMPC were evaluated using identical models, controller settings, and solver parameters, with only the penalty components defined in Section 2.2 varied. The straight-line tests used initial and reference longitudinal velocities of 0.5 m/s, whereas the circular reference velocity was determined by the prescribed radius and angular velocity. Table 3.6 reports the mean and standard deviation of the eight performance metrics over ten initial conditions.

Table 3.6 Statistical Results of Comprehensive Performance Metrics under Multi-Initial-Condition Boundary-Recovery Tests

(a) Straight-line trajectory

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
improved_ NMPC_hard	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
NMPC	0.068 ±0.019	0.084 ±0.019	0.009 ±0.002	0.010 ± 0.003	0.019 ±0.005	0.044± 0.012	11.06 ±6.03	6.75 ±0.12
soft_fixed	0.032 ±0.007	0.081 ±0.019	0.006 ±0.001	0.055 ± 0.044	0.009 ±0.002	0.120 ± 0.059	2.11 ±0.37	9.82 ±0.23
soft_no_adaptive	0.030 ±0.007	0.081 ±0.019	0.006 ±0.001	0.058 ± 0.046	0.008 ±0.002	0.125 ± 0.060	2.04 ±0.34	11.43 ±0.20
improved_NMPC	0.023 ±0.009	0.081 ±0.019	0.006 ±0.002	0.115 ± 0.090	0.008 ±0.005	0.157 ± 0.085	1.52 ±0.17	11.41 ±0.22

(b) Circular trajectory

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
improved_ NMPC_hard	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
NMPC	0.071 ±0.019	0.083 ±0.019	0.013 ±0.004	0.014 ±0.004	0.014± 0.006	0.043 ±0.014	12.58 ±6.87	8.18 ± 0.14
soft_fixed	0.037 ±0.007	0.081 ±0.019	0.009 ±0.002	0.064 ±0.046	0.008± 0.004	0.120 ±0.065	2.31 ±0.34	10.47 ±0.33
soft_no_adaptive	0.035 ±0.008	0.081 ±0.019	0.008 ±0.002	0.068 ±0.047	0.008± 0.003	0.125 ±0.067	2.19 ±0.31	11.49 ±0.09
improved_NMPC	0.032 ±0.011	0.081 ±0.019	0.011 ±0.009	0.163 ±0.141	0.007± 0.003	0.132 ±0.073	1.66 ±0.23	11.47 ±0.20

The hard-constrained NMPC was infeasible for all tested initial conditions. For the straight-line trajectory, soft_fixed reduced the mean latP95 and violation rate from 0.068 ± 0.019 m and 11.06 ± 6.03% for the standard NMPC to 0.032 ± 0.007 m and 2.11 ± 0.37%, respectively. Soft_no_adaptive provided a modest further improvement, whereas improved_NMPC achieved the lowest values of 0.023 ± 0.009 m and 1.52 ± 0.17%.

For the circular trajectory, the mean latP95 and violation rate decreased from 0.071 ± 0.019 m and 12.58 ± 6.87% for the standard NMPC to 0.037 ± 0.007 m and 2.31 ± 0.34% for soft_fixed. Improved_NMPC further reduced these values to 0.032 ± 0.011 m and 1.66 ± 0.23%, confirming the additional, although modest, benefit of the linear and adaptive penalty terms.

Because latMax was primarily determined by the prescribed initial error, it was not used to assess recovery performance. Improved_NMPC achieved better lateral recovery at the cost of larger transient longitudinal errors, with mean lonMax values of 0.115 ± 0.090 m and 0.163 ± 0.141 m for the straight-line and circular trajectories, respectively. Its mean solution times remained below the 50 ms sampling period for both trajectories.

4 Hardware Experimental Validation

4.1 Experimental Platform and Parameter Settings

Hardware trajectory-tracking experiments were conducted on a differential-drive container transferring mobile robot to evaluate the practical implementability of the proposed controller. The onboard control system consisted of an Advantech ARK-1250 industrial computer with an Intel Core i5-1145G7E processor and 8 GB of RAM, running a ROS-based control algorithm. The computer transmitted wheel-torque commands via a CAN bus to two Kinco FD134S-CB-000 servo drives, which controlled Kinco SMC80S-0075-30MAK-5DSU motors through 1:18 reduction mechanisms.

The hardware experiments used the same robot geometry and virtual-corridor parameters as the simulations (Table 3.1). The sampling period was 0.05 s, the prediction horizon was 20, and the wheel-torque limits were ±5 N·m. Each experiment lasted approximately 64 s at a reference linear velocity of 0.08 m/s, with an initial lateral error of −0.08 m and an initial heading error of 0 rad.

