2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

NMPC Strategy for Safe Robot Navigation in Unknown Environments using Polynomial Zonotopes

Iuro B. P. Nascimento[1], Brenner S. Rego[2], Luciano C. A. Pimenta[3], and Guilherme V. Raffo[3]

Abstract— **This work proposes a nonlinear model predictive control (NMPC) strategy for robot navigation in cluttered unknown environments using polynomial zonotopes. The information provided by a laser sensor is used in the computation of the collision-free area. The procedure splits the area into convex subregions which are converted into polynomial zonotopes (PZs) to generate constraints for the NMPC optimal control problem. The PZ is a set representation that can describe polytopes using fewer constraints than conventional half-space representations, thus being more efficient while maintaining the accuracy equivalent to the polytopic case. Numerical experiments demonstrate the advantages of the proposed strategy.**

I. Introduction

In recent decades, the interest in mobile robots has surged, and a wide range of applications has emerged, including assistive robots [1], search and rescue missions [2], and power line inspections [3]. For applications in known environments, a prevalent two-layer solution involves offline trajectory planning, followed by trajectory tracking using a control system [4]. However, in unknown environments, the two-layer approach becomes unsuitable due to computationally expensive planning algorithms [5], [6] and the need to frequently recalculate trajectories due to changing environments. An alternative strategy is Model Predictive Control (MPC), which integrates optimal trajectory planning and optimal control into a unified optimization problem. MPC calculates optimal control signals and trajectories by minimizing a performance index while considering constraints such as system dynamics and admisible states, control inputs, and obstacles.

Different approaches exist for incorporating collision-free areas (CFA) into the MPC optimal control problem (OCP). The complexity, constraints, and accuracy of representations are influenced by how the CFAs are described. A trade-off between accuracy and efficiency is typically present, as more precise descriptions increase complexity and constraints, impacting computational efficiency. Simpler descriptions might suffice for sparse environments but prove inadequate in cluttered environments where narrow passages can be obstructed, rendering optimization infeasible due to oversimplification.

Since the CFA is often nonconvex, representing the CFA with linear constraints requires partitioning into convex subregions. Union formulations in OCPs often adopt a combinatorial approach by introducing binary variables, subsequently addressed through mixed-integer (MI) solvers [7]. In both trajectory planning [8], [9] and MPCs [10]–[12], numerous works have adopted combinatorial representations for the CFA. In the work of [13], a safe corridor of potentially overlapping convex subregions has been proposed connecting to the goal position. Binary variables have been used to make a union of the subregions to select the next step of a bipedal robot. The constraints of the convex regions have been represented as half-spaces. Similarly to [13], the authors in [14] have utilized convex lifting to create a partition of space, generating a feasible path that orients the creation of convex partition forming a corridor to the goal. Although this approach finds a path toward the goal, it may not yield the best path as a significant portion of the CFA remains unexplored. Additionally, the hyperplane arrangements scheme obtains the regions through a combinatorial algorithm, which may result in an exponential number of regions relative to the original polytope sides. In [15], the obstacles have been approximated as rectangles and binary variables have been used in big-M formulation to make sure the trajectory is out of at least one barrier for each obstacle in the environment. The representation conservativeness can make passages obstructed. In [16], binary variables are added to convey different security areas near obstacles, where different constraints are used depending on how close the robot is to obstacles.

The primary contribution of this work is the development of a nonlinear model predictive control (NMPC) strategy for safe navigation in unknown environments using a mixed-integer formulation to describe the CFA. This strategy combines methods for processing the CFA and employs *polynomial zonotopes* (PZ) [17], a non-convex set representation, to describe the CFA. This approach aims to reduce the computational complexity of the method. We incorporate polynomial zonotopes as constraints directly, eliminating the need to revert to half-space representation as done with zonotopes in [14]. This results in a significant reduction in the number of constraints compared to half-space presentations such as in [7], [11], [13]. Consequently, this leads to a decrease in the computational complexity of the method while preserving the polytopic accuracy of representation. Our goal is to

*This work was in part supported by the project INCT (National Institute of Science and Technology) under the grant CNPq (Brazilian National Research Council) 465755/2014-3 and FAPESP (São Paulo Research Foundation) grant numbers 2014/50851-0 and 2022/05052-8. This work was also supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) (Finance Codes 001 and 88887.136349/2017-00), CNPq (grant numbers 315258/2020-9, 315695/2020-0, and 407063/2021-8), and FAPEMIG, under grant APQ-03090-17.

[1] Iuro B. P. Nascimento is with Graduate Program in Electrical engineering, Federal University of Minas Gerais, Brazil iuro@ufmg.br

[2]Brenner Santana Rego is with Department of Electrical and Computer Engineering, University of São Paulo, Brazil, brennersr7@usp.br

[3]Luciano Cunha de A. Pimenta and Guilherme Vianna Raffo are with Department of Electronic Engineering, Federal University of Minas Gerais, Brazil lucpim@ufmg.br and raffo@ufmg.br

maintain the entire safe search space to prevent unfeasibility, unlike corridor approaches [13], [14]. Our tests with 2D environments have demonstrated a reduction in the number of regions by using a dynamic programming algorithm to obtain the smallest number of partitions. This contrasts with hyperplane arrangement approaches such as [14], which uses a combinatorial algorithm that can potentially generate an exponential number of convex regions with respect to the obstacle sides.

