Safe Trajectory Planning with Bernstein Polynomials Under Control Barrier Function Constraints

Jungeun Lee[1,†], Seongjae Lee[1,†] and Jeong hwan Jeon[1,2,*]

***Abstract*— Ensuring safe trajectory planning of autonomous mobile robots in environments with static and dynamic obstacles is a critical challenge for real-world deployment. Existing approaches, such as nonlinear predictive control (NMPC) with control barrier functions (CBFs), can provide safety guarantees. However, their computational complexity scales with the prediction horizon, which can hinder real-world feasibility. We propose an NMPC algorithm that combines discrete-time CBFs with Bernstein polynomial parameterization. By constraining a constant number of Bernstein control points within a dynamically generated safe corridor, the entire trajectory is guaranteed to remain forward invariant within the safe set. It also significantly reduces computational complexity compared to conventional approaches. Simulation results in both static and dynamic obstacle environments demonstrate that it achieves a high success rate while maintaining similar safety margins to existing baselines. At the same time, computation time is reduced by between 60% and 90%, depending on the scenario, and it remains nearly constant even as the prediction horizon increases. These results demonstrate the efficiency, robustness, and scalability of the proposed approach for safe trajectory planning in diverse scenarios.**

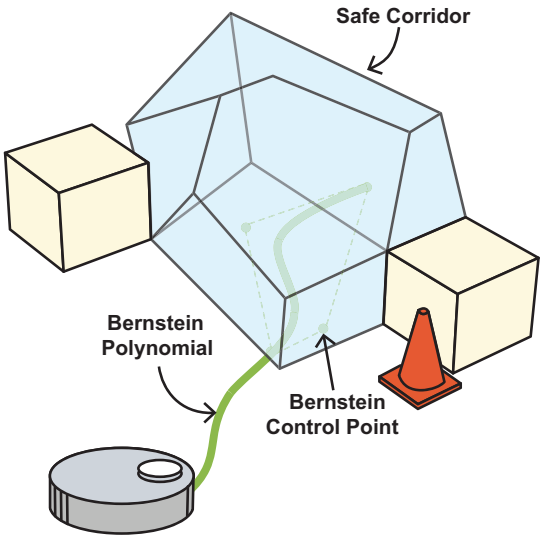


Fig. 1: Overview of the proposed NMPC approach. Safety is preserved by constraining Bernstein control points within a safe corridor, which guarantees forward invariance of the entire trajectory.

I. INTRODUCTION

Recent advances in mobile robotics are driving the integration of autonomous systems into human environments. These systems are extending their influence from improving industrial productivity to enhancing convenience in daily life. In particular, the adoption of autonomous mobile robots (AMRs) is accelerating in industrial sectors such as logistics and manufacturing. The global market for AMRs was estimated at approximately $4.07 billion in 2024 and is projected to grow to $9.56 billion by 2030, corresponding to a compound annual growth rate (CAGR) of 15.1% [1]. The proliferation extends to the domestic sphere. More than half of U.S. households use smart-home technologies, creating an ecosystem that enables autonomous appliances such as robotic vacuum cleaners. As robots become increasingly widespread in our lives, ensuring their safety has emerged as a critical challenge. In particular, mobile robots responsible for transporting people or stuff are required to navigate safely through environments containing both unpredictable dynamic obstacles and static obstacles.

To address collision avoidance, traditional methods such as the dynamic window approach (DWA) [2], timed elastic band (TEB) [3], and optimal reciprocal collision avoidance (ORCA) [4] have been widely used. DWA is a sampling-based method that searches for an optimal control input within the robot's dynamic constraints. In contrast, TEB [3] is an optimization-based approach that models the predicted trajectory as an elastic band, continuously deforming it to find a smooth and collision-free path. While these methods are effective in many scenarios, they lack guarantees of safety. These limitations have motivated the development of control barrier functions (CBFs) [5]. CBFs are founded on Lyapunov stability theory and are designed to ensure system safety by achieving forward invariance, which guarantees that the system state remains within a predefined safe set at all times. They have also been combined with model predictive control (MPC) to provide predictive safety and integrated with control Lyapunov functions (CLFs) to account for stability jointly. Furthermore, extending CBFs to discrete-time for practical implementation, known as DCBFs, has become a key area of research [5], [6]. Nevertheless, even with these extensions, standard CBFs still face several fundamental challenges. Especially for high-order dynamical systems where control inputs do not immediately affect the safety state, CBFs often result in overly cautious behaviors. In more severe cases, they lead to infeasibility, where no safe solution can be found. To enable robust handling of complex systems, high-order CBFs (HOCBFs) [7]–[9] have been developed by incorporating high-

[1] Department of Electrical Engineering, Ulsan National Institute of Science and Technology (UNIST), Ulsan, Republic of Korea
[2] Artificial Intelligence Graduate School, Ulsan National Institute of Science and Technology (UNIST), Ulsan, Republic of Korea
†These authors contributed equally.
*Corresponding author
Email addresses:{jungeunlee, lsjj1999, jhjeon}@unist.ac.kr

order derivative constraints. Furthermore, dynamic CBFs (D-CBFs) have been proposed to handle unpredictable moving obstacles [10]. A similar recent study [11] explores the use of Bernstein polynomial parameterizations for motion planning. This approach emphasizes certifying reachable sets that reflect the capabilities of low-level tracking controllers, enabling long-horizon reasoning in hierarchical systems. In contrast, our work pursues a different safety objective. We integrate discrete-time CBFs into an NMPC formulated with Bernstein control points as decision variables. It directly enforces forward invariance of the entire trajectory within a safe set, while reducing the number of constraints.

We propose a novel NMPC framework for trajectory planning that ensures safety while significantly reducing computational complexity. Conventional approaches [5]–[10] integrate CBFs into NMPC by enforcing safety constraints at every prediction step. In contrast, our method exploits the convex hull property of Bernstein polynomials to achieve safety with far fewer constraints. Restricting a constant number of Bernstein control points within a perception-driven safe corridor guarantees that the entire reference trajectory lies inside the safe set, as shown in Fig. 1. We reduce the number of safety constraints from $O(mN)$ to $O(m(d+1))$ by exploiting the convex hull property of Bernstein polynomials. Here, m denotes the number of half-spaces defining the safe corridor, N is the predictive horizon length, and d is the Bernstein polynomial degree. As a result, safety verification becomes independent of the prediction horizon length, yielding substantial computational savings while preserving safety and robustness. The main contributions of this paper are threefold:

- We introduce an NMPC formulation parameterized by Bernstein control points that enforces safety while ensuring forward invariance of the trajectory.
- Robust safety under bounded tracking errors is theoretically guaranteed by means of tightened safe sets.
- Experimental results show that our approach maintains performance on success rate and safety margin ratio, achieving a significant reduction in computation time compared to the conventional approaches, which integrate CBFs into NMPC [6], [7].