As shown in Fig. 4.1, the platform comprises a differential-drive chassis and an upper container-handling mechanism. Only the chassis was operated because the experiments focused on trajectory-tracking performance. The CAN topology is shown in Fig. 4.2; the left and right drives, assigned node IDs 1 and 3, respectively, performed closed-loop motor control using encoder feedback.

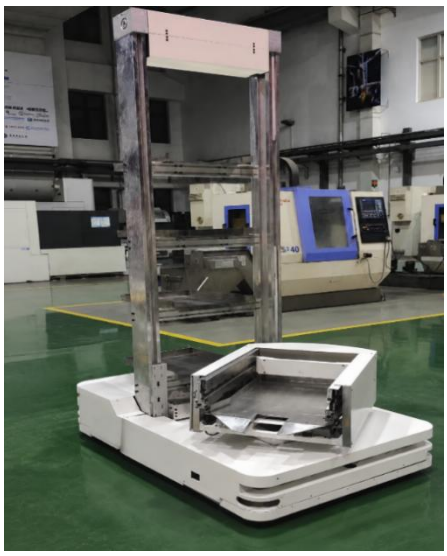


Fig. 4.1 Experimental Platform of the Container Transferring Mobile Robot

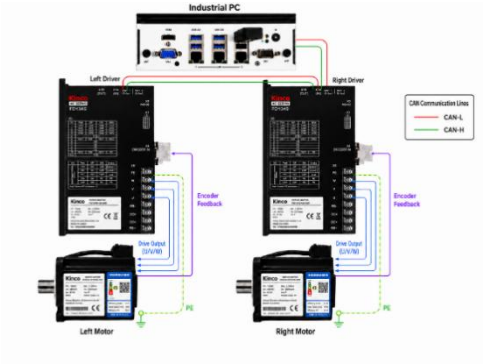


Fig. 4.2 CAN-Based Control and Drive System Architecture

4.2 Experimental Conditions and Controller Settings

Straight-line and circular trajectory-tracking experiments were conducted to evaluate controller feasibility and boundary recovery from an initial state outside the prescribed safety threshold. The standard NMPC, soft_fixed, and improved_NMPC were compared under identical experimental conditions. The hard-constrained NMPC was excluded because the initial lateral error exceeded the corridor threshold, rendering its optimization problem infeasible at the initial sampling instant.

Both trajectories were tracked at a reference linear velocity of 0.08 m/s for approximately 64 s, with an initial lateral error of −0.08 m and a safety threshold of 0.05 m. The circular trajectory had a radius of 2 m and a reference angular velocity of approximately 0.04 rad/s. Because of space and safety limitations, the robot traversed an arc of approximately 147° rather than a complete circle. The remaining experimental parameters are summarized in Table 4.1.

The straight-line and circular reference trajectories were generated using Eqs. (3.4) and (3.6), respectively, with $\alpha = 0$ for the straight-line trajectory.

Table 4.1 Hardware Experimental Parameters

Parameter	Straight line	Circular trajectory
Sampling time/s	0.05	0.05
Prediction horizon	20	20
Reference linear velocity/(m/s)	0.08	0.08
Reference angular velocity/(rad/s)	0	0.04
Trajectory radius/m	—	2.0

Initial lateral error/m	−0.08	−0.08
Initial heading error/rad	0	0
Lateral safety threshold/m	0.05	0.05
Wheel-torque limit/(N·m)	±5	±5
Experimental duration/s	Approximately 64	Approximately 64

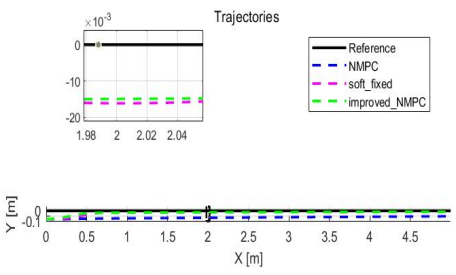
Considering the actuator response, ground resistance, and communication delay of the physical robot, the controller weighting matrices were adjusted from their simulation values. The three control strategies employed:

$$Q = \mathrm{diag}\,(100,300,120,300,80)$$
$$Q_T = \mathrm{diag}\,(200,600,240,500,120) \tag{4.1}$$
$$R = \mathrm{diag}\,(0.1,0.1)$$