II. Preliminaries

A *convex polytope* P with n_v vertices $v_i \in \mathbb{R}^n$ in vertex representation (V-rep) is described as $P_{\text{V}} \triangleq \{\sum_{i=1}^{n_v} \lambda_i v_i \mid \lambda_i \geq 0,\ \sum_{i=1}^{n_v} \lambda_i = 1\}$.

A convex polytope P in half-space representation (H-rep) is written as $P_{\text{H}} \triangleq \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$, where $\boldsymbol{A} \in \mathbb{R}^{n_h \times n}$ and $\boldsymbol{b} \in \mathbb{R}^{n_h}$. Using this H-rep as a constraint leads to n_h equations being employed in the OCP. A *constrained zonotope* (CZ) [18] is a representation (called CG-rep) of a convex polytope in the form $P_{\text{CZ}} \triangleq \{\boldsymbol{c} + \sum_{i=1}^{h} \alpha_i \boldsymbol{G}_{(\cdot,i)} \mid \sum_{i=1}^{h} \alpha_i \boldsymbol{A}_{(\cdot,i)} = \boldsymbol{b}, \alpha_i \in [-1, 1]\}$, where $\boldsymbol{c} \in \mathbb{R}^n$ is the center, n being the set dimension, and the matrix $\boldsymbol{G} \in \mathbb{R}^{n\times h}$ is the generator matrix, with each column corresponding to a generator. The matrix $\boldsymbol{A} \in \mathbb{R}^{s\times h}$ and the column vector $\boldsymbol{b} \in \mathbb{R}^s$ represent the constraints, where h and s are the numbers of generators and constraints, respectively. CZs have the advantage of being composed of linear equations, and they have explicit and efficient solutions to operations like linear maps, Minkowski sums, and intersections.

A sparse *polynomial zonotope* [19] is a compact efficient representation of a polynomial zonotope in the form

$$P_{\text{PZ}} \triangleq \left\{ \boldsymbol{c} + \sum_{i=1}^{h} \left(\prod_{k=1}^{p} \alpha_k^{\boldsymbol{E}(k,i)} \right) \boldsymbol{G}(\cdot, i) + \sum_{j=1}^{q} \beta_j \boldsymbol{G}_I(\cdot, j) \;\middle|\; \alpha_k, \beta_j \in [-1,\ 1] \right\}, \tag{1}$$

called polynomial generator representation (PG-rep). The matrix $\boldsymbol{G} \in \mathbb{R}^{n\times h}$ contains the h dependent generators, $\boldsymbol{G}(.,i)$ is the column generator i of $\boldsymbol{G}$, the matrix $\boldsymbol{G}_I \in \mathbb{R}^{n\times q}$ contains q independent column generators, $\boldsymbol{G}_I(.,i)$, and $\boldsymbol{E} \in \mathbb{N}^{p\times h}$ is the exponential matrix with p exponential factors. PZs have exact efficient solutions to linear maps, Minkowski sum, exact addition, quadratic map, and convex hull. A PZ is written in compact form as $P_{\text{PZ}} = \langle \boldsymbol{c}, \boldsymbol{G}, \boldsymbol{G}_I, \boldsymbol{E} \rangle_{\text{PZ}}$.

Polytopes in V-rep, H-rep, zonotopes, and others, can be converted into PZs exactly. A polytope in V-rep can be converted to PG-rep by first converting each vertex to PZ and later making a convex hull of all the resulting PZs using Algorithm 1 of [20]. A vertex $\boldsymbol{v}$ can be converted to PZ by making $PZ = \langle \boldsymbol{v}, [], [], [] \rangle_{\text{PZ}}$, where $[]$ is an empty matrix. The convex hull [19] of two PZs without independent generators, $PZ_1 = \langle \boldsymbol{c}_1, \boldsymbol{G}_1, [], \boldsymbol{E}_1 \rangle_{\text{PZ}}$ and $PZ_2 = \langle \boldsymbol{c}_2, \boldsymbol{G}_2, [], \boldsymbol{E}_2 \rangle_{\text{PZ}}$, is given by $\text{Hull}(PZ_1, PZ_2) = \langle 0.5(\boldsymbol{c}_1 + \boldsymbol{c}_2), 0.5\overline{\boldsymbol{G}}, [], \overline{\boldsymbol{E}} \rangle_{\text{PZ}}$, where $\overline{\boldsymbol{G}} \triangleq [(\boldsymbol{c}_1 + \boldsymbol{c}_2)\ \ \overline{\boldsymbol{G}}_1\ \ \overline{\boldsymbol{G}}_1\ \ \overline{\boldsymbol{G}}_2\ \ -\overline{\boldsymbol{G}}_2]$, with $\overline{\boldsymbol{G}}_1 \triangleq 0.5[\boldsymbol{G}_1\ \ \boldsymbol{G}_1\ \ \boldsymbol{G}_1\ \ -\boldsymbol{G}_1]$, $\overline{\boldsymbol{G}}_2 \triangleq 0.5[\boldsymbol{G}_2\ \ \boldsymbol{G}_2\ \ \boldsymbol{G}_2\ \ -\boldsymbol{G}_2]$, and

