State-dependent dynamic tube MPC

A novel tube MPC method with a fuzzy model of model of disturbances

Surma, Filip; Jamshidnejad, Anahita

DOI
10.1002/rnc.7558

Publication date
2024

Document Version
Final published version

Published in
International Journal of Robust and Nonlinear Control

Received: 17 May 2023 | Revised: 8 July 2024 | Accepted: 13 July 2024

DOI: 10.1002/rnc.7558

RESEARCH ARTICLE

WILEY

State-dependent dynamic tube MPC: A novel tube MPC method with a fuzzy model of model of disturbances

Filip Surma | **Anahita Jamshidnejad**

Aerospace Engineering Faculty, TU Delft, Delft, South Holland, The Netherlands

Correspondence
Filip Surma, Aerospace Engineering Faculty, TU Delft, 2629 HS, Delft, South Holland, The Netherlands.
Email: f.surma@tudelft.nl

Funding information
Netherlands Organisation for Scientific Research (NWO), Grant/Award Number: 18120

Abstract

Most real-world systems are affected by external disturbances, which may be impossible or costly to measure. For instance, when autonomous robots move in dusty environments, the perception of their sensors is disturbed. Moreover, uneven terrains can cause ground robots to deviate from their planned trajectories. Thus, learning the external disturbances and incorporating this knowledge into the future predictions in decision-making can significantly contribute to improved performance. Our core idea is to learn the external disturbances that vary with the states of the system, and to incorporate this knowledge into a novel formulation for robust tube model predictive control (TMPC). Robust TMPC provides robustness to bounded disturbances considering the known (fixed) upper bound of the disturbances, but it does not consider the dynamics of the disturbances. This can lead to highly conservative solutions. We propose a new dynamic version of robust TMPC (with proven robust stability), called state-dependent dynamic TMPC (SDD-TMPC), which incorporates the dynamics of the disturbances into the decision-making of TMPC. In order to learn the dynamics of the disturbances as a function of the system states, a fuzzy model is proposed. We compare the performance of SDD-TMPC, MPC, and TMPC via simulations, in designed search-and-rescue scenarios. The results show that, while remaining robust to bounded external disturbances, SDD-TMPC generates less conservative solutions and remains feasible in more cases, compared to TMPC.

KEYWORDS

fuzzy-logic-based modeling, robust tube model predictive control, state-dependent disturbances

1 | INTRODUCTION

Model predictive control (MPC)[1,2] is a state-of-the-art control approach that can optimize multiple control objectives, handle state and input constraints, and provide guarantees on stability and feasibility. MPC heavily relies on a model that

Abbreviation: MPC, model predictive control.

Filip Surma and Anahita Jamshidnejad contributed equally to this study.

predicts the evolution of the states of the controlled system across a given prediction horizon. Thus, unmodeled external disturbances may lead to dangerous situations in real-life applications. For instance, search-and-rescue (SaR) robots should operate autonomously in unknown environments that are prone to external disturbances.[3,4] Therefore, for mission planning SaR robots need control methods that next to optimize the objectives of the SaR mission (e.g., maximizing the coverage of the area in the smallest possible time) and satisfying the constraints, also, deal with external disturbances that pose risks to the mission. Thus, MPC methods that can systematically handle the disturbances are promising for SaR robots.

Various extensions to standard MPC, including robust and stochastic MPC, have been developed.[5] In this article, we focus on robust tube MPC (TMPC),[6] which provides robustness to bounded disturbances, without significantly increasing the online computation time. TMPC generates control inputs that are composed of two parts: a nominal and an ancillary control input. The nominal states and control inputs of the system are determined by solving the nominal version of the MPC optimization (obtained by excluding external disturbances). The nominal states determine the centroids of a tube (i.e., a set of possible future states given chosen inputs and bounded disturbances) that propagates across the prediction horizon. The cross sections of the tube lie within the state space of the system, and as long as the realized states (i.e., the states in the presence of the external disturbances) at the corresponding time step remain in these cross sections, robustness is guaranteed. The ancillary controller ensures that the realized states remain inside the tube.

The nominal MPC in TMPC is not aware of the dynamics of the external disturbances. Thus, while the simplicity of the nominal optimization problem leads to a lower computation time, this may be at the cost of compromising the performance and generating overly conservative solutions, especially for systems with nonlinear dynamics and prone to dynamic disturbances. Moreover, existing MPC methods, including TMPC, do not naturally allow for incorporation of the expert knowledge. This gap should be addressed, particularly when extensive expert knowledge is available, but humans cannot or should not participate in real-time control procedures, for instance for autonomous onboard control of SaR robots.

1.1 | Main contributions

The main contributions of this article are:

1. We introduce state-dependent tube model predictive control (SDD-TMPC), which leverages classical tube model predictive control (TMPC) to incorporate a flexible tube that is adaptively optimized online.
   - Unlike TMPC which designs a static tube offline based on the maximum expected disturbances, SDD-TMPC allows both the size and the center of the tube to be determined by the optimization problem, taking into account the dynamics of the external disturbances as a function of the system's states. The resulting dynamic tube for SDD-TMPC is always a subset of the static tube of its counterpart TMPC.
   - Unlike TMPC where the ancillary control inputs are generated and included outside and independent of the optimization loop, the nominal SDD-TMPC optimization problem includes within its loop the dependence of the ancillary control inputs on the nominal states that are being determined. This alteration of the nominal SDD-TMPC optimization problem, compared to TMPC, results in completely different nominal trajectories, which based on theoretical discussions and numerical validations of the article, significantly reduce the conservativeness and thus risk of infeasibility of SDD-TMPC.
   - More specifically, while solving its nominal problem, SDD-TMPC ensures to steer the evolution of the states (i.e., the combination of the nominal states and errors due to the disturbances) according to online estimation of the state-dependent disturbances. Therefore, a trade-off is achieved between optimizing the estimated states and reducing the impact of the resulting disturbances. Accordingly, compared to TMPC, SDD-TMPC systematically reduces the risk of the infeasibility of the constrained optimization problem, as well as the conservatives (see Figure 1). This results in improved performance in terms of the realized cost, with affordable computations.
   - We prove the robust stability of SDD-TMPC under standard stability assumptions.
2. We introduce a set-based approach to model the dynamics of the state errors for SDD-TMPC based on the estimated dynamics of the (bounds of the) disturbances and the dynamics of the system itself. The mapping that evolves the error sets may be modeled as an analytical function or by data-based approaches (e.g., deep neural networks or fuzzy inference systems).
   - The mathematical conditions that such a mapping should generally satisfy are discussed.

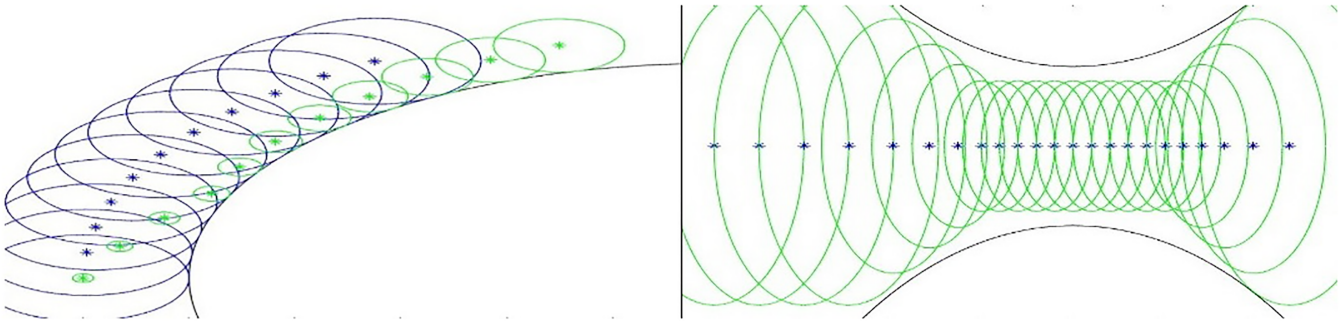

FIGURE 1 Comparison of SDD-TMPC and TMPC, where the sequences of the circular areas in both plots illustrate the evolution of the tube based on a prediction at the current time step. Left plot: The tube in SDD-TMPC (shown via the smaller varying green circles) is dynamic, and always a subset of the tube of regular TMPC (shown via the larger blue circles). Thus, SDD-TMPC allows the system to perform closer to the boundaries of hard constraints, and to potentially improve the performance, without violating these constraints. Right plot: The nominal trajectory (shown via the blue stars) and the tube of SDD-TMPC (shown via the green varying circles) are illustrated when the states are constrained to remain within a set shown by the black boundary curves. SDD-TMPC allows, for example, by reducing the speed of a robot, to formulate a feasible problem and provide solutions for it, and to safely move through the narrow corridor.

TABLE 1 Frequently used abbreviations and their explanation.

Abbreviation	Explanation	Abbreviation	Explanation
A*i*	The *i*th assumption	SaR	Search and Rescue
FIS	Fuzzy inference system	SDD-TMPC	State-dependent dynamic tube model predictive control
GA	Genetic algorithm	TSK-FIS	Takagi–Sugeno–Kang fuzzy inference system
MPC	Model predictive control	TMPC	Tube model predictive control
PS	Particle swarm algorithm	ZOH	Zero-order hold

- We specifically propose a mapping that learns the dynamics of the state-dependent disturbances, initialized by existing expert knowledge, and using a fuzzy inference system (FIS). Our main motivation for using a FIS is two-folded: First, FIS allows, without additional costs, to direct integration of the valuable knowledge of experts (e.g., experienced search-and-rescue staff) that is usually available as linguistic data, into the controller of a system (e.g., autonomous search-and-rescue robots). Second, while other modeling approaches, for example, neural networks require an extensive data set to be trained, a FIS can easily be initialized with human knowledge and then be fine-tuned online to adapt to the changing environment conditions.

3. We compare the performance of SDD-TMPC with regular MPC and with TMPC for both linear and nonlinear systems with state-dependent external disturbances. The simulated problems correspond to autonomous robots that navigate in unknown environments. The results confirm that SDD-TMPC outperforms both regular MPC and TMPC by generating less conservative control inputs that still guarantee the satisfaction of the hard constraints and by resulting in an overall smaller value for the objective function (considering a minimization problem).

Note that in various cases, by proper reformulation, model inaccuracies may also be represented as state-dependent disturbances, as will be illustrated in the second case study of this article. Thus, SDD-TMPC can be extended to deal with uncertainties due to model mismatches.

The rest of the article is structured as follows. In Section 2, we discuss the related work and identify the open challenges and limitations of the state-of-the-art methods. In Section 3, we explain our proposed approaches, including SDD-TMPC, and the fuzzy model of disturbances. We also provide proof for the stability of SDD-TMPC. In Section 4, we implement and compare the performance of MPC, TMPC, and SDD-TMPC via computer simulations for a ground robot. Finally, Section 6 concludes the article and suggests topics for future research. The abbreviations and mathematical notations that are frequently used in this article are listed in Tables 1 and 2, respectively.

TABLE 2 Frequently used mathematical notations and their definition.