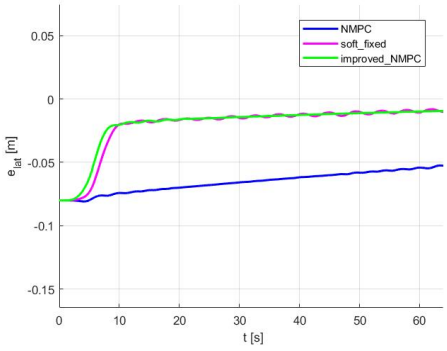
The soft-constraint parameters were set to $\rho_{\mathrm{base}} = 50000, \gamma = 0, \lambda_{\mathrm{lin}} = 0$ for soft_fixed,and to $\rho_{\mathrm{base}} = 50000, \gamma = 20, \lambda_{\mathrm{lin}} = 30$ for improved_NMPC, according to Eq. (2.13). Because the controller weights were tuned separately on the hardware platform, these experiments assess practical implementation and overall performance rather than provide a controlled ablation. The contributions of the individual penalty components are evaluated using identical weighting matrices in the simulations in Section 3. Hardware performance was assessed using the eight metrics defined in Section 3.

4.3 Straight-Line Trajectory Tracking Experiment

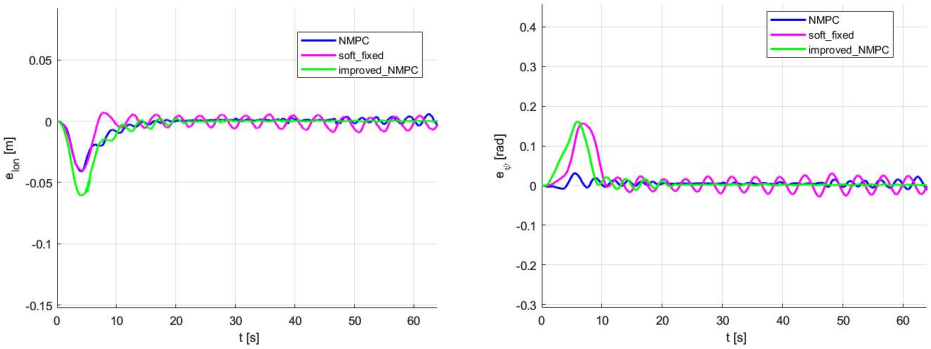
Figure 4.3 compares the trajectories and tracking errors of the three controllers in the straight-line experiment. All controllers maintained closed-loop operation without solver failure or fallback activation. The standard NMPC gradually reduced the lateral error but did not remain within the safety threshold by the end of the experiment, whereas both soft-constrained controllers recovered within the threshold, with improved_NMPC showing slightly faster recovery. Their initially larger longitudinal and heading errors subsequently decreased and fluctuated around zero.



(a) Straight-line trajectory



(b) Lateral error



(c)Longitudinal error (d)Heading-angle error

Fig. 4.3 Hardware Experimental Results of Straight-Line Trajectory Tracking: (a) trajectories; (b) lateral errors; (c) longitudinal errors; (d) heading-angle errors.

To further quantitatively compare the hardware performance of the three controllers, Table 4.2 lists the eight performance metrics defined in Section 3.

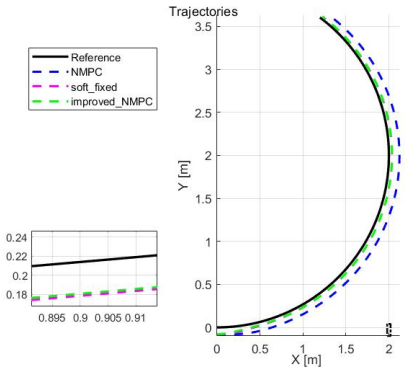
Table 4.2 Comparison of Comprehensive Performance Metrics for Hardware Straight-Line Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
NMPC	0.080	0.081	0.022	0.041	0.017	0.031	100.00	7.80
soft_fixed	0.079	0.080	0.023	0.040	0.115	0.157	10.77	9.80
improved_NMPC	0.077	0.080	0.039	0.061	0.103	0.161	9.13	13.13

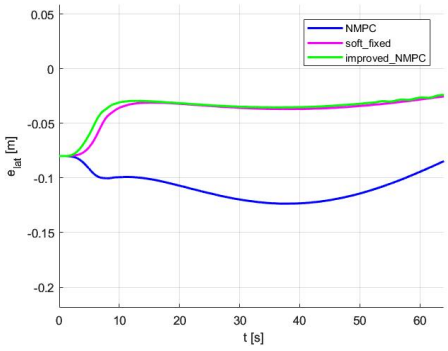
Compared with soft_fixed, improved_NMPC reduced the violation rate from 10.77% to 9.13% and latP95 from 0.079 to 0.077 m, but increased lonP95 from 0.023 to 0.039 m and lonMax from 0.040 to 0.061 m. All controllers completed the computation within the 50 ms sampling period.