$$\overline{\boldsymbol{E}} \triangleq \begin{bmatrix} \boldsymbol{0} & \overline{\boldsymbol{E}}_1 & \overline{\boldsymbol{E}}_1 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \overline{\boldsymbol{E}}_2 & \overline{\boldsymbol{E}}_2 \\ 1 & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix},$$

with

$$\overline{\boldsymbol{E}}_1 \triangleq \begin{bmatrix} \boldsymbol{E}_1 & \boldsymbol{E}_1 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{E}_1 & \boldsymbol{E}_1 \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}, \quad \overline{\boldsymbol{E}}_2 \triangleq \begin{bmatrix} \boldsymbol{E}_2 & \boldsymbol{E}_2 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{E}_2 & \boldsymbol{E}_2 \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}.$$

Since PG-rep is a polynomial representation, PZs can represent nonconvex and non-polytopic sets. Besides, polytopes in H-rep generate the number of constraints proportional to the number of facets of the polytope. In contrast, the PG-rep has the number of constraints proportional to the set dimension and can scale better in cluttered environments than polytopic representations.

III. Problem Statement and Control Structure

Consider a nonlinear continuous-time system described by

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{2}$$

where $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state vector, $\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^m$ is the input vector, and $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a nonlinear function. The objective is to design an NMPC strategy to safely navigate system (2) through an unknown environment, in order to reach a target position as fast as possible.

The control structure proposed in this work is composed of two main tasks as it is shown in Fig. 1, which are: (i) *Constraint generation* (CG); and (ii) *NMPC*.

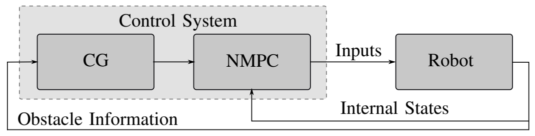

Fig. 1: Control Structure.

A. Constraint Generation Task

The CG task generates constraints describing the CFA, where the robot is allowed to navigate, using the raw data from the laser sensor (LIDAR, Light Detection And Ranging) and sensors to read the robot's position and orientation. The LIDAR scans its surroundings at regular angle intervals θ_{inc}, generating a point cloud of n_o distances o_i from obstacles measured at the angle θ_i, with $\theta_i = \theta_{i-1} + \theta_{\text{inc}}$, where each tuple $\boldsymbol{\eta}_i \triangleq (o_i, \theta_i)$ forms a point in polar coordinates. The n_o points $\boldsymbol{\eta}_i$ are converted to cartesian coordinates $\boldsymbol{\xi}_i \triangleq (x, y)$ to form a polytope that defines the CFA, also called *safe region* (SR), as its interior is the area where the LIDAR did not find any obstacles. Additionally, the LIDAR has a maximum detectable distance R_{sensor}, and the set of reachable points by the LIDAR is called the sensor field of vision (FOV), and the system's full state vector is assumed to be available through sensors.

Fig. 2 (a) shows an example of a 2D LIDAR output with two obstacles in gray and the possible openings that the system can go through to reach the target position $\boldsymbol{\xi}_{\text{goal}} = (x_{\text{goal}}, y_{\text{goal}})$. The vector $\boldsymbol{\xi}_0$ is the current position of the system and the baseline of the sensor is the red line. The

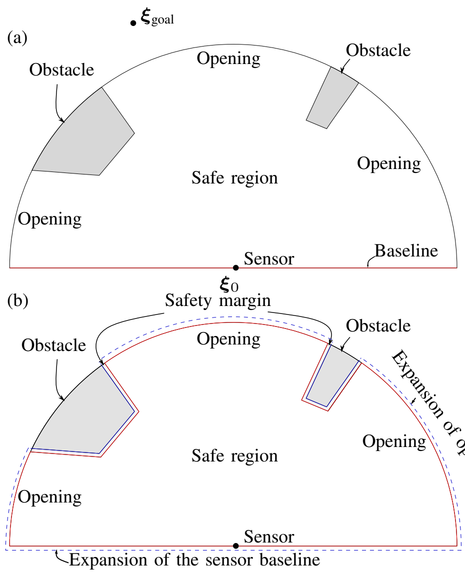

Fig. 2: Obstacle Polygon Expansion. (a) Original polygon with the baseline in red; (b) Expansion of the openings and the sensor baseline (dashed blue line) and its deflation into the red polygon.

CG is composed by three main subtasks:

1) Safe Region Creation: Creating the SR is the process of transforming the point cloud into a polytope. The 2D algorithm removes redundant points by using the line simplification algorithm [21]. The algorithm starts by receiving as input the list of points of the polytope and starts by building a line with the first and last points denoted as p_1 and p_2. Next, the furthest point from the line $\overline{p_1 p_2}$, denoted as p_t, is selected. If the distance between p_t and $\overline{p_1 p_2}$ is smaller than a tolerance value ϵ, the line can contain only its endpoints. Otherwise, p_t is maintained and two lines $\overline{p_1 p_t}$ and $\overline{p_t p_2}$ are created. The algorithm iteratively repeats these steps with each new line until all lines contain only two points.