Notation	Definition	Notation	Definition
k	Time step counter in the discrete-time domain	$\mathcal{K}$	A class of continuous and strictly increasing functions starting at zero
N	Prediction horizon	$\mathcal{K}_\infty$	A subclass of $\mathcal{K}$
$\boldsymbol{z}_k$	Nominal state vector	$\mathbb{W}(\boldsymbol{x}_k)$	Set of all possible disturbances given state $\boldsymbol{x}_k$ of the system at time step k
$\boldsymbol{x}_k$	State vector at time step k for a given dynamical system	$\overline{\mathbb{W}}_k$	Set of all possible disturbances for all possible states estimated at time step k
e_k	Error between the real and the nominal state for time step k	$\mathbb{X}$	Admissible set for the states
$\boldsymbol{w}(\boldsymbol{x}_k)$	Disturbance vector when the state of the system is $\boldsymbol{x}_k$	$\mathbb{Z}^{\mathrm{f}}$	Terminal set for the nominal states
$\boldsymbol{v}_k$	Nominal input vector	$\mathbb{U}$	Admissible set for the control inputs
$\boldsymbol{u}_k$	Input vector	$\mathbb{N}$	Set of natural numbers
$\tilde{\boldsymbol{z}}_k$	Sequence of the nominal state vectors across the prediction horizon (i.e., $\{\boldsymbol{z}_{k+1}, \dots, \boldsymbol{z}_{k+N}\}$) estimated at time step k	$\mathbb{R}$	Set of real numbers
$\tilde{\boldsymbol{x}}_k$	Sequence of the state vectors across the prediction horizon (i.e., $\{\boldsymbol{x}_{k+1}, \dots, \boldsymbol{x}_{k+N}\}$) estimated at time step k	$\overline{\mathbb{E}}_k$	Error set estimated for time step k, assuming an autonomous evolution of the error set (excluding the influence of the disturbances at previous time step $k-1$)
$\tilde{\boldsymbol{v}}_k$	Sequence of the nominal input vectors across the prediction horizon (i.e., $\{\boldsymbol{v}_k, \dots, \boldsymbol{v}_{k+N-1}\}$) estimated at time step k	$\mathbb{E}_k$	Set of all possible errors between the real and the nominal state for time step k
$\tilde{\boldsymbol{u}}_k$	Sequence of the input vectors across the prediction horizon (i.e., $\{\boldsymbol{u}_k, \dots, \boldsymbol{u}_{k+N-1}\}$) estimated at time step k	$\mathbb{T}_k^{\max}$	Tube of regular robust TMPC for prediction horizon $\{k+1, \dots, k+N\}$ with a fixed cross section area $\mathbb{E}^{\max}$
$\pi(\cdot)$	Control policy of SDD-TMPC	$\mathbb{T}_k$	Tube of SDD-TMPC for prediction horizon $\{k+1, \dots, k+N\}$ with a dynamic cross section area $\mathbb{E}_i$ for $i = k+1, \dots, k+N$
$\epsilon(\cdot)$	Autonomous error dynamic function, which determines $\overline{\mathbb{E}}_k$ based on $\mathbb{E}_{k-1}$	$\oplus$	Minkowski summation
$(\cdot)_{i\|k}$	prediction of a dynamic variable for time step i, given the measurements at time step k	$\ominus$	Minkowski difference

2 | RELATED WORK

2.1 | Modeling via fuzzy inference systems

FIS can approximate any nonlinear function, when the inputs of the function are bounded.[7] Various supervised learning algorithms, for example, gradient descent,[8] learning from example,[9] and genetic algorithm (GA)[10] have been proposed for the FIS to learn such functions from data. One of the most common FISs used in various applications, including robotics, is Takagi-Sugeno-Kang (TSK) FIS. TSK FISs can be updated online in a stable way using, for example, reinforcement learning (see applications to robot navigation,[11] balancing a bipedal robot,[12] controlling a continuum robot[13]). To the best of our knowledge, FISs have never been used to model the dynamics of external disturbances, especially for providing robustness for model-based control approaches, for example, MPC.

2.2 | Robust MPC

Linear MPC is a well-known, theoretically mature control approach that has successfully been implemented in many applications.[14,15] Linear MPC was utilized to regulate multiple vehicles across various domains such as spacecraft altitude control,[16] ground vehicle steering,[17] and drone trajectory tracking.[18] The LMPC framework incorporates a linear system, linear constraints, and a quadratic cost function. In real-life problems, including SaR, in order to deal with nonlinear cost functions and nonlinear constraints, a nonlinear version of MPC may be used. Nonlinear model predictive control (NMPC) has been implemented in various applications including exploration of a grid world with multiple agents,[19] decision making for simultaneous localization and mapping (SLAM),[20] obstacle avoidance,[21] and global path planning.[22] Although MPC is robust to small disturbances (due to operating upon state feedback), linear and nonlinear MPC cannot deal with large external disturbances.[5] Therefore, robust MPC approaches have been introduced.

One of the most attractive robust MPC methods is robust tube MPC (TMPC),[6] where a robust control strategy that is designed offline is used to keep the state trajectory within an invariant tube, the center-line of which is a nominal trajectory of the states that is determined online. For the sake of brevity of the notations, we refer to robust TMPC as TMPC in the rest of the article. The nominal state trajectory can be determined for the system whenever the controller has a perfect model of the system and there are no disturbances. Although the nominal states are predictable in such cases, the actual states cannot be known in advance because the external disturbances are unknown. However, since the disturbances are assumed to be bounded and thus the maximum error between the actual and the nominal states can be determined, it is possible to compute a sequence $\mathbb{T}_k^{\max}$ of N sets per time step k, where each set $\{z_{i|k}\} \oplus \mathbb{E}^{\max}$ within this sequence includes all the possible actual states for a given time step i within the prediction window $\{k+1, \dots, k+N\}$, and N is the prediction horizon of TMPC. Then the state constraints are guaranteed to be satisfied within the prediction window, if all the states within sequence $\mathbb{T}_k^{\max}$ satisfy the constraints. In order to prevent the actual states from deviating significantly from the corresponding nominal states, an ancillary control law, for example, an error state feedback controller,[6] another MPC controller,[23] or a sliding mode control,[21] is used to reduce the error between the actual and the nominal state.

There are two common ways of designing and implementing TMPC[24]:

- The first approach involves determining a single error set (i.e., tube) for the entire prediction window based on the error dynamics and the selected ancillary control law, where this error set serves as a positive robust invariant set. In other words, if the current error belongs to this positive robust invariant set, then all future errors will remain within the set. This approach benefits from a low online computational complexity since the positive robust invariant set can be computed offline. Moreover, tighter constraints can be imposed on the nominal set, such that if the center of the positive robust invariant set is inside the tightened constraints, then the actual state remains in the original feasible set. However, using this approach may result in more restrictive control inputs.
- The second approach involves computing the error set per time step within the prediction window. This approach results in a sequence of regions, known as reachable sets, which are the smallest sets of possible states for a system prone to external disturbances that guarantee that the states remain in these sets for all time steps. This approach may yield less conservative solutions for TMPC but requires additional online computations compared to the first approach.

Some variations of TMPC for nonlinear systems have been proposed. Nonlinear TMPC is much more challenging than linear TMPC, particularly because it is not trivial to choose an ancillary control law and design a tube. One way to reduce the computational complexity of nonlinear TMPC is by using parameterized TMPC, such as the approach in Reference 25, which reduces the required online optimization into a linear programming problem. Another alternative is to employ ellipsoids as tubes, instead of polytopic sets (see, e.g., Reference 26). Some examples of nonlinear TMPC can be found in References 21,23,27–29.

The first implementation of robust MPC (although not TMPC) when state-dependent disturbances exist includes.[30] In Reference 31 the optimization problem of robust MPC has been handled as a 2nd-order cone program. This approach, however, can only be used for linear systems subject to a certain group of additive disturbances (i.e., disturbances defined as the summation of an independent component from a polytope and a state and input-dependent component

bounded via a nonconvex inequality). Similarly, in Reference 32 robust MPC has been discussed for a special class of state-dependent disturbances. However, our proposed TMPC-based approach, that is, SDD-TMPC, is not limited to any type of nonlinearity. Authors in Reference 21 propose a stable controller for an agent that should avoid obstacles in an environment with disturbances that are proportional to the square of its velocity. The proposed method, however, is restricted to a sliding-mode ancillary control law, and thus can only be used for feedback-linearizable or minimum-phase systems. In Reference 33, a dynamic tube is used and is parameterized as a sublevel set of incremental Lyapunov functions. This method is extended in Reference 34, where chance constraints and state or input-dependent disturbances are included. This article has the most similarities with SDD-TMPC, since it uses a dynamic tube that evolves in time, and tightens the constraints online. Using a sublevel set of incremental Lyapunov functions reduces the conservativeness with a small increase in parameters and equations (and probably in computations) compared to other approaches in the literature. The parameterization, however, restricts the shape of the tube, which potentially leads to conservative decisions. In Reference 27, a neural network has been used to learn the dynamics of the tube of robust TMPC. The neural network, however, cannot be initialized without an extensive dataset, and thus another controller should be used until a sufficient number of data is gathered. Moreover, no proof has been provided that the trajectory of the states remains inside the tube. In Reference 28, a Gaussian process is used to learn the disturbance set. The stability of the algorithm for linear systems has been proven. However, this method requires heavy offline computations, and discretization of the state space, and may thus become intractable for systems with large state spaces.

SDD-TMPC is proven to be stable under standard assumptions. SDD-TMPC can be used in unknown and time-varying environments, being initially provided with intuitive human knowledge about similar environments, using fuzzy rules, and afterward being updated online or based on newly collected real-life data from the environment. Unlike most state-of-the-art methods, SDD-TMPC is not limited to any specific class of nonlinearity and does not restrict the choice of the ancillary control law. Moreover, unlike state-of-the-art methods that assume the bounds or the probability distribution of the disturbances are known, for SDD-TMPC it is possible to learn the disturbances online.

3 | SDD-TMPC: IDEA, FORMULATION, AND STABILITY

In this section, we describe and formulate the problem and discuss the proposed methods for SDD-TMPC.

3.1 | Problem statement

We consider a dynamic system that is described in discrete time with the state vector $\boldsymbol{x}_k$ and control input $\boldsymbol{u}_k$, and is affected by additive state-dependent external disturbances $\boldsymbol{w}(\boldsymbol{x}_k)$ at time step $k \in \mathbb{N}$, where $\boldsymbol{x}_k \in \mathbb{X} \subseteq \mathbb{R}^n$, $\boldsymbol{u}_k \in \mathbb{U} \subseteq \mathbb{R}^m$, and $\boldsymbol{w}(\boldsymbol{x}_k) \in \mathbb{W}(\boldsymbol{x}_k) \subseteq \mathbb{R}^n$. While the admissible sets $\mathbb{X}$ and $\mathbb{U}$ for, respectively, the state and the control input are static, we allow the admissible set $\mathbb{W}(\boldsymbol{x}_k)$ of the external disturbances to vary as a function of the state. Learning (an approximate of) the admissible set $\mathbb{W}(\boldsymbol{x}_k)$, such that it bounds the external disturbances per state, instead of considering a set that bounds all potential external disturbances for the entire admissible set $\mathbb{X}$ in time, will introduce dynamics and reduced conservativeness into SDD-TMPC.

The dynamic system is given by:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}(\boldsymbol{x}_k), \tag{1}$$

where $f(\cdot)$ is a time-invariant time-discrete nonlinear Lipschitz function.

We assume that the states are measured by perfect sensors, that is, per time step k the value of the state $\boldsymbol{x}_k$ is perfectly known, whereas the external disturbances are unknown and may take any arbitrary value within $\mathbb{W}(\boldsymbol{x}_k)$. The aim is to control the dynamics of system (1), such that a given cost function $J(\cdot)$ is minimized across a prediction horizon of size N, while it is guaranteed for the controlled system that for all admissible external disturbances the hard constraints are always satisfied. Thus, for time step k we have:

$$
\begin{aligned}
J^*(\boldsymbol{x}_k) = \min_{\tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{u}}_k} \; & J(\tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{u}}_k) \\
\text{s.t. for } i = k, \ldots, k+N-1 & \\
& \boldsymbol{x}_{i+1|k} = f(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) + \boldsymbol{w}(\boldsymbol{x}_{i|k}) \\
& \boldsymbol{x}_{k|k} = \boldsymbol{x}_k \\
& \boldsymbol{x}_{i+1|k} \in \mathbb{X} \\
& \boldsymbol{u}_{i|k} \in \mathbb{U} \\
& \boldsymbol{w}(\boldsymbol{x}_{i|k}) \in \mathbb{W}(\boldsymbol{x}_{i|k})
\end{aligned} \tag{2}
$$

with $\tilde{\boldsymbol{x}}_k = [\boldsymbol{x}_{k+1|k}^\top, \ldots, \boldsymbol{x}_{k+N|k}^\top]^\top$, $\boldsymbol{x}_{i|k}$ for $i > k$ the value of $\boldsymbol{x}_i$ predicted at time step k, $\tilde{\boldsymbol{u}}_k = [\boldsymbol{u}_{k|k}^\top, \ldots, \boldsymbol{u}_{k+N-1|k}^\top]^\top$, $\boldsymbol{u}_{i|k}$ for $i \geq k$ the value of $\boldsymbol{u}_i$ determined at time step k, and $J^*(\boldsymbol{x}_k)$ an optimal value for the cost function obtained by solving the constrained minimization problem. The cost function $J(\cdot)$ is composed of a stage cost that is accumulated across the prediction horizon $\{k, \ldots, k+N-1\}$, and a terminal cost that is computed at the terminal time step $k+N$. The cost function is continuous and finite for $\boldsymbol{x}_i \in \mathbb{X}, \; \forall i \in \{k, \ldots, k+N\}$.