The remainder of this paper is organized as follows. Section II introduces the mathematical preliminaries. Section III details our proposed NMPC-based pipeline, focusing on how safety is formally guaranteed while reducing computational complexity through Bernstein control point parameterization. Section IV evaluates the proposed approach in diverse scenarios to demonstrate its efficiency and robustness. Finally, Section V concludes the paper and outlines directions for future research.

II. PRELIMINARIES

In this section, the concept of a CBF for discrete-time systems is introduced. Additionally, the Bernstein trajectory ensuring system safety is defined

Definition 2.1 (Discrete-Time Control Barrier Function [5]): Consider the discrete-time control system

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), \quad k \in \mathbb{N}_{\geq 0}, \tag{1}$$

where $\mathbf{x}_k \in X \subset \mathbb{R}^n$ is the state and $\mathbf{u}_k \in U \subset \mathbb{R}^m$ is the control input at time step k. Let $h : X \to \mathbb{R}$ be a continuous function that defines the safe set

$$\mathcal{S} := \{\mathbf{x} \in X | h(\mathbf{x}) \geq 0\}. \tag{2}$$

The continuous function h is a **discrete-time control barrier function (CBF)** for (1) with respect to the safe set (2) if there exists a class $\mathcal{K}$ function [12] α with $\alpha(r) < r$ for all $r > 0$ such that

$$h(f(\mathbf{x}_k, \mathbf{u}_k)) \geq h(\mathbf{x}_k) - \alpha(h(\mathbf{x}_k)), \quad \forall \mathbf{x}_k \in X. \tag{3}$$

Definition 2.2 (Safe Control Set [8]): For every state $\mathbf{x} \in X$, the **safe control set** is

$$C := \{\mathbf{u} \in U \mid h(f(\mathbf{x}, \mathbf{u})) \geq h(\mathbf{x}) - \alpha(h(\mathbf{x}))\}, \tag{4}$$

where α is a class $\mathcal{K}$ function with $\alpha(r) < r$ for all $r > 0$.

Definition 2.3 (Forward Invariance [7], [8]): For the system (1), a safe set S (2) is **forward invariant** if every trajectory that starts inside S never leaves it:

$$\mathbf{x}_0 \in S \Rightarrow \mathbf{x}_k \in S, \quad \forall k \in \mathbb{N}_{\geq 0}. \tag{5}$$

Remark 2.1: If h is a discrete-time CBF and, at every step k, the control input is in the safe control set $\mathbf{u}_k \in C(\mathbf{x}_k)$, then the safe set S is forward invariant.

Proof: For the system (1), assume $\mathbf{x}_0 \in S$ so that $h(\mathbf{x}_0) \geq 0$. Suppose $h(\mathbf{x}_k) \geq 0$ and choose $\mathbf{u}_k \in C(\mathbf{x}_k)$. Then, $h(\mathbf{x}_{k+1}) \geq h(\mathbf{x}_k) - \alpha(h(\mathbf{x}_k)) \geq 0$ since $\alpha(r) < r$ for all $r > 0$. Hence $\mathbf{x}_{k+1} \in S$. By induction, $h(\mathbf{x}_k) \geq 0$ for all $k \in \mathbb{N}_{\geq 0}$. Consequently, the safe set S is forward invariant. ■

Definition 2.4 (Discrete-Time Bernstein Trajectory): Let $N, d \in \mathbb{N}$. Given control points $\{\mathbf{p}_i\}_{i=0}^{d} \subset \mathbb{R}^n$, define the Bernstein basis

$$B_i^d(\tau) = \binom{d}{i} \tau^i (1-\tau)^{d-i}. \tag{6}$$

The **degree-d Bernstein trajectory** sampled over the prediction horizon N is

$$\mathbf{x}_k = \sum_{i=0}^{d} B_i^d(\tau_k)\, \mathbf{p}_i, \tag{7}$$

where τ_k is k/N for all $k \in \{0, \ldots, N\}$. Each $\mathbf{x}_k$ is a convex combination of the control points since $B_i^d(\tau_k) \geq 0$ and $\sum_{i=0}^{d} B_i^d(\tau_k) = 1$. In particualr, $\mathbf{x}_0 = \mathbf{p}_0$ and $\mathbf{x}_N = \mathbf{p}_d$.

III. METHODOLOGY

This section details the formulation of a nonlinear model predictive control (NMPC) problem that ensures the trajectory remains collision-free. To achieve this, our pipeline, shown in Fig. 2 and Algorithm 1, follows a flow from perception to control. At each time step, our framework generates an occupancy grid map (OGM) by discretizing the sensor data into a grid of occupied and free cells. The A* algorithm then uses this map to plan a global path, omitting replanning for efficiency when the previous remains collision-free. Subsequently, a dynamic safe corridor is constructed from nearby obstacles to define a safe region, as shown in Fig. 3. Each safe corridor is a convex region, defined as the intersection of half-spaces, $\mathcal{S} := \bigcap_{i=1}^{m} \left\{\mathbf{x} | \mathbf{a}_i^T \mathbf{x} + \mathbf{b}_i \geq 0\right\}$. The corresponding

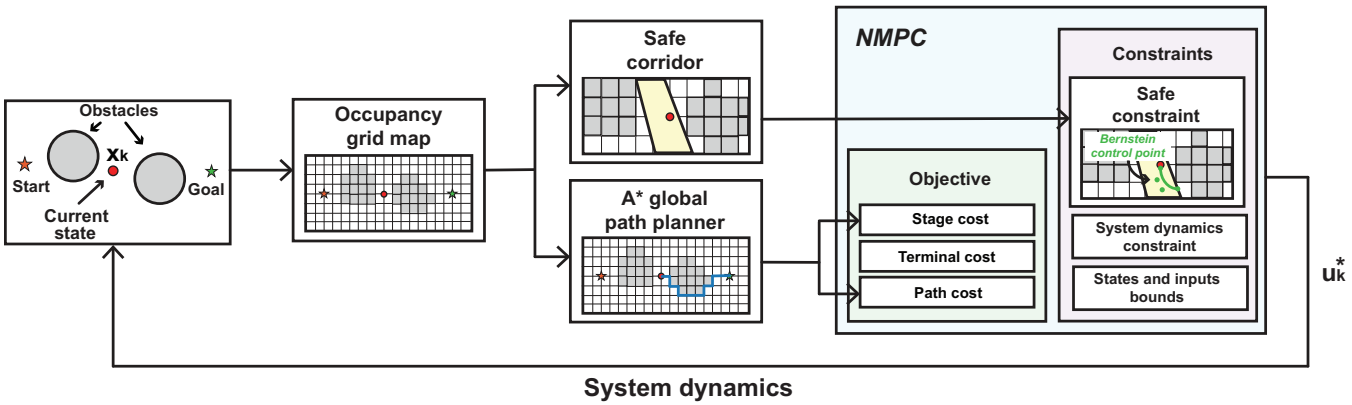


Fig. 2: NMPC-based collision-free trajectory planning and control pipeline. This framework ensures the entire reference trajectory remains in a safe set by constraining its Bernstein control points within a perception-driven corridor.