2) Addition of a Safety Margin: The obstacle size is increased to account for the measurement and approximation errors, besides the robot size, allowing the OCP to consider the robot as a single point. We employ Vatti's clipping algorithm [22] to add a safety margin by inflating and deflating polytopes by a desired value d. The algorithm uses boolean operations to offset polytope border segments. It is noteworthy that only the obstacles have to be expanded, not the openings. In order to accomplish this, we first expand the baseline of the sensor and the openings of the polytope of Fig. 2 (a) to deflate later the whole polytope returning the baseline and openings to their original size. The result is an inward expansion of obstacles within the SR. The expanded polytope is shown in blue and the final deflated polytope is shown in red in Fig. 2 (b).

3) Safe Region Decomposition: The SR is divided into convex subregions. This technique enables the conversion of constraints into convex forms, leading to a simpler constraints. For 2D environments, the optimal convex partition

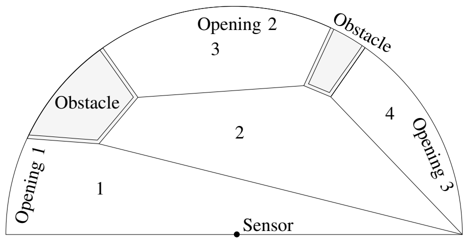

Fig. 3: Subregions and openings formed by processing the LIDAR output.

algorithm introduced in [23] is utilized to obtain the minimum number of convex partitions. The decomposition of the polytope of Fig. 2 can be observed in Fig. 3.

B. Nonlinear MPC Task

Given the constraints from the CG task, the NMPC executes the optimal control problem by using a nonlinear optimization solver to obtain an optimal trajectory and the corresponding optimal control signal, which is described as follows:

1) Obtaining the subregion constraints: Since the output of the decomposition consists of convex polytopes in V-rep, it is necessary to convert from V-rep to PG-rep by using the procedure from [19]. Therefore, we convert each vertex to a PG-rep and compute a convex hull of all PZs to form the subregion.

By using the PG-rep (1) as a constraint and adding the $\boldsymbol{\alpha}$ as decision variables, we need to find $\boldsymbol{\alpha}$ that makes the PZ equation equal to the desired test point $\boldsymbol{\xi} = \boldsymbol{C}\boldsymbol{x} \in \mathbb{R}^n$, ensuring it is inside the PZ. The set dimension is n, the matrix $\boldsymbol{C} = [\boldsymbol{I}_n,\ \boldsymbol{0}_{n \times N_x - n}]$, with $\boldsymbol{I}_n$ being an identity matrix of size n. The $\boldsymbol{\beta}$ variables in (1) are not used in the conversion from V-rep to PG-rep, therefore they are not used in this constraint, which is defined as

$$\mathcal{P}(\boldsymbol{x}, \boldsymbol{\alpha}) = \boldsymbol{c} - \boldsymbol{C}\boldsymbol{x} + \sum_{i=1}^{h} \prod_{k=1}^{p} \left(\alpha_k^{\boldsymbol{E}_{(k,i)}} \boldsymbol{G}_{(\cdot,i)} \right) = \boldsymbol{0}_{n \times 1}, \quad (3)$$
$$\alpha_k \in [-1, 1]$$

Eq. (3) results in n constraints per PZ per point $\boldsymbol{\xi}$, with $\boldsymbol{\xi} \in \mathbb{R}^n$ being the position of the robot. If there exist any $\boldsymbol{\alpha}$ that make (3) feasible, then the PZ equation can be equal to $\boldsymbol{\xi}$. Therefore, $\boldsymbol{\xi}$ is inside the PZ.

2) NMPC Strategy: The OCP is formulated aiming to obtain an optimal trajectory and optimal control signals with respect to the cost functional criteria. The constraints are imposed to maintain the trajectory inside the SR, which is generally nonconvex when obstacles are present and this may generate multiple local optima solutions. Given the subregions constraints of Eq. (3), the strategy consists of using binary variables in a mixed-integer formulation to build the union of all subregions, forming the nonconvex SR. Each state $\boldsymbol{x}(t_k)$ at time point t_k of the trajectory is constrained by all subregions S_l with its own vector $\boldsymbol{\alpha}_k = \left[\alpha_{[k,1]}, \ldots, \alpha_{[k,p_l]}\right]$ in the OCP as objective variables.

The goal is to find the best opening to reach the target point according to the criteria defined in the cost functional.

Since we may have multiple local minima, we formulate d OCPs with initial guess trajectory toward each of the d openings to reach solutions with different local minima. Since each OCP is independent of the others, all OCPs can be solved in parallel. The solution with the lowest cost is chosen.