3.2 | State-dependent dynamic tube MPC (SDD-TMPC)

TMPC works based on the assumption of bounded external disturbances, where the boundary set $\mathbb{W}^{\max}$ for the disturbances is known and given.[6] A main challenge for SDD-TMPC in solving (2) per time step k is that the dynamic set $\mathbb{W}(\boldsymbol{x}_i)$ (for $i = k, \ldots, k+N-1$) is not known (neither for the current time step nor for the future time steps) and should thus be estimated. The link between the known static set $\mathbb{W}^{\max}$ and the dynamic set $\mathbb{W}(\boldsymbol{x}_i)$ of disturbances that should be estimated by SDD-TMPC for every prediction horizon $\{k+1, \ldots, k+N\}$ is given by:

$$
\bigcup_{i=k, \ldots, k+N-1} \mathbb{W}(\boldsymbol{x}_i) \subseteq \mathbb{W}^{\max}. \tag{3}
$$

We assume that $\mathbb{W}(\boldsymbol{x}_i)$ always contains the origin. Regular TMPC uses $\mathbb{W}^{\max}$ to determine a sequence of N control inputs and a tube $\mathbb{T}_k^{\max}$ for the entire prediction window per time step k, for which the cross section $\mathbb{E}^{\max}$ remains unchanged. The tube $\mathbb{T}_k^{\max}$ is a robust positive invariant set. The main difference between TMPC and our proposed approach, SDD-TMPC, is in the formulation of their nominal MPC. This allows SDD-TMPC to make use of the dynamics of the external disturbances, in order to generate per time step k a tube $\mathbb{T}_k$ with a generally time-varying cross section $\mathbb{E}_i$ for $i = k+1, \ldots, k+N$ across the prediction horizon that improves the control performance by generating less conservative solutions. Note that set $\mathbb{T}_k$ is a robust control invariant set, that is, there is at least one control policy that prevents the state trajectory from leaving the tube. The nominal problem of SDD-TMPC is given by:

$$
V^*(\boldsymbol{x}_k, \boldsymbol{z}_k) = \min_{\tilde{\boldsymbol{z}}_k, \tilde{\boldsymbol{v}}_k, \mathbb{T}_k} \sum_{i=k}^{k+N-1} l(\boldsymbol{z}_{i|k}, \boldsymbol{v}_{i|k}) + V_{\mathrm{f}}(\boldsymbol{z}_{k+N|k}), \tag{4a}
$$

$$
\text{s.t. for } i = k+1, \ldots, k+N:
$$

$$
\boldsymbol{z}_{i|k} = f(\boldsymbol{z}_{i-1|k}, \boldsymbol{v}_{i-1|k}), \tag{4b}
$$

$$
\overline{\mathbb{E}}_{i|k} = \epsilon(\mathbb{E}_{i-1|k}), \tag{4c}
$$

$$
\boldsymbol{\Theta}_{i-1|k} = f^{\mathrm{dis}}\big(\{\boldsymbol{z}_{i-1|k}\} \oplus \mathbb{E}_{i-1|k}\big), \tag{4d}
$$

$$
\mathbb{E}_{i|k} = \overline{\mathbb{E}}_{i|k} \oplus \overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\Theta}_{i-1|k}), \tag{4e}
$$

$$
\boldsymbol{x}_k, \boldsymbol{z}_k \text{ are given} \tag{4f}
$$

$$
\mathbb{E}_k = \{\boldsymbol{x}_k - \boldsymbol{z}_k\}, \tag{4g}
$$

$$\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k} \subseteq \mathbb{X}, \tag{4h}$$

$$\boldsymbol{z}_{N|k} \in \mathbb{Z}^{\mathrm{f}}, \tag{4i}$$

$$\mathbb{T}_k = \left\{\{z_{k+1|k}\} \oplus \mathbb{E}_{k+1|k}, \ \dots \ , \{z_{N|k}\} \oplus \mathbb{E}_{k+N|k}\right\}, \tag{4j}$$

$$\pi\left(\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k}, \boldsymbol{v}_{i|k}, \boldsymbol{z}_{i|k}\right) \subseteq \mathbb{U}. \tag{4k}$$

In (4a), $l(\cdot)$ is the stage cost, $V_{\mathrm{f}}(\cdot)$ is the terminal cost, and $\tilde{\boldsymbol{z}}_k$ and $\tilde{\boldsymbol{v}}_k$ are the trajectory/sequence of, respectively, the nominal states and the nominal control inputs within the prediction window $\{k+1, \ \dots \ , k+N\}$. The realized states of the system follow the nominal state sequence when there are no external disturbances. The nominal states $\boldsymbol{z}_{i|k}$ are predicted at time step k for time step $i = k+1, \ \dots \ , k+N$ according to (4b). Moreover, $\oplus$ represents the Minkowski addition,[35] which for every two given sets A and B is defined by:

$$\mathbb{A} \oplus \mathbb{B} = \{a + b \ : \ a \in \mathbb{A}, b \in \mathbb{B}\}. \tag{5}$$

The error set $\mathbb{E}_{i|k}$ (for $i = k+1, \ \dots \ , k+N$) for SDD-TMPC is computed in three steps: First, in (4c) the system uses the autonomous error dynamic function $\epsilon(\cdot)$ to estimate the evolved error set, without considering the influence of the state-dependent disturbances on the errors. The autonomous error dynamic function is determined via state-of-the-art methods and based on the nominal model $f(\cdot)$ of the system dynamics and the policy $\pi(\cdot)$, assuming no external disturbances exist. Note that the error vector is defined in the prediction window by $\boldsymbol{e}_{i|k} = \boldsymbol{x}_{i|k} - \boldsymbol{z}_{i|k}$ for $i = k+1, \ \dots \ , k+N$. For details on how $\epsilon(\cdot)$ can be determined, see, for example, Reference 6, for linear TMPC and Reference 21 for nonlinear TMPC. Second, in (4d) a mapping, f^{dis}, is used to determine, based on the estimated nominal states (see (4b)) and the error set corresponding to the previous time step, a vector of parameters $\boldsymbol{\Theta}_{i-1|k}$. This vector is then used in (4e) to determine an admissible set $\overline{\mathbb{W}}_{i-1|k}$ per time step i ($i = k+1, \ \dots \ , k+N$) that contains all the possible disturbances at time step $i-1$ that can result in all the possible states at time step i. Note that mathematically, f^{dis} can in general be represented by any mappings (e.g., an analytical function or a deep neural network) that properly formulate the existing knowledge about the state-dependent disturbances, as long as the mapping meets the conditions explained in Section 3.3.1. The disturbance model (i.e., f^{dis} in (4c)) is in general developed offline, based on the existing general knowledge that corresponds the expected disturbance levels with the states of the environment, thus the states of the robot. Such prior information may be available as quantified data or in linguistic terms (e.g., via expert search-and-rescue staff). Accordingly, the most suitable ways for modeling (e.g., machine learning, fuzzy-logic-based modeling) will be used. Third, the sets obtained via the previous two steps (i.e., via (4c) and (4d)) are combined in (4e). In other words, the set $\mathbb{E}_{i|k}$ of all possible errors for time step i is obtained by combining the autonomously evolved error set and the set of all possible disturbances from the previous time step that can result in further errors. Based on Definition 1 given next, $\mathbb{E}_{i|k}$ for $i = k+1, \ \dots \ , k+N$ is an estimation of a robust forward reachable set for $\mathbb{E}_{i-1|k}$.

Definition 1. Consider the autonomous system that is formulated via (4c)–(4e). A one-step robust reachable set from the set of errors $\mathbb{E}_{i-1|k}$ that is estimated for time step $i-1$ (with $i = k+1, \ \dots \ , k+N$) is defined based on Reference 36, via:

$$\mathcal{R}_{\epsilon}(\mathbb{E}_{i-1|k}, \overline{\mathbb{W}}_{i-1|k}) = \left\{\boldsymbol{e} \in \left(\mathbb{X} \ominus \{\boldsymbol{z}_{i|k}\}\right) | \exists \boldsymbol{e}_{i-1|k} \in \mathbb{E}_{i-1|k}, \exists \boldsymbol{w}(\boldsymbol{x}_{i-1|k}) \in \overline{\mathbb{W}}_{i-1|k} \ : \ \boldsymbol{e} = \epsilon(\boldsymbol{e}_{i-1|k}) + \boldsymbol{w}(\boldsymbol{x}_{i-1|k})\right\}.$$

The three steps explained above for estimation of the error set $\mathbb{E}_{i|k}$ (for $i = k+1, \ \dots \ , k+N$) are illustrated in Figure 2. Constraint (4g) initializes the error set per time step when the nominal SDD-TMPC is solved. Constraint (4h) assures that the realized states, estimated online based on the nominal state and the error values, remain inside the admissible state set $\mathbb{X}$. Finally, (4i) is a terminal constraint with $\mathbb{Z}^{\mathrm{f}} \subseteq \mathbb{X}$, where $\mathbb{Z}^{\mathrm{f}}$ is a control invariant set for the nominal system that guarantees the recursive feasibility.

The dynamic tube $\mathbb{T}_k$ of SDD-TMPC for time step k is given by (4j). The optimization variables of (4) are the nominal state sequence $\tilde{\boldsymbol{z}}_k$, the nominal control input sequence $\tilde{\boldsymbol{v}}_k$, and the SDD-TMPC tube $\mathbb{T}_k$ (i.e., an ordered set including the influence of the error sets across the entire prediction horizon). By including the tube as an optimization variable for the nominal SDD-TMPC problem, the optimizer determines solutions that foresee the dynamics of the error in the

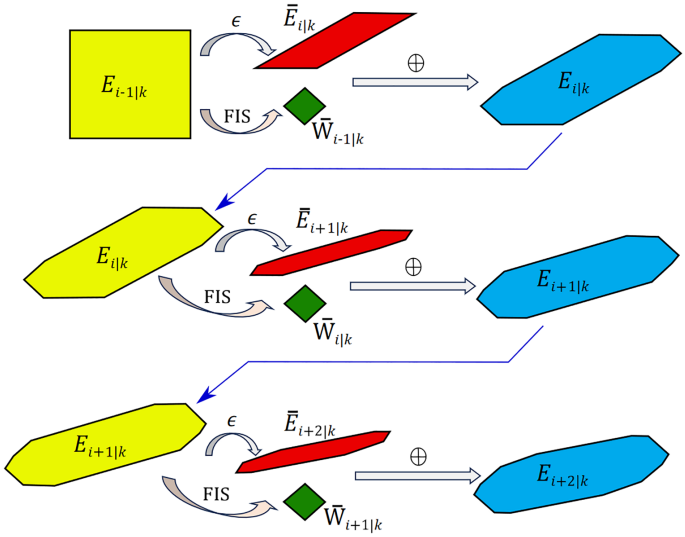

FIGURE 2 Illustration of the development of the error set $\mathbb{E}_{\cdot|k}$ for SDD-TMPC via (4c)–(4e) in 2 dimensions: Each row corresponds to one prediction time step (illustrated in this figure for time steps i, $i+1$, $i+2$). The 2-dimensional sets in the first column (shown in yellow) illustrate the error set $\mathbb{E}_{\cdot-1|k}$ that has been determined at the previous time step. By propagating this error set through the autonomous error dynamics (i.e., mapping ϵ) via (4c), the 2-dimensional sets $\overline{\mathbb{E}}_{\cdot|k}$ at the top side of the second column of each row (shown in red) are generated. Moreover, the 2-dimensional sets $\overline{\mathbb{W}}_{\cdot-1|k}$ at the bottom part of the second column in each row (shown in green) are obtained via the disturbance model (mapping $f^{\text{dis}}(\cdot)$) using (4d) where the input of the mapping is the error set that is centered around the nominal state trajectory. Finally, the red and green 2-dimensional sets of the second column per row are combined using Minkowski summation to obtain the error set $\mathbb{E}_{\cdot|k}$ via (4e). This set is then used as the starting set for the next time step (i.e., at the start of the next row).