Algorithm 1 NMPC for collision-free trajectory planning and control with Bernstein control points.

Require: Sensor data stream $\mathbf{Z}_k$, Goal state $\mathbf{g}$, System dynamics $\mathbf{f}$, Maximum number of half-spaces defining the safe corridor m

while robot not at goal **do**
  Initialize current state $\mathbf{x}_0$
  Get an occupancy grid map $\mathbf{O}_k$ from $\mathbf{Z}_k$ and $\mathbf{x}_k$
  Get a global path π_{global} from $\mathbf{O}_k$ and $\mathbf{x}_k$ using the A* algorithm
  Construct the safe corridor defined by $\mathcal{S} = \bigcap_{i=1}^{m} \{\mathbf{x} | \mathbf{a}_i^T \mathbf{x} + \mathbf{b}_i \geq 0\}$ from $\mathbf{O}_k$ and $\mathbf{x}_k$
  Get an optimal control $\mathbf{u}_k^* \leftarrow \text{NMPC}(\mathbf{x}_k, \pi_{\text{global}}, \mathcal{S})$
  Apply the control input $\mathbf{u}_k^*$ and update the state $\mathbf{x}_k$
end while

procedure NMPC($\mathbf{x}_k$, π_{global}, $\mathcal{S}$)
  Define the decision variables X, U, and P_{ctrl}
  Minimize the cost (8) over the decision variables
  subject to
    system dynamics $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$
    state and input bounds
    safety constraint (19)
  return first optimal control input $\mathbf{u}_k^*$
end procedure

discrete-time CBF is then given by $h_i(\mathbf{x}) := \mathbf{a}_i^T \mathbf{x} + \mathbf{b}_i$, with the safety condition $h_i(\mathbf{x}) \geq 0$ for all i. Finally, an optimal control problem is solved to find the control input that confines the system's state to the convex regions defined by the safe corridors. The remainder of this section will delve into our optimal control problem, which is formulated to find a safe local path and the corresponding control input required to follow it. For clarity, the formulations in this section are presented at the first time step, but they are recursively applied at each subsequent time step, as outlined in Algorithm 1.

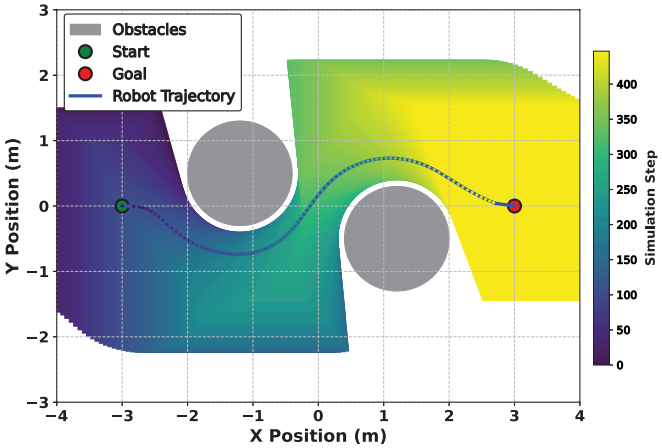


Fig. 3: Safe corridors accumulated over time steps. The safe corridors are dynamically reshaped by the mobile robot's position relative to nearby obstacles, ensuring safety throughout the trajectory with a predefined safety margin of 0.1 m.

A. Objective

The objective function achieves a balance among several competing goals, including state tracking, control effort, and input smoothness. Furthermore, it includes penalty terms to encourage global path guidance, reaching the goal, and minimizing slack variables. It is formulated to find the optimal control input which minimizes the cost function J at the first time step:

$$J(X, U, P_{\text{ctrl}}) = \sum_{k=0}^{N-1} L_k(\mathbf{x}_k, \mathbf{u}_k) + L_T(\mathbf{x}_N) + L_P(P_{\text{ctrl}}), \quad (8)$$

where $X = \{\mathbf{x}_0, ..., \mathbf{x}_N\}$ is the predicted state sequence, $U = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$ is the control input sequence, and $P_{\text{ctrl}} = \{\mathbf{p}_0, ..., \mathbf{p}_d\}$ is the set of Bernstein control points of a degree-d Bernstein polynomial. As shown in (8), the objective function is composed of three cost components.

- **Stage cost** (L_k) The term represents the sum of penalties for undesirable behaviors at each step k over the prediction horizon N. It is formulated as

$$L_k(\mathbf{x}_k, \mathbf{u}_k) = J_{\text{track},k} + J_{\text{ctrl},k} + J_{\text{smooth},k}. \quad (9)$$

The state tracking cost $J_{\text{track},k}$ penalizes deviations from a Bernstein reference trajectory parameterized by Bernstein control points:

$$J_{\text{track},k} = w_k \left(\mathbf{x}_k - \mathbf{x}_{r,k}\right)^T \mathbf{Q} \left(\mathbf{x}_k - \mathbf{x}_{k,r}\right), \tag{10}$$

where $\mathbf{x}_k$ denotes the local trajectory, and $\mathbf{x}_{r,k}$ is its reference. The w_k is a time-varying positive weight, and $\mathbf{Q}$ is a constant semi-definite weight. The control effort cost $J_{\text{ctrl},k}$ is a quadratic penalty on the deviation of the control input from its reference:

$$J_{\text{ctrl},k} = \left(\mathbf{u}_k - \mathbf{u}_{r,k}\right)^T \mathbf{R} \left(\mathbf{u}_k - \mathbf{u}_{r,k}\right), \tag{11}$$

where $\mathbf{u}_k$ and $\mathbf{u}_{r,k}$ are the control input and its reference at step k, respectively. The weighting matrix $\mathbf{R} \succeq 0$ sets the relative penalty on each input channel. The input smoothness cost $J_{\text{smooth},k}$ penalizes changes in the control between consecutive steps:

$$J_{\text{smooth},k} = w_{\Delta u} \left|\left|\mathbf{u}_k - \mathbf{u}_{k-1}\right|\right|_2^2, \quad k = 1, ..., N-1, \tag{12}$$

where $w_{\Delta u} \geq 0$ is a constant weight.

- **Terminal cost (L_T)** The terminal cost L_T is a Mayer term that strongly enforces convergence to the final reference state $\mathbf{x}_{r,N}$ at the end of the horizon:

$$L_T(\mathbf{x}_N) = \left(\mathbf{x}_N - \mathbf{x}_{r,N}\right)^T \mathbf{P} \left(\mathbf{x}_N - \mathbf{x}_{r,N}\right), \tag{13}$$

where $\mathbf{P} \succeq 0$.