IV. NMPC Formulation

The OCP formulation is given by

$$\min_{\substack{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\alpha},\\ \boldsymbol{\beta},\boldsymbol{\lambda},T_p}} \quad \mathfrak{T}\left(\boldsymbol{x}(T_p), \boldsymbol{x}_{\text{goal}}, T_p\right) + \int_0^{T_p} \mathfrak{I}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{x}_{\text{goal}}\right) dt \tag{4}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right), \tag{5}$$

$$\boldsymbol{x}_{\min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{\max}, \tag{6}$$

$$\boldsymbol{u}_{\min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{\max}, \tag{7}$$

$$\sum_{l=1}^{N} \mathcal{P}_l(\boldsymbol{x}(t), \boldsymbol{\alpha}_l)\lambda_l = \boldsymbol{0}, \tag{8}$$

$$\boldsymbol{\alpha}_l(j) \in [-1,\ 1], \forall\, j \in \mathbb{N}_{p_l}, l \in \mathbb{N}_N, \tag{9}$$

$$\sum_{l=1}^{N} \lambda_l = 1, \tag{10}$$

$$\lambda_l \in \{0,\ 1\},\ \forall l \in \mathbb{N}_N, \tag{11}$$

$$|\dot{\boldsymbol{u}}(t)| \leq \boldsymbol{a}_{\max}, \tag{12}$$

$$\mathfrak{F}\left(\boldsymbol{x}(T_p)\right) \leq \boldsymbol{0}, \tag{13}$$

with $\mathbb{N}_g \triangleq \{1,\ ,2,\ \ldots,\ g\}$ for any $g \in \mathbb{N}$, with N being the number of subregions, p_l is the number of $\boldsymbol{\alpha}$ variables in subregion l, and T_P is the time horizon. Eq. (4) is the cost functional, (5) is the system dynamics, Eqs. (6) and (7) are the physical limits of state and control signals. Eqs. (8) to (11) are the constraints associated to the union of the subregions S_l, where (8) is the PG-rep constraint (as (3)) for the points $\boldsymbol{x}(t)$. The external summation of (8) is over the subregions S_l, adding N PZ_l terms that constrain $\boldsymbol{x}(t)$ inside each S_l for each time $t \in [0, T_P]$. N binary variables λ_l are associated to each t. Eq. (10) states that only one λ_l is selected for a particular t, ensuring that only one PZ_l is selected to constrain $\boldsymbol{x}(t)$. For example, for two subregions, we have $PZ_1(\boldsymbol{x}(t_1))\lambda_1 + PZ_2(\boldsymbol{x}(t_1))\lambda_2 = \boldsymbol{0}$, $\lambda_1 + \lambda_2 = 1$. If a solution gives $\lambda_1 = 0$, and $\lambda_2 = 1$, $\boldsymbol{x}(t_1)$ is constrained inside PZ_2, while if $\lambda_1 = 1$, and $\lambda_2 = 0$, $\boldsymbol{x}(t_1)$ is constrained inside PZ_1. Additionally, in order to make the inputs physically realizable, we impose a maximum absolute value for the input derivatives of Eq. (12), where $\dot{\boldsymbol{u}}(t) \in \mathbb{R}^m$ are the input derivatives, the vector $\boldsymbol{a}_{\max} \in \mathbb{R}^m$ is the vector of maximum input derivatives.

The terminal constraint (13) depends on whether the target is inside the sensor FOV or not. If the target $\boldsymbol{x}_{goal}$ is outside of the sensor FOV, the terminal constraint is given by

$$R_{\text{sensor}} - \delta \leq \|\boldsymbol{\xi}^{(N)}(T_N) - \boldsymbol{\xi}^{(1)}(T_0)\| \leq R_{\text{sensor}}, \tag{14}$$

where R_{sensor} is the sensor range, δ is a tolerance value to allow the terminal region to be close to R_{sensor} , and $\boldsymbol{\xi}(t) = (x(t),\ y(t))$.

If the target is inside the sensor field of vision, the constraint is described as

$$\mathcal{B}(\boldsymbol{\xi}_{\text{goal}}, \sigma) \triangleq \{\boldsymbol{\xi} : \|\boldsymbol{\xi} - \boldsymbol{\xi}_{\text{goal}}\|_\infty \leq \sigma\}, \tag{15}$$

where $\mathcal{B}$ is a ball of infinity norm smaller than or equal to σ, centered at $\boldsymbol{\xi}_{\text{goal}}$. Additionally, the horizon $T_N = T_P$ is a decision variable, thus, the optimization can obtain a T_P large enough to allow the system the reach the terminal region.

A. *Cost Functional*

In the proposed method, the cost functional contains: (i) a weighted time horizon term to penalize the final horizon time T_P; and (ii) a terminal cost that penalizes the distance from the last trajectory point to the target point ($\boldsymbol{x}_{\text{goal}}$), yielding

$$\mathfrak{T}(\boldsymbol{x}(T_P), \boldsymbol{x}_{\text{goal}}, T_P) \triangleq\ w_t T_P + \|\boldsymbol{x}(T_P) - \boldsymbol{x}_{\text{goal}}\|^2_{\boldsymbol{P}}. \tag{16}$$

The stage cost penalizes the distance from each state point to the target, and the control signals energy, resulting in

$$\mathfrak{I}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{x}_{\text{goal}}) \triangleq \|\boldsymbol{x}(t) - \boldsymbol{x}_{\text{goal}}\|^2_{\boldsymbol{Q}} + \|\boldsymbol{u}(t)\|^2_{\boldsymbol{R}}, \tag{17}$$

where $\boldsymbol{P}$ is obtained by solving the Ricatti algebraic equation $\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}$, where we linearize the system around the current state $\boldsymbol{x}_0$ and control signal $\boldsymbol{u}_0$ to obtain the matrices $\boldsymbol{A} \triangleq \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}}$, and $\boldsymbol{B} \triangleq \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}}$.