generation of the online nominal state trajectory (i.e., the center-line of tube $\mathbb{T}_k$). Therefore, the nominal state trajectory of SDD-TMPC is in general different from that of regular TMPC (see Figure 1). Moreover, the SDD-TMPC policy $\pi : \mathbb{X} \times \mathbb{U} \times \mathbb{X} \to \mathbb{U}$, which receives the nominal state trajectory of SDD-TMPC as input, is incorporated within the optimization loop of SDD-TMPC via constraint (4k), as well as in the estimation of the autonomous error dynamic function as explained before. The policy $\pi(\cdot)$ of SDD-TMPC, which combines the nominal SDD-TMPC inputs and ancillary control inputs, may be generated such that the closed-loop system is stabilized (see, e.g., References 21 and 23). Constraint (4k) enforces the generated control inputs to remain inside the admissible control input set $\mathbb{U}$ for all possible realizations of the state. Thus, the policy also plays a role in the computation of the online nominal state trajectory that is determined via SDD-TMPC. Note that for computing the policy in (4k), where the input of the function is a set, Definition 2 given next is used.

Definition 2. Whenever a function $g(\cdot)$ takes a set $\mathbb{S}$ as input, then the output $\mathbb{O}$ is also a set defined by $\mathbb{O} := \{g(i) \big| i \in \mathbb{S}\}$, when $g(i)$ itself is not a set, and by $\bigcup_{i\in\mathbb{S}} g(i)$ when $g(i)$ itself is a set. Thus, if $\mathbb{S}_1 \subseteq \mathbb{S}_2$, then $g(\mathbb{S}_1) \subseteq g(\mathbb{S}_2)$.

State-of-the-art TMPC methods, however, solve the nominal MPC problem without incorporating the policy $\pi(\cdot)$. In fact, only after determining the nominal state trajectory via an optimization procedure, the policy of TMPC is used outside of the optimization loop to generate the actual control input $\boldsymbol{u}_k$.

In summary, the inclusion of the error evolution model into the optimization procedure of SDD-TMPC, which is done via incorporating the dynamic tube of SDD-TMPC in the optimization variables, as well as including the policy of SDD-TMPC into the constraints, are expected to result in solutions for SDD-TMPC that are less conservative than the solutions of TMPC.

The optimization problem (4) is in general nonlinear and nonconvex. It is common to assume convexity for the admissible sets $\mathbb{X}$ and $\mathbb{U}$ for the states and the control inputs,[6,28] but this is not the case in some applications (e.g., in collision avoidance,[21,36] which is relevant for SaR). Therefore, to solve the nonconvex, nonlinear optimization problem, solvers such as genetic algorithm,[10] particle swarm,[37] sequential quadratic programming[38] may be used. In particular, to address the issues that are raised due to nonconvexity, especially falling into local optima (due to the gradient-based nature of the optimizer or the limited number of iterations for the optimization) one may use multi-start optimization.

3.3 | Modeling the dynamics of the external disturbances

We assume that the predicted and actual states will be different, and that the difference is bounded, but the bound is state-dependent. This difference can be the result of a real external force that affects the system differently depending on its state. For example, when a robot moves in an environment, the external force may vary in different parts of the environment. This difference can also be the result of an inaccurate model, and the inaccuracy can be state-dependent. For example, it is well known that when a system is linearized around a certain state, a model is very accurate around that state, but becomes less accurate as it moves away. Both of these examples will be shown in the case studies.

The goal is to use a disturbance model (f^{dis}) to model the set of external disturbances per time step $i-1$, where i belongs to the prediction horizon $\{k+1, \dots, k+N\}$, as a function of the system states, that is, to approximate $\mathbb{W}(\boldsymbol{x}_{i-1|k})$ given in (3) by $\overline{\mathbb{W}}_{i-1|k}$ using (4d). For the type of the set we consider an ellipsoid or a polytope and assume that the origin always belongs to the set. Ellipsoids and polytopes are the most commonly used sets in the related References 6 and 26 and—when parameterized—are bounded if these parameters are bounded.

Next, we parameterize the set and show the parameterized set by $\overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\Theta}_{i-1|k})$, where $\boldsymbol{\Theta}_{i-1|k}$ is the vector of all parameters at time step $i-1$. Therefore, the problem of approximating the disturbance set is translated into the problem of determining the vector $\boldsymbol{\Theta}_{i-1|k}$. This vector is returned by the mapping $f^{\text{dis}}(\cdot)$, which takes as input the corresponding state of the system (see (4d)). Mathematically, f^{dis} can in general be represented by any mappings (e.g., an analytical function or a deep neural network) that properly formulates the existing knowledge about the state-dependent disturbances, as long as the conditions explained in Section 3.3.1 are met by the mapping.

3.3.1 | Stability condition

To ensure robustness for SDD-TMPC to all values of the external disturbances, that is, to ensure that (3) holds, the parameter vector $\boldsymbol{\Theta}_{i-1|k}$ should be determined such that the actual disturbance set $\mathbb{W}(\boldsymbol{x}_{i-1|k})$ for all time steps $i \in \{k+1, \dots, k+N\}$ is a subset of the approximate disturbance set $\overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\Theta}_{i-1|k})$, which itself should be a subset of $\overline{\mathbb{W}}^{\max}$. The formulation of the robust control problem is based on the assumption of the existence of $\mathbb{W}^{\max}$, which bounds all possible disturbances. Thus, the evolution of the disturbance set generated by the above $f^{\text{dis}}(\cdot)$ is stable if the approximate disturbance set $\overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\Theta}_{i-1|k})$ is bounded by $\mathbb{W}^{\max}$ for all $i \in \{k+1, \dots, k+N\}$.

3.3.2 | TSK-FIS for search-and-rescue robotics

The rule base of the TSK-FIS of SDD-TMPC is composed of rules with the formulation given by (6), where $\widetilde{X}^l$ is a fuzzy set that mathematically represents a linguistic term that describes the input variable x:

$$\text{Rule } l: \ \text{IF } x \text{ is } \widetilde{X}^l, \text{ THEN } y^l \text{ is } h^l(x). \tag{6}$$

The output generated by each rule concerns only one element of parameter vector $\boldsymbol{\Theta}_{i-1|k}$. More specifically, for $i \in \{k+1, \dots, k+N\}$, the input x is replaced by $\boldsymbol{x}_{i-1|k}$, and the output y^l is replaced by an element $\theta^l_{i-1|k}$ of the parameter vector. Note that more than one rule in the rule base may generate a candidate value for this element of the parameter vector. Therefore, the superscript l is used to show that the value is generated for this element via the l^{th} rule within the set of all L rules that generate a candidate value for this element. Moreover, $h^l(\cdot)$ shows a generally nonlinear mapping from the input space to the output space (i.e., from the admissible set of the state variables of the dynamical system to the admissible set for element $\theta_{i-1|k}$ of the vector $\boldsymbol{\Theta}_{i-1|k}$). Then the final value for the parameter is computed by:

$$\theta_{i-1|k} = \frac{\sum_{l=1}^{L} \mu^l(\boldsymbol{x}_{i-1|k}) h^l(\boldsymbol{x}_{i-1|k})}{\sum_{l=1}^{L} \mu^l(\boldsymbol{x}_{i-1|k})}, \tag{7}$$

where $\mu^l(\cdot)$ is the membership function of fuzzy set $\widetilde{X}^l$ and L is the number of the rules in the rule base that generate a value for element $\theta_{i-1|k}$.

The approximate set $\overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\theta}_{i-1|k})$ is bounded, whenever all elements of vector $\boldsymbol{\Theta}_{i-1|k}$ are bounded (based on the assumption of an ellipsoid or polytope set or any other set that satisfies the condition of bounded output for bounded input). The membership functions in (7) are restricted to the interval $[0, 1]$ by definition, implying that to keep the elements of $\boldsymbol{\Theta}_{i-1|k}$ bounded, the functions $h^l(\cdot)$ (with $l \in \{1, \cdots, L\}$) should be formulated such that for all $\boldsymbol{x}_{i-1|k} \in \mathbb{X}$ the function remains bounded. Finally, to ensure that the approximate set $\overline{\mathbb{W}}_{i-1|k}(\boldsymbol{\Theta}_{i-1|k})$ that is designed to be bounded, is also a subset of $\mathbb{W}^{\max}$, this set is defined as the union of the output of the TSK-FIS and $\mathbb{W}^{\max}$. Therefore, the proposed TSK-FIS satisfies the stability conditions explained in Section 3.3.1.

3.4 | Stability of SDD-TMPC

In this section, we discuss the stability of SDD-TMPC, where our discussions are based on the following assumptions:

A1 There exists $\mathbb{T}_k^{\max} = \{\{z_{k+1|k}\} \oplus \mathbb{E}^{\max}, \ldots, \{z_{N|k}\} \oplus \mathbb{E}^{\max}\}$ that propagates across the prediction horizon at time step k and is positive robust invariant under the policy $\pi(\cdot) : \mathbb{X} \times \mathbb{U} \times \mathbb{X} \to \mathbb{U}$, that is, when the system follows policy $\pi(\cdot)$, the state remains inside this set.

A2 There exists a terminal control invariant set $\mathbb{Z}^{\mathrm{f}} \subseteq \mathbb{X}$ for the nominal system, such that $\mathbb{Z}^{\mathrm{f}} \oplus \mathbb{E}^{\max} \subseteq \mathbb{X}$.

A3 There exists a control law $\kappa : \mathbb{Z}^{\mathrm{f}} \to \mathbb{U}$ for the nominal system, such that for $\boldsymbol{z}_k \in \mathbb{Z}^{\mathrm{f}}$ and $i = k, \ldots, k+N-1$:

- $f(\boldsymbol{z}_{i|k}, \kappa(\boldsymbol{z}_{i|k})) \in \mathbb{Z}^{\mathrm{f}}$,
- $V_{\mathrm{f}}(f(\boldsymbol{z}_{i|k}, \kappa(\boldsymbol{z}_{i|k}))) - V_{\mathrm{f}}(\boldsymbol{z}_{i|k}) \leq -l(\boldsymbol{z}_{i|k}, \kappa(\boldsymbol{z}_{i|k}))$ (where $V_{\mathrm{f}}(\cdot)$ is the terminal cost as is given in (4a)),
- $\pi(\boldsymbol{x}_{i|k}, \kappa(\boldsymbol{z}_{i|k}), \boldsymbol{z}_{i|k}) \in \mathbb{U}, \boldsymbol{x}_{i|k} \in \{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}^{\max}$.