4.4 Circular Trajectory Tracking Experiment

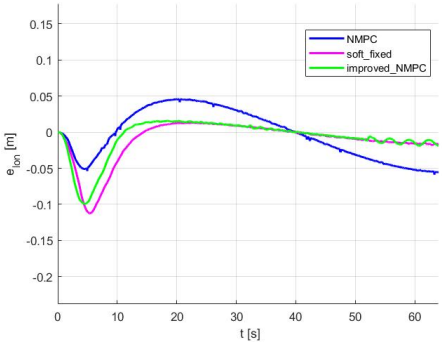
Figure 4.4 compares the trajectories and tracking errors of the three controllers in the circular experiment. The standard NMPC exhibited a persistent lateral deviation and did not enter the prescribed safety threshold. Both soft-constrained controllers recovered toward the reference trajectory, with improved_NMPC showing slightly faster lateral-error recovery. Their transient longitudinal and heading errors gradually decreased after the initial correction stage.



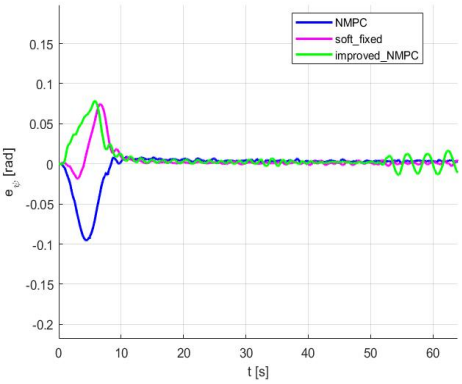
(a) Circular trajectory



(b)Lateral error



(c)Longitudinal error



(d)Heading-angle error

Fig. 4.4 Hardware Experimental Results for Circular Trajectory Tracking: (a) trajectories; (b) lateral errors; (c) longitudinal errors; and (d) heading-angle errors.

Table 4.3 Comparison of Comprehensive Performance Metrics for Hardware Circular Trajectory Tracking

Controller	latP95	latMax	lonP95	lonMax	psiP95	psiMax	viol%	t_ms
NMPC	0.123	0.124	0.053	0.059	0.059	0.095	100.00	7.45
soft_fixed	0.079	0.080	0.088	0.112	0.027	0.075	11.32	9.95
improved_NMPC	0.075	0.080	0.077	0.099	0.049	0.079	9.29	13.25

Compared with soft_fixed, improved_NMPC reduced latP95 from 0.079 to 0.075 m, the violation rate from 11.32% to 9.29%, and the recovery time from 7.25 to 5.95 s. It also reduced longitudinal errors but produced slightly larger heading errors. All controllers completed the computation within the 50 ms sampling period without solver failure or fallback activation.

5 Conclusions and Future Work

An improved soft-constrained NMPC method was developed for trajectory tracking of a container transferring mobile robot in narrow corridors. A virtual-corridor constraint was constructed based on the robot width, aisle width, and safety margin. Slack variables were introduced to maintain optimization feasibility, while linear and slack-dependent penalties were used to strengthen boundary recovery. Simulations showed that the proposed method improved trajectory-tracking accuracy and boundary-recovery performance, reducing latP95 and the boundary-violation rate, although faster recovery produced larger transient longitudinal and heading errors.

Hardware experiments confirmed recovery from an initial lateral error of −0.08 m. Compared with soft_fixed, improved_NMPC slightly improved trajectory-tracking accuracy and reduced the recovery time and boundary-violation rate for both straight-line and circular trajectories. Its average solution time remained below the 50 ms sampling period, demonstrating real-time implementability.

Nevertheless, the softened virtual corridor does not provide an absolute safety guarantee. Future work will include repeated hardware tests under unified controller weights and evaluations at different velocities, payloads, and external disturbances.

Fund project: Research Initiation Fund Project of Civil Aviation University of China (No. 2020KYQD84)
Fundamental Research Funds for the Central Universities (No. 3122022052)