V. Numerical Experiments

This section presents numerical results that demonstrate the performance of the proposed control strategy. The experiments were executed on a desktop computer with a 12 cores Intel i7 12700 4.8 GHz CPU, and 16 GB of RAM. The simulation is performed using Simulink and MATLAB R2023a, with the Robotics System Toolbox emulating the LIDAR sensor. The nonlinear dynamical equations of the system are simulated in Simulink and the controller is implemented using the CasADi toolbox [24] to formulate the OCP. The CGAL library is used to perform tasks such as the polytope partition [25], while the Clipper library [26] is used to add the safety margin to the obstacles.

Each OCP is transformed into a Mixed-Integer Nonlinear Programming (MINLP) problem using the hp-adaptive pseudospectral method, as described in [27]. These MINLP problems are subsequently solved using the Basic Open-source Nonlinear Mixed Integer Programming (Bonmin) [28] algorithm, which is built upon the Coin-or Branch and Cut open-source software [29]. Bonmin utilizes the interior-point filter line-search method implemented in IPOPT [30] (Interior Point OPTimizer).

A. *Wheeled Mobile Robot in a 2D Unknown Environment*

A wheeled mobile robot is simulated in a 2D environment to reach a target position. A differential-drive robot kinematic model, based on [31], is used in this study, considering a tracking point displaced by a distance of r in front of the geometric center of the robot. The equations of motion can be expressed as:

$$\begin{cases} \dot{x}(t) = \left(v_r(t) + v_l(t)\right)\left(\cos(\psi(t)) - d\sin(\psi(t))\right)/2, \\ \dot{y}(t) = \left(v_r(t) + v_l(t)\right)\left(\sin(\psi(t)) + d\cos(\psi(t))\right)/2, \\ \dot{\psi}(t) = \left(v_r(t) - v_l(t)\right)/W_{\text{dist}}, \end{cases} \tag{18}$$

where $v_r(t)$ and $v_l(t)$ are the right and left wheel's linear velocities, respectively, $\boldsymbol{x}(t) \triangleq (x(t), y(t), \psi(t))$ is the state vector of the system, where x and y are the linear translations of the robot from the inertial frame, ψ is the orientation of the robot, W_{dist} is the distance between the robot wheels, and $\boldsymbol{u}(t) \triangleq (v_r, v_l)$ is the input vector. The robot starts at $\boldsymbol{x}(0) = (0.2, 0.2, \pi/2)$, while the target state is $\boldsymbol{x}_{\text{goal}} = (20, 20, \pi/2)$. The obstacles are rectangles with randomly generated centers and shapes, on a map of side of 20×20 m. The obstacle facet lengths are randomly chosen from 1 to 1.6 m with uniform distribution.

B. Results and Discussion

The simulation considered the following disturbances: (i) the 2D LIDAR has 0.1 m as uncertainty amplitude of LIDAR measurement; and (ii) an additive white noise to measured state vector of maximum amplitude $\begin{bmatrix}0.1 & 0.1 & 1.5 \cdot 10^{-3}\end{bmatrix}^T$. The simulation parameters are: $W_{\text{dist}} = 1.0$ m, $\sigma = 0.1$ m, $\delta = 0.5$ m, $\epsilon = 0.1$, $R_{\text{sensor}} = 5$ m, $w_t = 0.1$, $\boldsymbol{Q} = \text{diag}(1/5^2, 1/5^2, 1/(40\pi^2))$, $\boldsymbol{R} = \text{diag}(1/(2)^2, 1/(2)^2)$, $\boldsymbol{P} = \text{diag}(1, 1, 1)$, $U_{\text{max}} = 1$ m/s, $U_{\text{min}} = -1$ m/s, and $\boldsymbol{a}_{\text{max}} = [3, 3]$ are input rate limit vector.

Figs. 4 to 6 illustrate the simulation outcomes using PZs for CFA representation. The robot navigates through all obstacles while respecting the safety margins. Fig. 5 displays the positions and velocities, and Fig. 6 depicts the control signals produced during simulation. The characteristic abrupt on/off behavior often observed in optimal solutions of OCPs with minimal time cost is mitigated by the input rate constraints of Eq. (12).

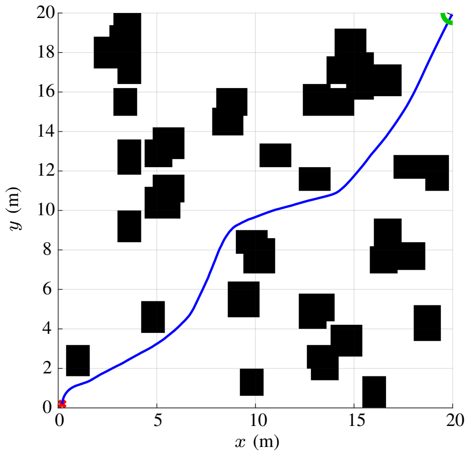

Fig. 4: Robot trajectory during simulation. The black areas represent the obstacles, the red × is the starting point, and the green circle is the target region around the target point. The blue line is the trajectory obtained in simulation.