A4 There exists $\mathcal{K}_\infty$ functions $\alpha_1(\cdot)$ and $\alpha_{\mathrm{f}}(\cdot)$ that satisfy the following inequalities (note that a function belongs to class $\mathcal{K}$, if it is continuous, zero at zero, and strictly increasing; and a function belongs to class $\mathcal{K}_\infty$, if it is in class K and is unbounded[1]):

- $l(\boldsymbol{z}_{i|k}, \boldsymbol{v}_{i|k}) \geq \alpha_1(|\boldsymbol{z}_{i|k}|) \quad \forall \boldsymbol{z}_{i|k} \in \mathbb{X}, \forall \boldsymbol{v}_{i|k} \in \mathbb{U}$ (where $l(\cdot)$ is the stage cost as is given in (4a)).
- $V_{\mathrm{f}}(\boldsymbol{z}_{i|k}) \leq \alpha_{\mathrm{f}}(|\boldsymbol{z}_{i|k}|) \quad \forall \boldsymbol{z}_{i|k} \in \mathbb{Z}^{\mathrm{f}}$.

Assumption A1 implies that a stabilizable TMPC law can be determined for the system (i.e., the error does not go to infinity). Moreover, $\mathbb{T}_k^{\max}$ is the tube with a constant cross section that is used in TMPC. Assumption A2 (existence of a nominal invariant set) is a standard assumption in TMPC literature.[6] The first two items of Assumption A3 and Assumption A4 are standard assumptions in MPC literature[1] that are required to use the optimal cost as a Lyapunov function. The third item of Assumption A3 indicates that there exists a control law in the terminal set, such that the input constraints are satisfied.

We first give some theorems that are used in the discussions on stability.

Theorem 1. *If* $\mathbb{C} \subseteq \mathbb{A}$, *then* $\mathbb{C} \oplus \mathbb{B} \subseteq \mathbb{A} \oplus \mathbb{B}$.

Proof. Each element in $\mathbb{C} \oplus \mathbb{B}$ is given by $c + b$, where $c \in \mathbb{C}$ and $b \in \mathbb{B}$. Since $\mathbb{C} \subseteq \mathbb{A}$, then $c \in \mathbb{A}$. Thus, from the definition of Minkowski addition (5), we have $(c + b) \in \mathbb{A} \oplus \mathbb{B}$. ∎

Theorem 2. *If* $\mathbb{C} \subseteq \mathbb{A}$ *and* $\mathbb{D} \subseteq \mathbb{B}$, *then* $\mathbb{C} \oplus \mathbb{D} \subseteq \mathbb{A} \oplus \mathbb{B}$.

Proof. From the previous theorem, we have $\mathbb{C} \oplus \mathbb{B} \subseteq \mathbb{A} \oplus \mathbb{B}$ and $\mathbb{C} \oplus \mathbb{D} \subseteq \mathbb{C} \oplus \mathbb{B}$, which together imply that $\mathbb{C} \oplus \mathbb{D} \subseteq \mathbb{A} \oplus \mathbb{B}$. ∎

Theorem 3. *If* $\mathbb{E}_k \subseteq \mathbb{E}^{\max}$ *and the system follows the SDD-TMPC policy* $\pi(\cdot)$, *then* $\mathbb{E}_{i|k} \subseteq \mathbb{E}^{\max}$ *for* $i = k+1, \ldots, k+N$, *where* $\{z_{i|k}\} \oplus \mathbb{E}_{i|k}$ *are the elements of the tube* $\mathbb{T}_k$ *of SDD-TMPC*.

Proof. By contradiction, if the theorem is not true, that is, if there exists $\mathbb{E}_{i|k}$ for $i = k+1, \ldots, k+N$, such that $\mathbb{E}_{i|k} \nsubseteq \mathbb{E}^{\max}$ then there is at least one element $\boldsymbol{e}_{i|k} = \boldsymbol{x}_{i|k} - \boldsymbol{z}_{i|k}$ corresponding to a possible realization $\boldsymbol{x}_{i|k}$ of the state at time step i, such that $\boldsymbol{e}_{i|k} \in \mathbb{E}_{i|k}$, but $\boldsymbol{e}_{i|k} \notin \mathbb{E}_{\max,i|k}$. Therefore, for the corresponding external

disturbances the state leaves tube $\mathbb{T}^{\max}$, which is not possible because $\mathbb{T}^{\max}$ is a robust positive invariant set for the policy $\pi(\cdot)$ according to assumption A1. ∎

Theorem 4. *The set* $\Phi := \{\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k} \,|\, \boldsymbol{z}_{i|k} \in \mathbb{Z}^{\mathrm{f}},\ \mathbb{E}_{i|k} \subseteq \mathbb{E}^{\max},\ i = k+1, \dots, k+N\}$ *is a set robust positive invariant set for the SDD-TMPC inputs* $\pi(\boldsymbol{x}_{i|k}, \kappa(\boldsymbol{z}_{i|k}), \boldsymbol{z}_{i|k})$, *for* $i = k, \dots, k+N-1$. *(Note that according to Reference 6 a set robust positive invariant set is defined as a set of sets. Since the terminal real state* $x_{k|N+k}$ *should be bounded, we need to constrain both the nominal state and the potential errors, thus a set of tubes (i.e.,* Φ*) is needed.)*

Proof. From assumption A3 it follows that if $\boldsymbol{z}_k \in \mathbb{Z}^{\mathrm{f}}$, then $\boldsymbol{z}_{i|k} \in \mathbb{Z}^{\mathrm{f}}$ for $i = k+1, \dots, k+N$. Moreover, from Theorem 3 if $\mathbb{E}_k \subseteq \mathbb{E}^{\max}$, then $\mathbb{E}_{i|k} \subseteq \mathbb{E}^{\max}$ for $i = k+1, \dots, k+N$. Therefore, from the definition of Φ, we have $\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k} \in \Phi$, which proves the theorem. ∎

Theorem 5. *If the sequence of nominal control inputs* $\tilde{\boldsymbol{v}}_k = [\boldsymbol{v}_{k|k}^\top, \boldsymbol{v}_{k+1|k}^\top, \dots, \boldsymbol{v}_{k+N-1|k}^\top]^\top$ *of SDD-TMPC is feasible for the admissible pair* $(\boldsymbol{z}_k, \boldsymbol{x}_k)$, *that is, using these inputs and policy* $\pi(\cdot)$, *the controlled system satisfies* $\boldsymbol{x}_{i|k} \in \mathbb{X}$ *for* $i = k+1, \dots, k+N$, *then the shifted sequence of nominal control inputs* $\tilde{\boldsymbol{v}}_{k+1} = [\boldsymbol{v}_{k+1|k}^\top, \boldsymbol{v}_{k+2|k}^\top, \dots, \boldsymbol{v}_{k+N-1|k}^\top, \kappa^\top(\boldsymbol{z}_{k+N|k})]^\top$ *is feasible for the admissible pair* $(\boldsymbol{z}_{k+1}, \boldsymbol{x}_{k+1})$ *under the SDD-TMPC policy* $\pi(\cdot)$.

Proof. The nominal system is deterministic, and at time step $k+1$ for the first $N-1$ time steps the same nominal control inputs as for time step k are applied. Thus one can write $\boldsymbol{z}_{i|k+1} = \boldsymbol{z}_{i|k}$ for $i = k+1, \dots, k+N$. The error set $\mathbb{E}_{k+1|k}$ contains all the possible errors of $\boldsymbol{x}_{k+1|k}$ with respect to $\boldsymbol{z}_{k+1|k}$. At time step $k+1$, however, $\mathbb{E}_{k+1|k+1}$ contains only one element of $\mathbb{E}_{k+1|k}$ based on the realized state under the SDD-TMPC policy. Thus, we have $\mathbb{E}_{k+1|k+1} \subseteq \mathbb{E}_{k+1|k}$.

Since $\boldsymbol{z}_{k+1|k+1} = \boldsymbol{z}_{k+1|k}$ and $\mathbb{E}_{k+1|k+1} \subseteq \mathbb{E}_{k+1|k}$, from Theorem 2 we have $\{\boldsymbol{z}_{k+1|k+1}\} \oplus \mathbb{E}_{k+1|k+1} \subseteq \{\boldsymbol{z}_{k+1|k}\} \oplus \mathbb{E}_{k+1|k}$. From (4d)-(4e), the next error set is derived based on a mapping from all elements of the current error set added to the nominal state. Based on the last two statements, Definition 2, and Theorem 2, we conclude that $\mathbb{E}_{k+2|k+1} \subseteq \mathbb{E}_{k+2|k}$. Similarly, in an iterative way it can be shown that $\mathbb{E}_{i|k+1} \subseteq \mathbb{E}_{i|k}$ for $i = k+1, \dots, k+N$.

The sequence $\tilde{\boldsymbol{v}}_k$ is assumed to be feasible for the admissible pair $(\boldsymbol{z}_k, \boldsymbol{x}_k)$. Thus the elements $\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k}$ for $i = k+1, \dots, k+N-1$ belong to the admissible state set. Now since $z_{i|k+1} = z_{i|k}$ and $\mathbb{E}_{i|k+1} \subseteq \mathbb{E}_{i|k}$ for $i = k+1, \dots, k+N$, based on Theorem 4, the elements $\{\boldsymbol{z}_{i|k+1}\} \oplus \mathbb{E}_{i|k+1}$ for $i = k+1, \dots, k+N-1$ belong to the admissible state set. Finally, since at time step $k+N$ the SDD-TMPC input is determined based on the nominal control input $\kappa(\boldsymbol{z}_{k+N|k})$ (or equivalently, based on $\kappa(\boldsymbol{z}_{k+N|k+1})$), and since $\mathbb{E}_{k+N|k+1} \subseteq \mathbb{E}_{k+N|k} \subseteq \mathbb{E}^{\max}$, from Theorem 4 we conclude that $\{\boldsymbol{z}_{k+N|k+1}\} \oplus \mathbb{E}_{k+N|k+1}$ belongs to the admissible state set. ∎

Theorem 6. *The optimal cost function* $V^*(\cdot)$ *in (4a) is a Lyapunov function for system (1) (which implies stability).*

Proof. Given assumption A4, the optimal cost function $V^*(\cdot)$ is lower bounded by $\alpha_1(\cdot)$. Moreover, an upper bound can be found for $V^*(\cdot)$, since it is continuous and finite for $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{X}$. The optimal sequence of nominal control inputs corresponding to the optimal cost $V^*(\boldsymbol{x}_k, \boldsymbol{z}_k)$ is given by $\boldsymbol{v}_k^* = [\boldsymbol{v}^{*\top}_{k|k}, \boldsymbol{v}^{*\top}_{k+1|k}, \dots, \boldsymbol{v}^{*\top}_{k+N-1|k}]^\top$. Based on Theorem 5, for the next control time step the sequence $\tilde{\boldsymbol{v}}_{k+1} = [\boldsymbol{v}^{*\top}_{k+1|k}, \boldsymbol{v}^{*\top}_{k+2|k+1}, \dots, \boldsymbol{v}^{*\top}_{k+N-1|k}, \kappa^\top(\boldsymbol{z}_{N|k})]^\top$ is feasible for the admissible pair $(\boldsymbol{z}_{k+1}, \boldsymbol{x}_{k+1})$ under the SDD-TMPC policy $\pi(\cdot)$. The difference between $V^*(\boldsymbol{x}_k, \boldsymbol{z}_k)$ and the cost under sequence $\tilde{\boldsymbol{v}}_{k+1}$ is $l(\boldsymbol{z}_{k|k}, \boldsymbol{v}_{k|k}) + V^{\mathrm{f}}(\boldsymbol{z}_{k+N|k}) - l(\boldsymbol{z}_{k+N|k+1}, \kappa(\boldsymbol{z}_{k+N|k+1})) - V^{\mathrm{f}}(f(\boldsymbol{z}_{k+N|k+1}, \kappa(\boldsymbol{z}_{k+N|k}))$. From assumption A3, this difference is positive, implying that the optimal cost $V^*(\boldsymbol{x}_k, \boldsymbol{z}_k)$ is larger than the cost under $\tilde{\boldsymbol{v}}_{k+1}$, which itself is larger than or equal to the optimal cost $V^*(\boldsymbol{x}_{k+1}, \boldsymbol{z}_{k+1})$. Therefore, we have $V^*(\boldsymbol{x}_{k+1}, \boldsymbol{z}_{k+1}) < V^*(\boldsymbol{x}_k, \boldsymbol{z}_k)$. ∎

4 | CASE STUDIES

In this section, we provide the following two numerical case studies that have been designed for a SaR robot:

- **Case study 1:** A SDD-TMPC controller is developed for a linear model of the robot, which is impacted by state-dependent disturbances. With this experiment we showcase the unique capability of SDD-TMPC, in comparison with MPC and TMPC, to provide robustness with respect to the disturbances, while improving the feasibility and reducing the conservativeness. The codes for this experiment have been published in Reference 39.
- **Case study 2:** To showcase the performance of SDD-TMPC for nonlinear systems, an SDD-TMPC controller is developed for a nonlinear reference tracking problem, which has previously been addressed in Reference 40 using nonlinear TMPC. While the robot steered via nonlinear TMPC suffers from a large rise time, with SDD-TMPC, the robot achieves a better performance, by using less conservative control inputs. Consequently, the robot safely moves faster, still guaranteeing robustness to disturbances. Thus, our results indicate a decreased rise time using SDD-TMPC, compared to nonlinear TMPC, without violating the constraints. The code has been published in Reference 41.