- **Path cost (L_P)** The path cost L_P serves to optimize both the shape and guidance of the Bernstein reference trajectory:

$$L_P\left(P_{\text{ctrl}}\right) = J_{\text{smooth}} + J_{\text{wp}} + J_{\text{goal}}. \tag{14}$$

The Bernstein path smoothness cost J_{smooth} penalizes the squared distance between consecutive control points to ensure a smooth path:

$$J_{\text{smooth}} = w_b \sum_{i=0}^{d-1} \left|\left|\mathbf{p}_{i+1} - \mathbf{p}_i\right|\right|_2^2, \tag{15}$$

where d is the degree of the Bernstein curve, and $\mathbf{p}_i \in \mathbb{R}^n$ denotes the i-th control point of the Bernstein curve. The waypoint guidance cost J_{wp} guides the Bernstein trajectory towards the waypoints sampled by the A* algorithm:

$$J_{\text{wp}} = w_{\text{wp}} \sum_{i=0}^{d} \min_j \left\|\mathbf{p}_{i,\text{pos}} - \mathbf{w}_j\right\|_2, \tag{16}$$

where $\mathbf{p}_{i,\text{pos}}$ is the position components of the i-th control point, and $\mathbf{w}_j$ is the j-th waypoint. The global attraction cost strongly attracts the final control point to the goal.

$$J_{\text{goal}} = w_{\text{goal}} \left\|\mathbf{p}_{d,\text{pos}} - \mathbf{g}\right\|_2, \tag{17}$$

where $\mathbf{g}$ is the goal position. The weights w_b, w_{wp}, and w_{goal} are constants associated with (15), (16), and (17), respectively.

B. Constraints

The constraints ensure a physically feasible and safe trajectory by enforcing four conditions: system dynamics, bounds on states and control inputs, corridor-based safety, and geometric restrictions on the Bernstein control points. Specifically, the predicted trajectory must satisfy the discrete-time system dynamics $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$, with all states and inputs remaining within the predefined bounds $\mathbf{x}_l \leq \mathbf{x}_k \leq \mathbf{x}_u$ and $\mathbf{u}_l \leq \mathbf{u}_k \leq \mathbf{u}_u$. Furthermore, the initial predicted state $\mathbf{x}_0$ must coincide with the robot's current measured state. In addition to the physical and boundary constraints, safety is a critical requirement. The reference trajectory $\left\{\mathbf{x}_{r,k}\right\}_{k=0}^{N}$ parameterized by Bernstein control points is

$$\mathbf{x}_{r,k} = \sum_{j=0}^{d} B_j^d(\tau_k)\mathbf{p}_j, \tag{18}$$

where τ_k is k/N for all $k \in \{0, \ldots, N\}$. The controller ensures safety by restricting the system to a predefined safe corridor, mathematically expressed as the intersection of multiple linear inequalities. If the CBF is defined by m linear inequalities $\mathbf{a}_i^T\mathbf{x} + \mathbf{b}_i \geq 0$, $i = 1, ..., m$, the constraints on each control point $\mathbf{p}_j$ are expressed in the relaxed form:

$$h_i(\mathbf{p}_j) - (1-\gamma)h_i(\mathbf{x}_0) \geq 0, \quad \forall i \in \{1, \ldots, m\}, \forall j \in \{0, \ldots, d\}. \tag{19}$$

Here, $h_i(\cdot)$ is the CBF associated with the i-th half-space defining the corridor. The parameter $\gamma \in [0, 1)$ plays the same role as the class-$\mathcal{K}$ function $\alpha(\cdot)$ in Definition 2.2, serving as a relaxation factor that softens the constraint near the boundary to enhance feasibility. Additionally, constraint (19) guarantees forward invariance of the Bernstein trajectory with respect to the safe corridor. Since the first control point coincides with the current state $\mathbf{p}_0 = \mathbf{x}_{r,0} = \mathbf{x}_0$, feasibility at the initial condition implies that all subsequent control points remain within the safe corridor. Due to the convex hull property of Bernstein polynomials, this further ensures that the entire trajectory is contained within the safe corridor.

Theorem 3.1 (Safety of Bernstein Trajectory): Let the safe corridor be defined as

$$\mathcal{S} \coloneqq \bigcap_{i=1}^{m} \left\{\mathbf{x} | \mathbf{a}_i^T\mathbf{x} + \mathbf{b}_i \geq 0\right\}. \tag{20}$$

If the initial state coincides with the first control points, i.e., $\mathbf{p}_0 = \mathbf{x}_0$, and satisfies $h_i(\mathbf{x}_0) \geq 0$ for all $i = 1, \cdots, m$, then the entire Bernstein trajectory remains within the safe corridor:

$$\mathbf{x}_{r,k} \in \mathcal{S}, \quad \forall k = 0, \ldots, N. \tag{21}$$

Proof: By assumption, the initial state satisfies the safe corridor constraints, $h_i(\mathbf{x}_0) \geq 0$ for all i. From (19), each control point $\mathbf{p}_j$ of the Bernstein trajectory must satisfy

$$h_i(\mathbf{p}_j) \geq 0, \quad \forall j = 0, \ldots, d. \tag{22}$$

Since each $h_i(\cdot)$ is affine, if all control points satisfy (22), then any convex combination of them must also satisfy $h_i(\mathbf{x}_{r,k}) \geq 0$ due to the convex hull property of Bernstein polynomials. Thus, we have $\mathbf{x}_{r,k} \in \mathcal{S}$ for all k. The entire Bernstein trajectory remains within the safe corridor. ■

Definition 3.1 (Tightened Safe Set under Tracking Error): Let $\mathbf{x}_k$ and $\mathbf{x}_{r,k}$ denote the actual and reference Bernstein trajectories at step k, respectively. Assume the tracking error is uniformly bounded:

$$\left\|\mathbf{x}_k - \mathbf{x}_{r,k}\right\|_2 \leq \epsilon, \quad \forall k = 0, ..., N. \tag{23}$$

The ϵ-tightened safe set is

$$\mathcal{S}_{\ominus\epsilon} := \{\mathbf{x} \in \mathbb{R}^n | \mathbb{B}_\epsilon(\mathbf{x}) \subseteq \mathcal{S}\}, \tag{24}$$

where $\mathbb{B}_\epsilon(\mathbf{x}) := \{\mathbf{y} \in \mathbb{R}^n | ||\mathbf{y} - \mathbf{x}||_2 \leq \epsilon\}$ denotes the closed Euclidean ball of radius ϵ centered at $\mathbf{x}$.