References

[1] Klančar, G., & Seder, M. (2022). Coordinated multi-robotic vehicles navigation and control in shop floor automation. *Sensors*, *22*(4), 1455.
[2] Achirei, S. D., Mocanu, R., Popovici, A. T., & Dosoftei, C. C. (2023). Model-predictive control for omnidirectional mobile robots in logistic environments based on object detection using CNNs. *Sensors*, *23*(11), 4992.
[3] Lucet, Eric, Antoine Lucazeau, and Jason Chemin. "Autonomous Forklift Navigation Inside a Cluttered Logistics Factory." *ICINCO* (2). 2024.
[4] Zhang, L., Ao, W., Zhang, H., Liu, P., Li, Z., & Minchala, L. I. (2024). Dynamics modeling and trajectory tracking control of a life support robot using sliding-mode control with extended state observer. *Journal of the Franklin Institute*, *361*(12), 107013.
[5] Moritz, J., Musa, M., & Wejinya, U. (2024). Design and modeling of a two-wheeled differential drive robot. *International Journal of Control, Automation and Systems*, *22*(7), 2273-2282.
[6] Liu, Q., & Cong, Q. (2022). Kinematic and dynamic control model of wheeled mobile robot under internet of things and neural network. *The Journal of Supercomputing*, *78*(6), 8678-8707.
[7] Köhler, J., Müller, M. A., & Allgöwer, F. (2024). Analysis and design of model predictive control frameworks for dynamic operation—An overview. *Annual Reviews in Control*, *57*, 100929.

[8] Wang, J., Liu, Z., Chen, H., Zhang, Y., Zhang, D., & Peng, C. (2023). Trajectory tracking control of a skid-steer mobile robot based on nonlinear model predictive control with a hydraulic motor velocity mapping. *Applied Sciences*, *14*(1), 122.

[9] Peng, J., Xiao, H., & Lai, G. (2024). Nonlinear disturbance observer incorporated model predictive strategy for wheeled mobile robot's trajectory tracking control. *International Journal of Control, Automation and Systems*, *22*(7), 2251-2262.

[10] Tang, M., Zhang, Y., Yu, S., Li, J., & Tang, K. (2025). Trajectory tracking model predictive control for mobile robot based on deep Koopman operator modeling. *Robotics and Autonomous Systems*, 105152.

[11] Tang, Jiawei, et al. "GMPC: Geometric model predictive control for wheeled mobile robot trajectory tracking." *IEEE Robotics and Automation Letters* 9.5 (2024): 4822-4829.

[12] Fnadi, M., Du, W., Plumet, F., & Benamar, F. (2021). Constrained model predictive control for dynamic path tracking of a bi-steerable rover on slippery grounds. *Control Engineering Practice*, *107*, 104693.

[13] Nam, N. N., & Han, K. (2024). Path-tracking robust model predictive control of an autonomous steering system using LMI optimization with independent constraints enforcement. *International Journal of Control, Automation and Systems*, *22*(11), 3352-3363.

[14] Jian, Zhuozhu, et al. "Dynamic control barrier function-based model predictive control to safety-critical obstacle-avoidance of mobile robot." *2023 IEEE international conference on robotics and automation (ICRA)*. Ieee, 2023.

[15] Khan, S., Guivant, J., & Li, X. (2022). Design and experimental validation of a robust model predictive control for the optimal trajectory tracking of a small-scale autonomous bulldozer. *Robotics and Autonomous Systems*, *147*, 103903.

[16] Wabersich, K. P., Krishnadas, R., & Zeilinger, M. N. (2021). A soft constrained MPC formulation enabling learning from trajectories with constraint violations. *IEEE Control Systems Letters*, *6*, 980-985.

[17] Gracia, V., Krupa, P., Limon, D., & Alamo, T. (2024). Implementation of soft-constrained MPC for tracking using its semi-banded problem structure. *IEEE Control Systems Letters*, *8*, 1499-1504.

[18] Liu, Huajian, et al. "Optimization-based local planner for a nonholonomic autonomous mobile robot in semi-structured environments." *Robotics and Autonomous Systems* 171 (2024): 104565.

[19] Sun, Y., Yang, J., Zhao, D., Okonkwo, M. C., Zhang, J., Wang, S., & Liu, Y. (2024). Enhancing stability and safety: A novel multi-constraint model predictive control approach for forklift trajectory. *IET Cyber-Systems and Robotics*, *6*(4), e70004.

[20] Magalhães, André Chaves, and Luciano Cunha de Araujo Pimenta. "Distributed Model Predictive Control for Safety-Critical Obstacle Avoidance in Multi-Robot Systems with Social Preferences." *IFAC-PapersOnLine* 59.21 (2025): 56-61.

[21] Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. Automatica, 36(6), 789-814.

[22] Chen, H., & Allgöwer, F. (1998). A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, *34*(10), 1205-1217.
[23] Yang, H., Guo, M., Xia, Y., & Cheng, L. (2018). Trajectory tracking for wheeled mobile robots via model predictive control with softening constraints. *IET Control Theory & Applications*, *12*(2), 206-214.
[24] Zheng, W., & Zhu, B. (2021). Stochastic time-varying model predictive control for trajectory tracking of a wheeled mobile robot. *Frontiers in Energy Research*, *9*, 767597.