The results of these case studies, where the performance of SDD-TMPC is compared with MPC and regular TMPC, are presented and discussed in Section 5. Table 3 includes the mathematical notations that are frequently used in this section. The parameters and values used in the case studies are given in Appendix A.

4.1 | Case study 1

4.1.1 | Simulation setup

We simulate a holonomic ground robot with the following discrete-time state-space representation for its kinematics:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 1 & 0 & T^{\mathrm{s}} & 0 \\ 0 & 1 & 0 & T^{\mathrm{s}} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ T^{\mathrm{s}} & 0 \\ 0 & T^{\mathrm{s}} \end{bmatrix} \boldsymbol{u}_k + \begin{bmatrix} \boldsymbol{w}_k \\ 0_{2\times 1} \end{bmatrix}, \tag{8}$$

where the state $\boldsymbol{x}_k = [p_{x,k}, p_{y,k}, v_{x,k}, v_{y,k}]^\top$ and control input $\boldsymbol{u}_k = [a_{x,k}, a_{y,k}]^\top$ vectors include, respectively, the position and velocity, and the acceleration of the robot in the x and y directions, T^{s} is the discretization sampling time and $\boldsymbol{w}_k$ is the 2-dimensional vector of the external disturbances that influence the position of the robot. We assume that the disturbances affect the position only (similarly as in Reference 40). The set $\mathbb{W}(\boldsymbol{x}_k)$ of external disturbances when the state of the robot is $\boldsymbol{x}_k$ is a two-dimensional ellipse, with its major parallel to the direction of the movement of the robot. The magnitude of

TABLE 3 Frequently used mathematical notations of the case study and their definition.

Notation	Definition	Notation	Definition
$[A^{\mathrm{ineq}}_{n_1\times n_2}, b^{\mathrm{ineq}}_{n_1\times 1}]$	Pair of matrices describing a polytope in n_2 dimensions with n_1 inequalities	m	meters
$p_{x,k}$	Component of robot position parallel to x axis at time step k	$\frac{\mathrm{m}}{\mathrm{s}}$	meters per second
$p_{y,k}$	Component of robot position parallel to y axis at time step k	$\frac{\mathrm{m}^2}{\mathrm{s}}$	meters per squared second
$v_{x,k}$	Component of robot velocity parallel to x axis at time step k	$r^{\max}(\boldsymbol{x}_k)$	The major of the ellipsoidal disturbance set given current state
$v_{y,k}$	Component of robot velocity parallel to y axis at time step k	$r^{\min}(\boldsymbol{x}_k)$	The minor of the ellipsoidal disturbance set given current state
$a_{x,k}$	Component of robot acceleration parallel to x axis at time step k	T^{s}	Sampling time
$a_{y,k}$	Component of robot acceleration parallel to y axis at time step k	$I_{n\times n}$	Identity matrix with dimension n

the disturbances is nonlinearly dependent on the state of the robot, that is,

$$r^{\max}(\boldsymbol{x}_k) = 0.202\sqrt{v_{x,k}^2 + v_{y,k}^2} + r^{\min}(\boldsymbol{x}_k), \quad r^{\min}(\boldsymbol{x}_k) = 0.225\beta^2(\boldsymbol{x}_k)\sqrt[4]{v_{x,k}^2 + v_{y,k}^2} \tag{9}$$

with $r^{\max}(\boldsymbol{x}_k)$ and $r_{\min}(\boldsymbol{x}_k)$ the length of the, respectively, major and minor of the ellipse. The external disturbances may vary across the environment; thus, the parameter $\beta(\boldsymbol{x}_k)$, an indication of the ground slipperiness, is defined to include this variation in the simulations. We assume that the robot has perfect knowledge of its own state and of $\beta(\boldsymbol{x}_k)$, but has no information on (9). The constraints on the position of the robot vary in different simulations, while the constraints on the inputs and the velocity of the robot are always the following:

$$|v_{x,k}| < 2\frac{\mathrm{m}}{\mathrm{s}},\ |v_{y,k}| < 2\frac{\mathrm{m}}{\mathrm{s}},\ |a_{x,k}| < 5\frac{\mathrm{m}}{\mathrm{s}^2},\ |a_{y,k}| < 5\frac{\mathrm{m}}{\mathrm{s}^2}. \tag{10}$$

4.1.2 | Formulating SDD-TMPC for path planning

For path planning of the SaR robot, we formulate an SDD-TMPC problem with a quadratic cost function given by:

$$V(\boldsymbol{z}_k, \boldsymbol{v}_k) = \sum_{i=k}^{k+N-1} \left(\|\boldsymbol{z}_{i|k}\|_Q^2 + \|\boldsymbol{v}_{i|k}\|_R^2 \right) + \|\boldsymbol{z}_{k+N|k}\|_F^2. \tag{11}$$

A common choice for the ancillary control input is a linear state feedback, which for time step k is given by:

$$\boldsymbol{u}_k = K\boldsymbol{e}_k + \boldsymbol{v}_k, \tag{12}$$

where $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{z}_k$. For determining K we use a linear quadratic regulator (LQR) method (the corresponding matrices are given in Appendix A), such that the ancillary control law more aggressively reduces the position. Thus, the influence of the state-dependent external disturbances is more significantly reduced. The error $\boldsymbol{e}_k$ evolves according to:

$$\boldsymbol{e}_{k+1} = (A + BK)\boldsymbol{e}_k + \left[\boldsymbol{w}_k^{\top} | 0_{1\times 2}\right]^{\top} \tag{13}$$

with $A_{4\times4}$ and $B_{4\times2}$ the dynamic and input matrices in (8). Thus the equivalent of (4c) and (4k) for this case study are:

$$\overline{\mathbb{E}}_{i+1|k} = (A + BK)\mathbb{E}_{i|k}, \qquad i = k, \ldots, k+N-1, \tag{14}$$

$$K\mathbb{E}_{i|k} \oplus \{\boldsymbol{v}_{i|k}\} \subseteq \mathbb{U}, \qquad i = k, \ldots, k+N-1. \tag{15}$$

Definition 2 is used to compute (14) and (15). In order to reduce the computation time and since the external disturbance set changes slowly with the states, we simplify (4d), only considering the centroid of the tubes as input:

$$\overline{\mathbb{W}}_{i|k} = f^{\mathrm{dis}}\left(\boldsymbol{z}_{i|k} + (A + BK)^{i-k}\boldsymbol{e}_k\right). \tag{16}$$

To determine the terminal admissible set $\mathbb{Z}^{\mathrm{f}}$ (a positive invariant polytope) for the nominal state, we assume that the system follows an LQR control law $\kappa(\cdot)$ beyond the prediction horizon. Thus, the terminal cost in (11) is the limit of the cost for $k \to \infty$, when the system follows the LQR control law $\kappa(\cdot)$, and is determined by solving the following Riccati equation, using, for example, Matlab dlyap command[42]:

$$F = (A + BK)^T F(A + BK) + Q_\kappa + K^T R_\kappa K. \tag{17}$$

For details on the derivation of (17), see Reference 1. Control policy $\kappa(\cdot)$ should be designed such that Assumption A3 holds.

4.1.3 | Learning the external disturbance sets using a FIS

A fuzzy inference system (FIS) can describe the dynamics of a (generally nonlinear) system via rules that are formulated via linguistic terms (e.g., very high), where this nonlinear mapping is interpretable –unlike, for example, neural networks.[43] The rules of a FIS can be generated based on experimental data gathered before the beginning of a mission or can be deduced directly from expert knowledge available in human language. In this case study, we consider two TSK FISs, one for approximating the major and one for the minor in (9) per time step k, with their output functions defined by:

$$r^m(\boldsymbol{x}_k) = c_1^m \nu_k^2 + c_2^m \beta^2(\boldsymbol{x}_k) + c_3^m \nu_k \beta(\boldsymbol{x}_k) + c_4^m \nu_k + c_5^m \beta(\boldsymbol{x}_k) + c_6^m \tag{18}$$

with $\nu_k^2 = \nu_{x,k}^2 + \nu_{y,k}^2$, $m \in \{\max, \min\}$, and $\boldsymbol{\theta}^m = [c_1^m \quad \dots \quad c_6^m]^\top$.

However, Minkowski summation of two ellipses does not result in an ellipse. In order to solve this problem, we approximate $\mathbb{W}(\boldsymbol{x}_k)$ by $\overline{\mathbb{W}}_k$, which is a polytopic sets defined by $\{[x,y]^\top : A_{8\times2}^{\text{ineq}}[x,y]^\top \leq \boldsymbol{b}_{8\times1}^{\text{ineq}}\}$ with a fixed number of edges that encounters the resulting ellipse and is used as the disturbance set at time step k. For polytope sets, unlike ellipses, the Minkowski addition can be computed without approximation, and the set remains a polytope afterward.

4.1.4 | Implementation of SDD-TMPC

The following four scenarios, relevant for SaR, were considered. Each scenario assesses one key property of SDD-TMPC. The scenarios were simulated several times to show the behavior of the controllers under different disturbances.

Scenario 1: Closely following a reference path

The main aim of the comparison among MPC, TMPC, and SDD-TMPC in Scenario 1 is to assess how fast a robot that is controlled by each of these approaches reaches a given destination, following a reference path (i.e., an ordered set of given positions). As soon as the (nominal) state of the robot reaches a distance of 0.3 m from its reference position, the next reference position from the path is followed by the controller. Scenario 1 resembles a real-life SaR situation when a robot should precisely travel along a given path in the presence of external disturbances (e.g., to avoid hazards that exist in the close vicinity of the robot). Scenario 1 was simulated 3 times, considering the following situations:

1. The robot was not affected by external disturbances.
2. The external disturbances helped in attracting the robot to the current reference position.
3. The external disturbances resulted in repelling the robot from the current reference position.

Scenario 2: Mission planning in the vicinity of obstacles

SaR robots should often move close to obstacles. In such cases, it should be guaranteed that no crashes occurs, whereas overly conservative decisions significantly slow down the mission. In Scenario 2, the robot should move from position $[2, 0.01]^\top$ to position $[8.5, 0.01]^\top$. Whenever the vertical position of the robot falls below $p_y = 0$, a crash with an obstacle (e.g., a wall extended across $p_y = 0$) occurs. Moreover, the value of $\beta(\boldsymbol{x}_k)$ is determined via:

$$\beta(\boldsymbol{x}_k) = \frac{1}{|5 - p_{x,k}| + 1}. \tag{19}$$

Scenario 2 was simulated 3 times, considering the following situations:

1. The robot was not affected by external disturbances.
2. The robot was pushed upwards by external disturbances.
3. The robot was pushed downwards by external disturbances.