Theorem 3.2 (Robust Safety via Tightening): Suppose the Bernstein reference trajectory $\{\mathbf{x}_{r,k}\}_{k=0}^{N}$ lies entirely in the tightened safe set $\mathcal{S}_{\ominus\epsilon}$. If the tracking error is uniformly bounded by $\left\|\mathbf{x}_k - \mathbf{x}_{r,k}\right\|_2 \leq \epsilon$ for all k, then the actual trajectory also remains in the original safe set $\mathcal{S}$:

$$\mathbf{x}_k \in \mathcal{S}, \quad \forall k = 0, \ldots, N. \tag{25}$$

Proof: By Definition 3.1, we have $\mathbb{B}_\epsilon(\mathbf{x}_{r,k}) \subseteq \mathcal{S}$ for every reference point $\mathbf{x}_{r,k}$. Since the tracking error is bounded by $\left\|\mathbf{x}_k - \mathbf{x}_{r,k}\right\|_2 \leq \epsilon$, it follows that $\mathbf{x}_k \in \mathbb{B}_\epsilon(\mathbf{x}_{r,k})$. Therefore, $\mathbf{x}_k \in \mathcal{S}$ for all k, which establishes the robust safety guarantee. ■

Remark 3.1 (Practical Implementation): While Theorem 3.2 establishes robust safety via the ϵ-tightened safe set, in practice, we combine this tightening with the relaxed CBF condition (19) to improve feasibility. Specifically, the implemented constraint takes the form

$$h_i(\mathbf{p}_j) - (1-\gamma)h_i(\mathbf{x}_0) \;\geq\; \epsilon, \quad \forall i, j. \tag{26}$$

This modification enforces both a safety margin ϵ against tracking errors and a relaxation factor γ to mitigate infeasibility near the corridor boundary.

A key advantage of the proposed approach is that safety verification needs to be performed only at the Bernstein control points, rather than at every step of the prediction horizon. In conventional methods [6]–[9], m safety constraints are imposed at each of the N prediction steps, resulting in a total of mN constraints and a computational burden that grows linearly with the horizon length. In contrast, our Bernstein trajectory requires only $m(d+1)$ constraints, where d is the degree of the Bernstein polynomial and typically $d \ll N$. This reduction transforms the safety check into a constant-size problem independent of the horizon length, thereby yielding a substantial computational saving while still guaranteeing forward invariance of the entire trajectory through the convex hull property.

IV. EXPERIMENTAL RESULTS

In this section, we demonstrate our approach through simulation experiments. Specifically, we show that the proposed NMPC framework ensures forward-invariant safety while substantially lowering computational costs. The results highlight not only improved feasibility in complex environments but also significant efficiency gains over two baselines. We refer to our proposed approach as BP-NMPC-CBF, which stands for Bernstein polynomial parameterized nonlinear model predictive control with control barrier functions.

A. System model

1) System dynamics: To validate the proposed approach, numerical case studies were carried out on a unicycle model in discrete time. The state vector is $\mathbf{x}_k = \left[l_{x,k},\, l_{y,k},\, \theta_k,\, v_k\right]^T$, which consists of position, heading, and linear speed. The control input is $\mathbf{u}_k = \left[\omega_k,\, a_k\right]^T$, denoting angular velocity and linear acceleration. With a sampling period of $\Delta t > 0$, the system evolves according to the following state transition equation:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \triangleq \begin{bmatrix} l_{x,k} + v_k \cos\theta_k \Delta t \\ l_{y,k} + v_k \sin\theta_k \Delta t \\ \theta_k + \omega_k \Delta t \\ v_k + a_k \Delta_t \end{bmatrix}. \tag{27}$$

2) System configuration: All experiments are conducted in an 8 m × 8 m workspace from start $\mathbf{x}_0 = \left[-3.0,\, 0.0,\, 0.0,\, 0.5\right]^T$ to goal state $\mathbf{g} = \left[3.0,\, 0.0,\, 0.0,\, 0.0\right]^T$. In the static scenario, the radii of two circular obstacles are randomly generated for each run, resulting in a narrow passage with varying clearance between the obstacles. Weighting matrices are $\mathbf{Q} = \mathbf{P} = \mathrm{diag}(10,\, 10,\, 10,\, 10)$, $\mathbf{R} = \mathrm{diag}(1,\, 1)$, and $\mathbf{S} = \mathrm{diag}(1000,\, 1000)$. In the dynamic scenario, two circular obstacles oscillate, and each center moves along the x-axis, y-axis, or the 45° diagonal. Amplitude and frequency are independently sampled as $A \sim [0.3,\, 0.5]$ m and $f \sim [0.08,\, 0.18]$ Hz, respectively. A trial is considered successful if the agent's position is within 0.10 m of the goal. It is considered a failure if a collision occurs or if the time limit is exceeded. Each experiment was repeated for 100 randomized trials, where obstacle radii, oscillation amplitude, and frequencies were uniformly sampled within the specified ranges. This setup ensures statistical robustness and captures diverse environmental realizations for evaluation.

B. Baselines

To evaluate our approach, we compared our method against two baselines as follows.

- **NMPC-DCBF** ($m_{\mathbf{cbf}} = 1$) **[6]** The baseline integrates discrete-time control barrier functions (DCBFs) into the NMPC framework. Safety is enforced by imposing CBF constraints at every step of the prediction horizon.
- **NMPC-DCBF** ($m_{\mathbf{cbf}} = 2$) **[6], [7]** The extension incorporates high-order discrete-time CBFs (HOCBFs) to handle constraints with higher relative degrees [13], thereby ensuring forward invariance under more complex system dynamics.

The parameter m_{cbf} denotes the relative degree of the safety constraint and determines which baseline is applied to the unicycle model. For standard position-based constraints such as obstacle avoidance, the relative degree is 1 ($m_{\mathrm{cbf}} = 1$). In this case, the control input directly influences the constraint state. In contrast, the relative degree becomes 2 ($m_{\mathrm{cbf}} = 2$) when the safety condition involves higher-order dynamics such as

TABLE I: Performance comparison of control methods in a narrow passage formed by two static obstacles. The results are reported as the average of the mean and standard deviation over 100 trials.