Next, we verify the effect of using different representations of the CFA on the performance of the NMPC task, which involves the time solving the OCPs. We compare three representations of the CFA used in the constraints (8) to (10): (i) H-rep, (ii) PG-rep using Eq. (3), and (iii) CG-rep using a similar approach to Eq. (3), by using a CZ to restrain

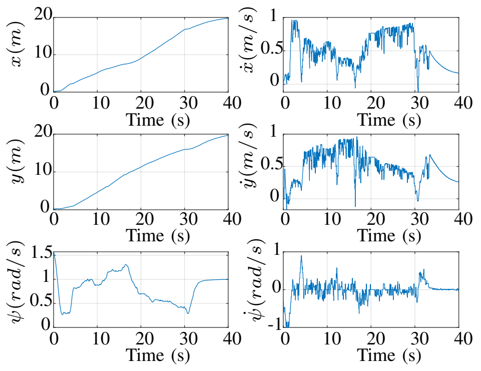

Fig. 5: Position and velocities during the experiment.

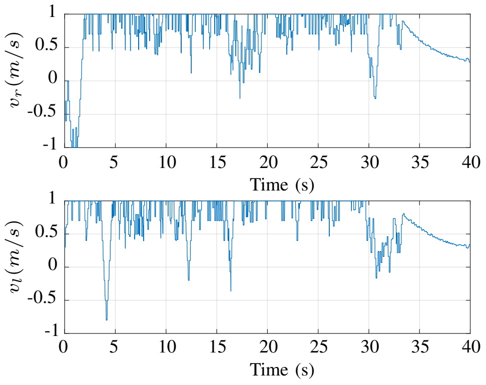

Fig. 6: Control signals applied to the robot during the experiment.

the point $\boldsymbol{\xi}$ obtaining the constraints

$$\boldsymbol{c} - \boldsymbol{C}\boldsymbol{x} + \sum_{i=1}^{p} \alpha_i \boldsymbol{G}_{(\cdot,i)} = \boldsymbol{0}_{d\times 1}, \tag{19}$$

$$\sum_{i=1}^{p} \alpha_i \boldsymbol{A}_{(\cdot,i)} = \boldsymbol{b}, \tag{20}$$

$$\alpha_i \in [-1,\ 1]. \tag{21}$$

While the Eq. (19) uses n constraints, the same as with PZs of Eq. (3), the Eq. (20) adds h_s constraints, where h_s is the number of half-spaces present in the H-rep (Theorem 1 of [18]).

We conduct CG and NMPC tasks for each test using the three representations on randomly generated maps with N_o obstacles. The NMPC task's execution time, which incurs higher computational costs, is measured. The tests are repeated 20 times per N_o, and the average time is computed. The speedup is defined as $S_{N_o} \triangleq T_{N_o,\text{H-rep}}/T_{N_o,\text{X-rep}}$, where X-rep $\in$ {H-rep, PG-rep, CG-rep}. $T_{N_o,\text{X-rep}}$ denotes the average optimization time across 20 maps with N_o obstacles using X-rep. $S_{N_o} > 1$ implies that X-rep requires, on average, less time for OCP optimization than H-rep, signifying a reduced computational cost.

The maps are generated with dimensions of 20×20 m containing N_o uniformly distributed obstacles with random sizes, as in the previous experiment. Six different N_o values (5 to 50) are used, as shown in Table I. It is observed that employing PG-rep and CG-rep as OCP constraints generally increases the NMPC task speedup compared to H-rep for all N_o, with PG-rep outperforming CG-rep. This results in reduced computational costs. As the obstacle count increases, both CG-rep and PG-rep exhibit increased speedups, highlighting the inferior scalability of H-rep in complex settings.

Rep. \ N_o	5	10	20	30	40	50
H-rep	1.00	1.00	1.00	1.00	1.00	1.00
PG-rep	7.13	9.18	17.63	16.18	17.92	12.23
CG-rep	3.52	4.98	10.82	10.23	13.90	10.72

TABLE I: Speedup S_N of optimization per safe region representation and per number of obstacles.

VI. Conclusion

This work developed an NMPC strategy which was able to navigate a mobile robot work in a cluttered 2D environment. Due to the minimum time cost, the control signals tend to be aggressive, but these were physically realizable thanks to the input rate constraints. Additionally, the optimization speedup using polynomial zonotopes indicates a clear reduction of computational burden. Besides, the optimization problem scaled better than other set representations when the environment was more cluttered.

References

[1] D. P. Losey, K. Srinivasan, A. Mandlekar, A. Garg, and D. Sadigh, "Controlling assistive robots with learned latent actions," in *IEEE ICRA*. IEEE, 2020, pp. 378–384.

[2] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019.

[3] M. Perez-Jimenez, MA. Montes-Grova, P. Ramon-Soria, BC. Arrue, and A. Ollero, "POSITRON: Lightweight active positioning compliant joints robotic arm in power lines inspection," in *IEEE ICUAS*, 2020, pp. 729–736.