Scenario 3: Approaching an unreachable target

SaR robots may need to move in very narrow corridors, avoiding crashes into the walls. TMPC may become infeasible in such cases due to conservativeness. Scenario 3 simulates such an infeasible problem for TMPC. The robot

starting at position $[5.5, 0]^\top$ with a zero initial speed should reach position $[11, 0]^\top$ in a narrowing corridor. To avoid crashes into the walls, the following hard constraints are defined on the robot position for all time steps k during the simulation:

$$p_{x,k} + 8p_{y,k} < 10; \quad p_{x,k} - 8p_{y,k} < 10. \tag{20}$$

Note that all coordinates are given in m. While TMPC is infeasible already at the initial position of the robot (the farthest position where TMPC generates a feasible tube is $p_x = 4.59$ m, whereas the width of the corridor is about 0.68 m (see Figure 5)), we investigate how far the robot moves forward using SDD-TMPC.

Scenario 3 was simulated twice, considering the following situations:

1. The robot was not affected by external disturbances.
2. The robot was pushed (relative to the speed value) upwards via external disturbances.

Scenario 4: Ability to make high-level optimal decisions

Next to robustness, SDD-TMPC should improve the performance by making high-level optimal decisions. In Scenario 4, we consider a case where the robot has to move from position $[2.3, 3]^\top$ to position $[3.25, 3.05]^\top$, with an obstacle on its way (see Figure 8). The robot initially has a horizontal speed of $v_x = 1 \frac{\text{m}}{\text{s}}$, and should select an initial vertical speed of either $v_y = 1.2 \frac{\text{m}}{\text{s}}$ (i.e., moving upwards) or $v_y = -1.2 \frac{\text{m}}{\text{s}}$ (i.e., moving downwards).

4.2 | Case study 2

4.2.1 | Simulation setup

In Reference 40, two controllers, nonlinear TMPC and nonlinear robust MPC, have been used for a nonlinear reference tracking problem. We simulated a similar experiment using SDD-TMPC. The only differences, to our knowledge, concern:

- Disturbance generator: In Reference 40, only the boundaries of the disturbances are given.
- Solver: Despite using the same algorithm (interior point) from Matlab's optimization toolbox, due to the nonconvex nature of the optimization problem, we cannot ensure to identify the exact same solutions.
- Tube initialization: In Reference 40, the solver chose the initial nominal position per time step, provided that the measured position remains within the tube. The starting nominal direction is, however, not given. We allowed the solver to freely choose the direction.
- Discretization method: In Reference 40, the system is discretized using the ICLOCS toolbox.[44] We used Matlab's c2d function, Zero-Order Hold (ZOH),[45] and Runge–Kutta 4 algorithm to discretize the linear and nonlinear systems.
- Terminal state sets
- Hardware

Two unicycle robots are simulated: a leader, which is assumed to remain unaffected by the external disturbances, and a follower, which we control and is affected by the disturbances. A unicycle robot is a circular robot, steered at time instant t by the velocities v_t^{l} and v_t^{r} of its, respectively, left and right wheels. The head of the robot is its front point, located on the main axis of the robot that is perpendicular to the axis of the wheels of the robot. The inputs to the robot at time instant t are the linear velocity v_t and the angular velocity ω_t, given by:

$$v_t = (v_t^{\text{l}} + v_t^{\text{r}})/2, \qquad \omega_t = (v_t^{\text{r}} - v_t^{\text{l}})/(2\rho), \tag{21}$$

where ρ represents the radius of the robot. Moreover, the admissible set of the inputs is defined by:

$$\mathbb{U} =: \left\{ [v, \omega]^T \in \mathbb{R}^2 \middle| \frac{|v|}{v^{\max}} + \frac{\rho|\omega|}{v^{\max}} \leq 1 \right\}, \tag{22}$$

where $v^{\max}$ is the maximum absolute velocity of the wheels.

The leader robot utilizes constant reference inputs v^{R} and ω^{R}. The state $\boldsymbol{x}_t^{\mathrm{R}}$ of the leader robot per time instant t is characterized by the position $(p_{x,t}^{\mathrm{R}}, p_{y,t}^{\mathrm{R}})$ of its center and by the orientation ψ_t^{R} of the robot. The state evolution, in the continuous time domain, for the leader robot at time instant t is determined by:

$$\begin{bmatrix} \dot{p}_{x,t}^{\mathrm{R}} \\ \dot{p}_{y,t}^{\mathrm{R}} \\ \dot{\psi}_t^{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} v^{\mathrm{R}} \cos(\psi_t^{\mathrm{R}}) \\ v^{\mathrm{R}} \sin(\psi_t^{\mathrm{R}}) \\ \omega^{\mathrm{R}} \end{bmatrix}. \tag{23}$$

The input $\boldsymbol{u}_t$ to the follower robot at time instant t consists of its linear v_t and angular ω_t velocities, and its state $\boldsymbol{x}_t$ includes the position $(p_{x,t}, p_{y,t})$ of its head and the robot orientation ψ_t. The position is impacted by the unknown disturbance vector $\boldsymbol{w}_t = [w_{x,t}, w_{y,t}]^T$, where $\sqrt{w_{x,t}^2 + w_{y,t}^2} < \eta$. The state evolution for the follower robot is given by:

$$\dot{\boldsymbol{x}}_t = \begin{bmatrix} \dot{p}_{x,t} \\ \dot{p}_{y,t} \\ \dot{\psi}_t \end{bmatrix} = \begin{bmatrix} v_t \cos(\psi_t) + \rho\omega_t \sin(\psi_t) \\ v_t \sin(\psi_t) + \rho\omega_t \cos(\psi_t) \\ \omega_t \end{bmatrix} + \begin{bmatrix} w_{x,t} \\ w_{y,t} \\ 0 \end{bmatrix} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{w}_t). \tag{24}$$

4.2.2 | Formulating SDD-TMPC

The nonlinear ancillary control law used for SDD-TMPC is the same as the one in Reference 40:

$$\boldsymbol{u}_t = \begin{bmatrix} \cos(\boldsymbol{x}_t[3]) & -\rho \sin(\boldsymbol{x}_t[3]) \\ \sin(\boldsymbol{x}_t[3]) & \rho \cos(\boldsymbol{x}_t[3]) \end{bmatrix}^{-1} \left(\begin{bmatrix} \cos(\boldsymbol{z}_t[3]) & -\rho \sin(\boldsymbol{z}_t[3]) \\ \sin(\boldsymbol{z}_t[3]) & \rho \cos(\boldsymbol{z}_t[3]) \end{bmatrix} \boldsymbol{v}_t - K^{\mathrm{e}} \boldsymbol{e}_t[1:2] \right), \tag{25}$$

where K^{e} is a constant design parameter and the notation $\boldsymbol{e}_t[1:2]$ implies elements 1 and 2 of vector $\boldsymbol{e}_t$. This ancillary control law linearizes the dynamics of the position error of the controlled system (see Appendix B for details). The dynamics of the position and direction errors are formulated via:

$$\dot{\boldsymbol{e}}_t[1:2] = K^{\mathrm{e}} \boldsymbol{e}_t[1:2] + [w_x, w_y]^T, \qquad \dot{\boldsymbol{e}}_t[3] = -\frac{1}{\rho} \boldsymbol{e}_t[3] \boldsymbol{v}_t[1] + w_t^{\mathrm{e}}, \tag{26}$$

where the description, derivation, and lower and upper bounds of w_t^{e} (which is state-dependent and contains the nonlinear terms in the directional error evolution equation) are given in detail in Appendix C. We give the above equations in the continuous time domain. At the end, the derived equations may be discretized in time.

In this case study, in order to speed up the computations, SDD-TMPC uses a box to represent the error set, which is larger than the real error set. However, since the disturbance set is a box and the error dynamics of the variables are independent, the error set is also a box set for the entire optimization iterations. This means that the entire error set can be described by 3 times as many state variables, which makes the computational complexity linear. In the previous case study, where we used the smallest possible sets, the computational complexity was growing exponentially with the size of the state vector and prediction horizon.

The nonlinear optimization problem (in the discrete time domain) solved by SDD-TMPC per time step is:

$$V^*(\boldsymbol{x}_k, \boldsymbol{z}_k) = \min_{\tilde{\boldsymbol{z}}_k, \tilde{\boldsymbol{v}}_k, \mathbb{T}_k} \sum_{i=k}^{k+N-1} \left(\left\| \boldsymbol{z}_{i|k}^{\mathrm{r}} \right\|_Q^2 + \left\| \boldsymbol{u}_{i|k}^{\mathrm{r}} \right\|_R^2 \right) + \left\| \boldsymbol{z}_{k+N|k}^{\mathrm{r}} \right\|_F^2, \tag{27a}$$

$$\text{s.t. for } i = k+1, \ldots, k+N:$$

$$\boldsymbol{z}_{i+1|k} = f^{\mathrm{d}}(\boldsymbol{z}_{i|k}, \boldsymbol{v}_{i|k}), \tag{27b}$$

$$\mathbb{T}_k = \{\{z_{k+1|k}\} \oplus \mathbb{E}_{k+1|k}, \ldots, \{z_{k+N|k}\} \oplus \mathbb{E}_{k+N|k}\}, \tag{27c}$$

$$\overline{\mathbb{E}}_{i|k} = \text{diag}\big(K^{\text{e,d}}, K^{\text{e,d}}, K^{\theta,\text{d}}\big)\mathbb{E}_{i-1|k}, \tag{27d}$$

$$\mathbb{E}_{i|k} = \overline{\mathbb{E}}_{i|k} \oplus \mathbb{W}_{i-1|k}(\mathbb{E}_{i-1|k}, \boldsymbol{z}_{i-1|k}, \boldsymbol{v}_{i-1|k}), \tag{27e}$$

$$\boldsymbol{x}_k, \boldsymbol{z}_k \text{ are given} \tag{27f}$$

$$\mathbb{E}_k = \{\boldsymbol{x}_k - \boldsymbol{z}_k\}, \tag{27g}$$

$$v_{i|k} \in \mathbb{V}(\mathbb{E}_{i|k}), \tag{27h}$$

$$\boldsymbol{z}^{\text{r}}_{k+N|k} \in \mathbb{Z}^{\text{f}}, \tag{27i}$$

where $f^{\text{d}}(\cdot)$ is the discretized dynamic function that is determined after time-discretization of (24), and $K^{\text{e,d}}$ and $K^{\theta,\text{d}}$ are coefficients that are multiplied by, respectively, discretized errors $\boldsymbol{e}_i[1]$ (or $\boldsymbol{e}_i[2]$) and $\boldsymbol{e}_i[3]$, after discretization of the error dynamics equation (26), and are explained in more detail in Appendix B. For control input constraint tightening (27h) the following set in the continuous time is proposed:

$$\mathbb{V}(\mathbb{E}_t) := \min_{\boldsymbol{e}_t[3]\in\mathbb{E}_t[3]} (\lambda(\boldsymbol{e}_t[3])\mathbb{U} \ominus \begin{bmatrix} -\cos(\boldsymbol{z}_t[3]) & -\sin(\boldsymbol{z}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{z}_t[3]) & -\frac{1}{\rho}\cos(\boldsymbol{z}_t[3]) \end{bmatrix} K^{\text{e}}(\mathbb{E}_t[1] \times \mathbb{E}_t[2]). \tag{28}$$

The details on the derivation of this tightened set are given in Appendix D.

5 | RESULTS AND DISCUSSIONS

All simulations were implemented in Matlab 2022b on the same computer (details can be found in the Table 4). In SDD-TMPC, the error set, that is, the cross section of the tube evolves in time. We have illustrated this evolution for a horizon of 30 in Figure 3. Since the tube is 4-dimensional, its projections on the planes of the positions and the velocities are

TABLE 4 Hardware description.