	Prediction horizon	Success ↑ rate (%)	Computation time (s) ↓	Safety margin ratio ↑
NMPC-DCBF ($m_{\text{cbf}} = 1$) [6]	N = 30	1.00	0.039 ± 0.018	0.977 ± 0.071
	N = 20	1.00	0.026 ± 0.100	0.985 ± 0.181
	N = 10	1.00	0.025 ± 0.010	0.979 ± 0.083
NMPC-DCBF ($m_{\text{cbf}} = 2$) [6], [7]	N = 30	1.00	0.112 ± 0.044	0.979 ± 0.049
	N = 20	1.00	0.071 ± 0.031	0.986 ± 0.015
	N = 10	1.00	0.033 ± 0.014	0.997 ± 0.221
BP-NMPC-CBF (Ours)	N = 30	1.00	0.018 ± 0.008	0.990 ± 0.044
	N = 20	1.00	0.009 ± 0.005	0.986 ± 0.058
	N = 10	1.00	0.013 ± 0.004	0.961 ± 0.111

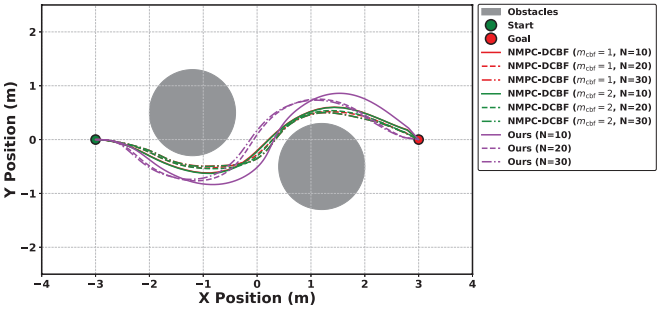


Fig. 4: Trajectory comparisons between baseline NMPC-DCBF methods and the proposed approach with varying relative degrees m_{cbf} and horizon lengths N. This sample scenario, in which the robot must navigate through a narrow passage, demonstrates which algorithms maintain a safe trajectory until the goal is reached.

acceleration limits. To ensure a fair comparison, all methods share the same solver settings and cost structure, with the only difference being the safety constraints. This design allows us to focus on the effect of our Bernstein-based CBF.

C. Results

To evaluate the performance of different control methods in Table I and II, we consider three metrics. The success rate is defined as the ratio of successful trials in which the agent reaches the goal without collision. The computation time measures the average computation time per decision step, where lower values indicate higher real-time feasibility. The safety margin ratio is computed by normalizing the distance between the agent and the nearest obstacle with respect to the predefined safety margin of 0.1 m. The safe margin corresponds to the ϵ defined in (23) and (26). Figs. 6 and 7 indicate that a value of 0.1 m is appropriate to use as the safety margin. All reported results were obtained on a MacBook Pro equipped with an M3 chip and 16 GB of RAM.

The trajectories of our algorithm, shown in Fig. 4 and Fig. 5, correspond to tracking a reference path parameterized by a Bernstein polynomial of degree 3. In the narrow passage scenario shown in Fig. 4, all methods achieve a 100% success rate, confirming consistent safety performance (Table I). The most notable difference is observed in computational efficiency. The NMPC-DCBF baseline with $m_{\text{cbf}} = 2$ exhibits the highest computational cost, requiring up to 0.112 seconds per iteration at $N = 30$. In contrast, our method requires only

TABLE II: Performance comparison of control methods in a dynamic narrow passage formed by two oscillating obstacles. The results are reported as the average of the mean and standard deviation over 100 trials.

	Prediction horizon	Success ↑ rate (%)	Computation time (s) ↓	Safety margin ratio ↑
NMPC-DCBF ($m_{\text{cbf}} = 1$) [6]	N = 30	1.00	0.044 ± 0.021	0.979 ± 0.090
	N = 20	0.99	0.047 ± 0.021	0.968 ± 0.103
	N = 10	0.99	0.024 ± 0.007	0.991 ± 0.051
NMPC-DCBF ($m_{\text{cbf}} = 2$) [6], [7]	N = 30	1.00	0.074 ± 0.030	0.966 ± 0.110
	N = 20	1.00	0.068 ± 0.027	0.969 ± 0.101
	N = 10	1.00	0.033 ± 0.012	0.980 ± 0.083
BP-NMPC-CBF (Ours)	N = 30	1.00	0.018 ± 0.053	0.976 ± 0.104
	N = 20	1.00	0.018 ± 0.040	0.965 ± 0.111
	N = 10	1.00	0.011 ± 0.006	0.990 ± 0.085

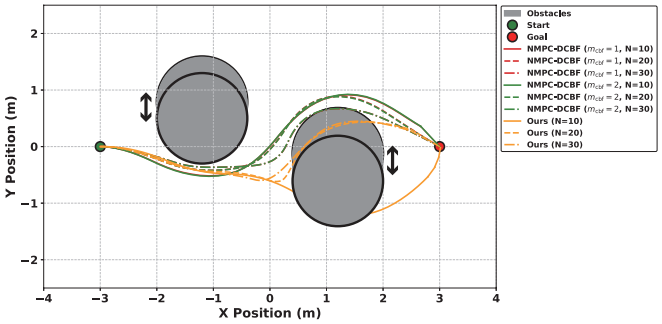


Fig. 5: Trajectory comparisons between the baseline NMPC-DCBF methods and the proposed approach are shown for varying relative degrees m_{cbf} and horizon lengths N. In the oscillating-obstacle scenario, the dynamic narrow passage evaluates each controller's ability to maintain safe navigation until the goal is reached. Each trajectory is collision-free, and the oscillating obstacle motions are depicted together for clarity.

about 0.018 seconds at $N = 30$. Compared to the $m_{\text{cbf}} = 1$ baselines, our method reduces computation time by about 65% – 70%, and achieves nearly 85% – 90% reduction relative to the $m_{\text{cbf}} = 2$ baselines. The safety margin ratio achieved by our method ranges from 0.96 to 0.99, which is comparable to that of the baselines. Fig. 4 compares the trajectories generated by the baseline NMPC-DCBF methods and our proposed approach for different relative degrees m_{cbf} and horizon lengths N. Additionally, we evaluate the evolution of the system states, including the position x and y, heading angle θ, and linear speed v, to analyze the dynamic behavior of different controllers. As shown in Fig. 8, the NMPC-DCBF baselines with $m_{\text{cbf}} = 1$ and $m_{\text{cbf}} = 2$ follow similar patterns, accelerating gradually from an initial velocity of 0.5 m/s while making more aggressive heading adjustments to remain within the safe corridor. In contrast, our method initiates at a higher velocity (about 1.4 m/s) and gradually decelerates toward the goal, leading to smoother and more stable heading adjustments. These results show that while all methods ensure safety, our approach converges faster and more smoothly, demonstrating computational efficiency and robust performance across different prediction horizons.

In the dynamic narrow passage scenario with oscillating obstacles as shown in Fig. 5, all methods achieve nearly perfect success rates, as summarized in Table II, confirming that

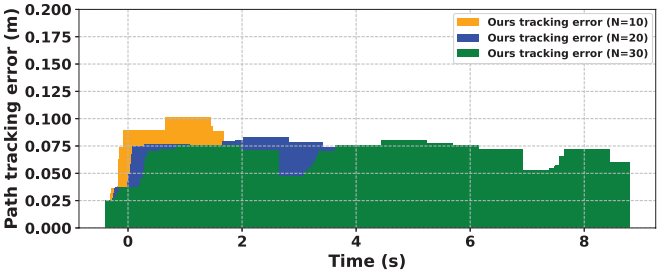


Fig. 6: Tracking errors of the proposed BP-NMPC-CBF under static obstacles forming a narrow passage for horizon lengths $N = 10$, 20, and 30.