[4] B. Wang, Y. Zhang, and W. Zhang, "Integrated path planning and trajectory tracking control for quadrotor UAVs with obstacle avoidance in the presence of environmental and systematic uncertainties: Theory and experiment," *Aerospace Science and Technology*, vol. 120, p. 107277, 2022.

[5] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *IJRR*, vol. 30, no. 7, pp. 846–894, 2011.

[6] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] D. Ioan, S. Olaru, S.-I. Niculescu, I. Prodan, and F. Stoican, "Navigation in a multi-obstacle environment. From partition of the space to a zonotopic-based MPC," in *IEEE ECC*, 2019, pp. 1772–1777.

[8] T. Wang, R. M. Lima, L. Giraldi, and O. M. Knio, "Trajectory planning for autonomous underwater vehicles in the presence of obstacles and a nonlinear flow field using mixed integer nonlinear programming," *Computers & Operations Research*, vol. 101, pp. 55–75, 2019.

[9] J. Tordesillas, B. T. Lopez, M. Everett, and J. P. How, "FASTER: Fast and Safe Trajectory Planner for Navigation in Unknown Environments," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 922–938, Apr. 2022.

[10] R. Deits and R. Tedrake, "Efficient mixed-integer planning for UAVs in cluttered environments," in *IEEE ICRA*. IEEE, 2015, pp. 42–49.

[11] F. Stoican, T.-G. Nicu, and I. Prodan, "A mixed-integer MPC with polyhedral potential field cost for obstacle avoidance," in *IEEE ACC*, 2022, pp. 2039–2044.

[12] A. Bürger, C. Zeile, A. Altmann-Dieses, S. Sager, and M. Diehl, "Design, implementation and simulation of an MPC algorithm for switched nonlinear systems under combinatorial constraints," *Journal of Process Control*, vol. 81, pp. 15–30, Sep. 2019.

[13] K. S. Narkhede, A. M. Kulkarni, D. A. Thanki, and I. Poulakakis, "A Sequential MPC Approach to Reactive Planning for Bipedal Robots Using Safe Corridors in Highly Cluttered Environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 831–11 838, Oct. 2022.

[14] D. Ioan, S. Olaru, I. Prodan, F. Stoican, and S.-I. Niculescu, "From obstacle-based space partitioning to corridors and path planning. A convex lifting approach," *IEEE Control Systems Letters*, vol. 4, no. 1, pp. 79–84, 2019.

[15] V. A. Battagello, N. Y. Soma, and R. J. M. Afonso, "Trajectory planning with a dynamic obstacle clustering strategy using Mixed-Integer Linear Programming*," in *2021 American Control Conference (ACC)*, 2021.

[16] L. Zong, J. Luo, M. Wang, and J. Yuan, "Obstacle avoidance handling and mixed integer predictive control for space robots," *Advances in Space Research*, vol. 61, no. 8, pp. 1997–2009, 2018.

[17] N. Kochdumper, "Extensions of polynomial zonotopes and their application to verification of cyber-physical systems," Ph.D. dissertation, Technische Universität München, 2022.

[18] J. K. Scott, D. M. Raimondo, G. R. Marseglia, and R. D. Braatz, "Constrained zonotopes: A new tool for set-based estimation and fault detection," *Automatica*, vol. 69, pp. 126–136, 2016.

[19] N. Kochdumper and M. Althoff, "Sparse polynomial zonotopes: A novel set representation for reachability analysis," *IEEE Transactions on Automatic Control*, vol. 66, no. 9, pp. 4043–4058, 2020.

[20] ——, "Representation of polytopes as polynomial zonotopes," *arXiv preprint arXiv:1910.07271*, 2019.

[21] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer graphics and image processing*, vol. 1, no. 3, pp. 244–256, 1972.

[22] B. R. Vatti, "A generic solution to polygon clipping," *Communications of the ACM*, vol. 35, no. 7, pp. 56–63, 1992.

[23] D. H. Greene, "The decomposition of polygons into convex parts," *Computational Geometry*, vol. 1, pp. 235–259, 1983.

[24] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *MPC*, vol. 11, no. 1, pp. 1–36, 2019.

[25] S. Hert, "2D polygon partitioning," in *CGAL User and Reference Manual*, 5th ed. CGAL Editorial Board, 2021.

[26] R. Wein, A. Baram, E. Flato, E. Fogel, M. Hemmer, and S. Morr, "CGAL: 2D minkowski sums," in *CGAL User and Reference Manual*, 5th ed. CGAL Editorial Board, 2021.

[27] C. L. Darby, W. W. Hager, and A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, vol. 32, no. 4, pp. 476–502, 2011.

[28] P. Bonami and J. Lee, "BONMIN user's manual," *Numerical mathematics (Hong Kong, China)*, vol. 4, pp. 1–32, 2007.

[29] J. Forrest and R. Lougee-Heimer, "Cbc (Coin-or branch and cut) user guide," in *Emerging Theory, Methods, and Applications*. INFORMS, 2005, pp. 257–277.

[30] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[31] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, and W. Burgard, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT press, 2005.