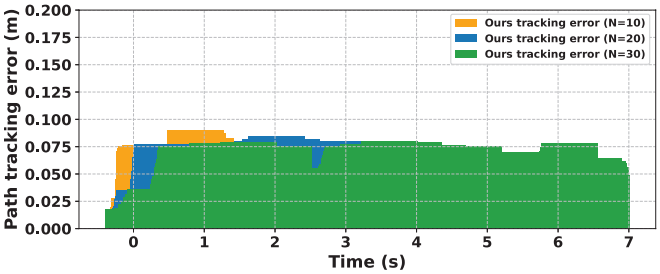
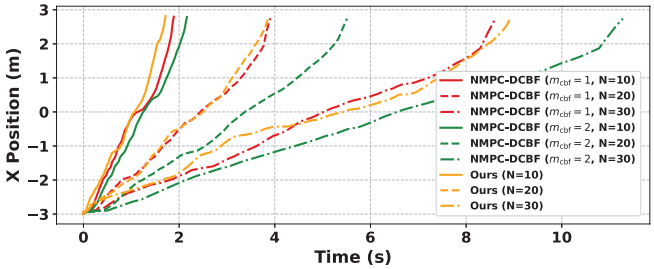


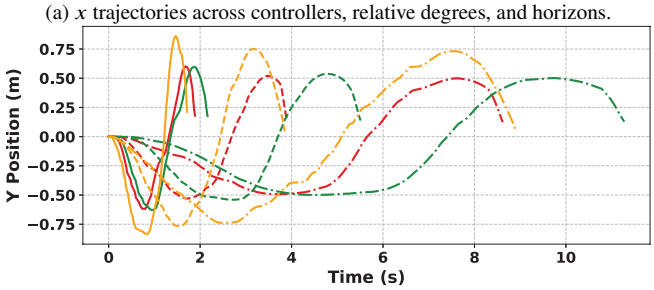
Fig. 7: Tracking errors of the proposed BP-NMPC-CBF under oscillating obstacles forming a dynamic narrow passage for horizon lengths $N = 10$, 20, and 30.

safety was consistently maintained. In this environment, the passage width continuously expands and contracts, testing each controller's ability to maintain safe trajectory planning until the goal is reached. Depending on m_{cbf} and N, the NMPC-DCBF baselines require computation times ranging from 0.024 to 0.074 seconds per iteration, whereas our method reduces this to between 0.011 and 0.018 seconds. This corresponds to a reduction of about 60% to 75% in computation time compared to the $m_{\text{cbf}} = 1$ baselines and nearly 80% relative to $m_{\text{cbf}} = 2$ baselines. At the same time, the safety margin ratios of our method remain comparable to those of the baselines, showing that the efficiency gains are achieved without compromising safety. In Fig. 9, we evaluate the state trajectories of the controllers within a dynamic narrow passage scenario created by two oscillating obstacles. The NMPC-DCBF baselines with $m_{\text{cbf}} = 1$ and $m_{\text{cbf}} = 2$ both gradually accelerate from the initial velocity of 0.5 m/s while frequently adjusting the heading angle to remain within the safe corridor formed by the moving obstacles. As the prediction horizon N increases, these adjustments become more conservative, leading to larger variations in the heading angle. In contrast, our method smoothly decelerates as the robot approaches the goal, resulting in a more stable trajectory and reduced oscillations in the heading angle. These results confirm that our approach achieves faster yet smoother convergence, demonstrating both robust performance and improved computational efficiency even under dynamic obstacle conditions.

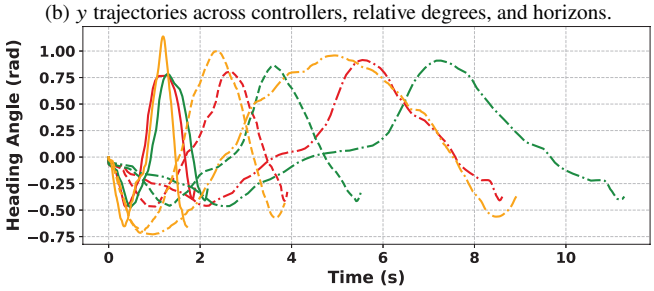
Fig. 10 compares controller computation times across different Bernstein polynomial degrees under static and oscillating obstacle environments. In the static case, as shown in Fig. 10a, the average computation times are approximately



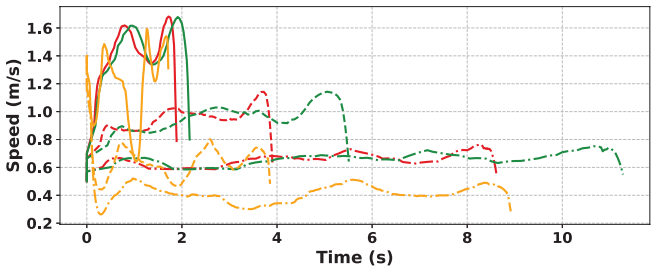
(a) x trajectories across controllers, relative degrees, and horizons.



(b) y trajectories across controllers, relative degrees, and horizons.



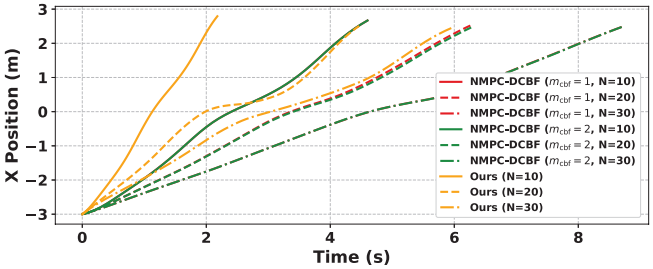
(c) Heading θ trajectories for different controllers, relative degrees, and horizons.



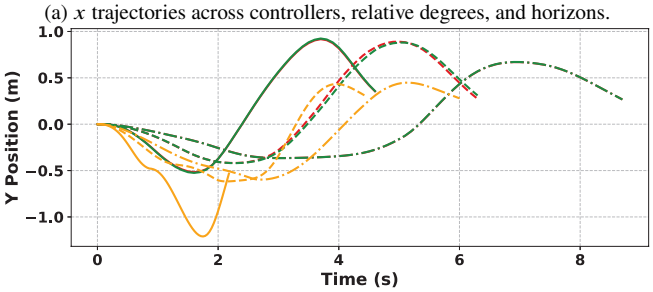
(d) Speed trajectories for different controllers, relative degrees, and horizons.

Fig. 8: Comparison of the state trajectories (x, y, θ, v) between the baselines and the proposed approach across relative degrees m_{cbf} and prediction horizons N. Colors represent controller types (orange: ours, red: $m_{\text{cbf}} = 1$, green: $m_{\text{cbf}} = 2$), and line styles denote horizon lengths (solid: $N = 10$, dashed: $N = 20$, dash-dot: $N = 30$).

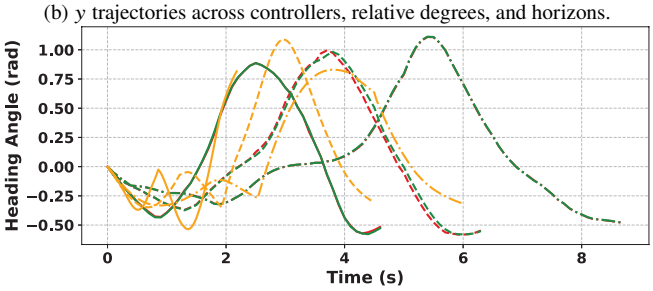
0.017 seconds, 0.020 seconds, and 0.024 seconds for degrees 3, 4, and 5, with standard deviations around $0.010 - 0.012$ seconds. This suggests that higher polynomial degrees result in only a modest increase in computation time, while variability remains consistently low. In contrast, in the oscillating obstacle case, as shown in Fig. 10b, the averages rise gradually from approximately 0.024 to 0.031 seconds, but the standard deviations are considerably larger ($0.066 - 0.078$ seconds), highlighting the greater uncertainty introduced by dynamic environmental changes.



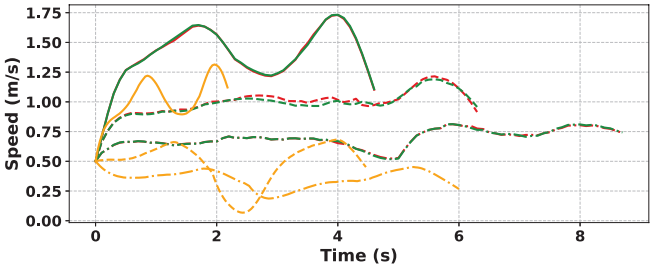
(a) x trajectories across controllers, relative degrees, and horizons.



(b) y trajectories across controllers, relative degrees, and horizons.

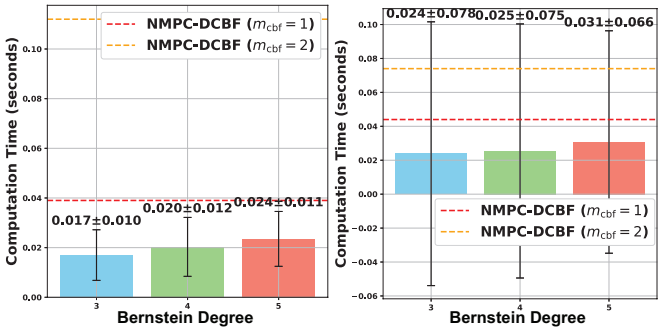


(c) Heading θ trajectories for different controllers, relative degrees, and horizons.



(d) Speed trajectories for different controllers, relative degrees, and horizons.

Fig. 9: State trajectories (x, y, θ, v) for the baselines and the proposed approach in the dynamic narrow passage scenario with oscillating obstacles. Colors indicate controller types (orange: proposed, red: $m_{\text{cbf}} = 1$, green: $m_{\text{cbf}} = 2$), and line styles represent horizon lengths (solid: $N = 10$, dashed: $N = 20$, dash-dot: $N = 30$).



(a) Static obstacles environment for different Bernstein degrees.

(b) Oscillating obstacles environment for different Bernstein degrees.

Fig. 10: Comparison of controller computation time distributions across different Bernstein degrees in two environments. The results are how the proposed approach scales with the Bernstein trajectory degree under varying environmental changes.

V. CONCLUSIONS

The paper proposes a nonlinear model predictive control (NMPC) algorithm parameterized by Bernstein control points with CBF constraints, termed BP-NMPC-CBF. The central contribution lies in reducing the number of safety constraints. This reduction decouples safety verification from the prediction horizon length, yielding significant computational savings without compromising safety guarantees.

Nevertheless, the study is limited to simulation-based evaluations on a unicycle model. While these case studies provide valuable insights into algorithmic feasibility, real-world robotic systems face additional challenges, including model uncertainties, actuator limits, sensing noise, and communication delays. Addressing these issues remains an essential step toward practical deployment.

REFERENCES

[1] Grand View Research, "Autonomous mobile robots market size, share & trends analysis report," https://www.grandviewresearch.com/industry-analysis/autonomous-mobile-robots-market, Jan 2024, accessed: 2025-09-25.

[2] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[3] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*. VDE, 2012, pp. 1–6.

[4] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. Int. Symp. Robotics Research (ISRR)*, 2011, pp. 3–19.

[5] M. Ahmadi, A. Singletary, J. W. Burdick, and A. D. Ames, "Safe policy synthesis in multi-agent pomdps via discrete-time barrier functions," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE Press, 2019, p. 4797–4803.

[6] J. Zeng, B. Zhang, Z. Li, and K. Sreenath, "Safety-critical control using optimal-decay control barrier function with guaranteed point-wise feasibility," in *2021 American Control Conference (ACC)*, 2021, pp. 3856–3863.

[7] Y. Xiong, D.-H. Zhai, M. Tavakoli, and Y. Xia, "Discrete-time control barrier function: High-order case and adaptive case," *IEEE Transactions on Cybernetics*, vol. 53, no. 5, pp. 3231–3239, 2022.

[8] S. Liu, J. Zeng, K. Sreenath, and C. A. Belta, "Iterative convex optimization for model predictive control with discrete-time high-order control barrier functions," in *2023 American Control Conference (ACC)*, 2023, pp. 3368–3375.

[9] S. Liu, Y. Mao, and C. A. Belta, "Safety-critical planning and control for dynamic obstacle avoidance using control barrier functions," in *2025 American Control Conference (ACC)*. IEEE, 2025, pp. 348–354.

[10] Z. Jian, Z. Yan, X. Lei, Z. Lu, B. Lan, X. Wang, and B. Liang, "Dynamic control barrier function-based model predictive control to safety-critical obstacle-avoidance of mobile robot," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3679–3685.

[11] N. Csomay-Shanklin and A. D. Ames, "Bézier reachable polytopes: Efficient certificates for robust motion planning with layered architectures," in *2025 American Control Conference (ACC)*, 2025, pp. 5052–5059.

[12] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.

[13] M. Sun and D. Wang, "Initial shift issues on discrete-time iterative learning control with system relative degree," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 144–148, 2003.