CPU	Clock speed
11th Gen Intel(R) Core(TM) i7-1185G7	3GHz

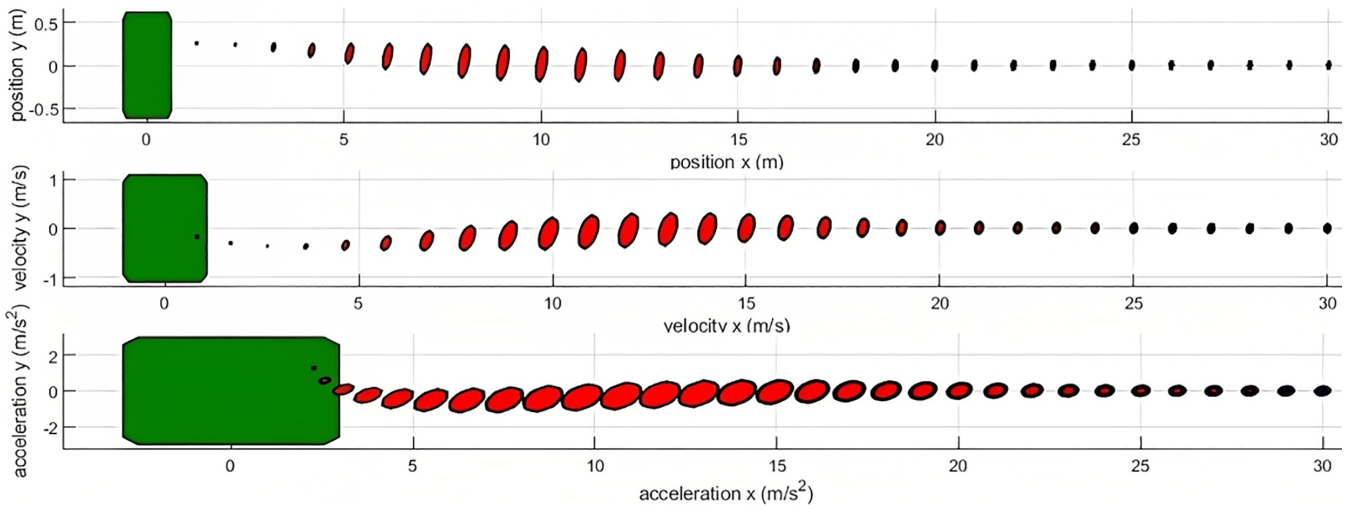

FIGURE 3 The projections of the fixed error set (i.e., cross section of the tube) of TMPC (shown in green) generated based on Reference 35 and the dynamic error set for SDD-TMPC (shown in blue) generated by the FIS, compared with the ground truth admissible sets of disturbances (shown in red), for a prediction horizon of 30. Since for regular TMPC the error set is fixed we have plotted it only once.

TABLE 5 Results for the trained FISs that estimate the lengths of the major and minor of the disturbance ellipse sets using the test data. The error values are normalized via dividing by the maximum ground truth value.

	FIS that estimates the major	FIS that estimates the minor
Number (in %) of cases where errors were negative	2.64	3.01
Mean value of negative errors	7.8×10^{-4}	1.93×10^{-3}
Mean value of positive errors	1.02×10^{-2}	2.32×10^{-2}
Maximum value of negative errors	5.89×10^{-3}	2.56×10^{-2}
Maximum value of positive errors	5.94×10^{-2}	2.45×10^{-1}

illustrated separately. Additionally, the admissible set of the control input (i.e., robot acceleration) is also demonstrated. The tube of SDD-TMPC evolves until there is convergence in all dimensions of the state, whereas this tube is never close to the tube of TMPC. The evolution of the tube can result in both its growth and shrinkage. In the simulations, the solution of TMPC is used as a warm start for SDD-TMPC, and in case TMPC is infeasible, the shifted trajectory from Theorem 5 is used.

5.1 | Case study 1: Training the FISs

First we give the results of training the FISs for case study 1. We generated 1240 input-output pairs, based on the ground truth ellipsoids given by (9). This data set was divided into a training and a validation set, with a proportion of 660 : 580. For the test set, we generated a large data set with 100000 pairs of input-output using (9) in order to extensively test the trained FISs. Given the training and the validation errors, the best results were achieved using 5 fuzzy sets per input, which resulted in 25 fuzzy rules with outputs that are described by (18). Note that compared to different expressions for the output of the rules, the expression given by (18) did not result in under or over-fitting. GA was used to train the FISs by minimizing the mean square error, with an additional penalty for negative errors.

The results obtained for the test set for the two FISs that estimate the lengths of the major and minor of the ellipse sets for the external disturbances are summarized in Table 5, where the error values have been normalized. In general, both FISs were able to approximate the ground truth with mostly positive-valued errors. In case negative-valued errors exist, a penalty for small positive errors may be considered to ensure to eliminate all these negative-valued errors as well.

5.2 | Case study 1: Comparison of MPC, TMPC, SDD-TMPC

In the simulations, regular MPC (29), TMPC (30), and SDD-TMPC (31) were implemented and their performance, with respect to minimizing the cost function, satisfying the hard constraints, and computation time were compared. In order to solve the optimization problems, the quadratic programming solver from Matlab's Optimization toolbox was used for MPC and TMPC, whereas for SDD-TMPC (which solves a nonconvex nonlinear optimization problem), the particle swarm (PS) algorithm[37] from Matlab's Global optimization toolbox was implemented. Unless stated otherwise, the optimization procedure for SDD-TMPC was terminated after 60 iterations, which took about 4 min to run and provided a balanced trade-off between the computation time and the accuracy of the solutions. In comparison, MPC and TMPC solved their quadratic problems in milliseconds. The initial nominal state was assumed to be equal to the real state. Terminal constraint and cost were not used in Scenarios 2 and 3, respectively, for computational efficiency and since the goal was infeasible. To compare the convergence of the optimization algorithms, in a sample problem the robot moved to the zero state, starting from state $[0.35, 0.65, 0, 0]^\top$. Table 6 shows the optimal costs for MPC, TMPC, and SDD-TMPC. Since SDD-TMPC solves a nonconvex, nonlinear optimization problem, it may not be possible to find a global minimum. Therefore, the changes in the value of the cost by iteration of the PS algorithm are given in the table (the number of iterations is given in parentheses). The rate of these changes depends strongly on how constrained the problem is (compare the third and sixth rows of the table). As expected, the cost values corresponding to TMPC and MPC are, respectively, an upper and a lower bound for the cost of SDD-TMPC.

TABLE 6 Comparing the cost values for MPC, TMPC, and different numbers of iterations of PS, which solves SDD-TMPC when the robot should reach the zero state from state $[0.35, 0.65, 0, 0]^\top$ (no stage position constraints).

Without terminal constraints and terminal cost						
MPC	TMPC	PS(40)	PS(80)	PS(120)	PS(160)	PS(200)
339.23	379.76	340	339.6	339.6	339.6	339.6
With terminal constraints and terminal cost						
MPC	TMPC	PS(40)	PS(80)	PS(120)	PS(160)	PS(200)
402.91	781.32	667.05	597.10	540.48	531.28	528.31

TABLE 7 The mission time required by the robot in Scenario 1 to closely track the entire reference path.

	Mission time (s) without terminal cost and constraint			**Mission time (s) with terminal cost and constraint**		
	MPC	**TMPC**	**SDD-TMPC**	**MPC**	**TMPC**	**SDDT-MPC**
Case 1	10.8	12.9	10.7	12.5	13.3	11.5
Case 2	7.8	12.9	10.7	8.6	13.3	11.3
Case 3	18	12.9	10.7	19.8	13.3	11.4

$$V^*(\boldsymbol{x}_k) = \min_{\tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{u}}_k} \sum_{i=k}^{k+N-1} \left(\|\boldsymbol{x}_{i|k}\|_Q^2 + \|\boldsymbol{u}_{i|k}\|_R^2 \right) + \|\boldsymbol{x}_{k+N|k}\|_F^2, \tag{29}$$

$$\text{s.t. for } i = k+1, \dots, k+N: \boldsymbol{x}_{i|k} = A\boldsymbol{x}_{i-1|k} + B\boldsymbol{u}_{i-1|k}, \quad \boldsymbol{x}_k \text{ is given}, \quad \boldsymbol{x}_{i|k} \in \mathbb{X}, \quad \boldsymbol{u}_{i|k} \in \mathbb{U}, \boldsymbol{x}_{N|k} \in \mathbb{Z}^{\mathrm{f}}, \tag{30}$$

$$V^*(\boldsymbol{z}_k) = \min_{\tilde{\boldsymbol{z}}_k, \tilde{\boldsymbol{v}}_k} \sum_{i=k}^{k+N-1} \left(\|\boldsymbol{z}_{i|k}\|_Q^2 + \|\boldsymbol{v}_{i|k}\|_R^2 \right) + \|\boldsymbol{z}_{k+N|k}\|_F^2 \text{ s.t. for } i = k+1, \dots, k+N:$$

$$\boldsymbol{z}_{i|k} = A\boldsymbol{z}_{i-1|k} + B\boldsymbol{v}_{i-1|k}, \quad \boldsymbol{z}_k \text{ is given}, \quad \boldsymbol{z}_{i|k} \in \mathbb{X} \ominus \mathbb{E}_{\max}, \quad \boldsymbol{v}_{i|k} \in \mathbb{U} \ominus \mathbb{E}_{\max}, \quad \boldsymbol{z}_{N|k} \in \mathbb{Z}^{\mathrm{f}}, \tag{31}$$

$$V^*(\boldsymbol{x}_k, \boldsymbol{z}_k) = \min_{\tilde{\boldsymbol{z}}_k, \tilde{\boldsymbol{v}}_k, \mathbb{T}_k} \sum_{i=k}^{k+N-1} \left(\|\boldsymbol{z}_{i|k}\|_Q^2 + \|\boldsymbol{v}_{i|k}\|_R^2 \right) + \|\boldsymbol{z}_{k+N|k}\|_F^2$$

$$\text{s.t. for } i = k+1, \dots, k+N:$$

$$\boldsymbol{z}_{i|k} = A\boldsymbol{z}_{i-1|k} + B\boldsymbol{v}_{i-1|k}, \quad \mathbb{T}_k = \{\{z_{k+1|k}\} \oplus \mathbb{E}_{k+1|k}, \dots, \{z_{N|k}\} \oplus \mathbb{E}_{k+N|k}\}, \quad \overline{\mathbb{E}}_{i|k} = (A+BK)\mathbb{E}_{i-1|k},$$

$$\overline{\mathbb{W}}_{i-1|k} = \mathrm{FIS}\big(\boldsymbol{z}_{i|k} + (A+BK)^{i-k}(\boldsymbol{x}_k - \boldsymbol{z}_k)\big), \quad \mathbb{E}_{i|k} = \overline{\mathbb{E}}_{i|k} \oplus \overline{\mathbb{W}}_{i-1|k}, \quad \mathbb{E}_k = \{\boldsymbol{x}_k - \boldsymbol{z}_k\},$$

$$\{\boldsymbol{z}_{i|k}\} \oplus \mathbb{E}_{i|k} \subseteq \mathbb{X}, \quad K\mathbb{E}_{i|k} \oplus \{\boldsymbol{v}_{i|k}\} \subseteq \mathbb{U}, \quad \boldsymbol{z}_{N|k} \in \mathbb{Z}^{\mathrm{f}} \quad \boldsymbol{x}_k, \boldsymbol{z}_k \text{ are given}.$$

Remark 1. For determining $\mathbb{Z}^{\mathrm{f}}$ we start by computing the Minkowski difference of $\mathbb{X}$ and $\mathbb{E}^{\max}$, to generate a candidate admissible terminal set, shown by $\mathbb{Z}^{\mathrm{c}}$. This set is then transformed into the set $\mathbb{Z}^{\mathrm{d}}$ through the system dynamics, using the terminal control law (see Section 4.1.2). In case the resulting set $\mathbb{Z}^{\mathrm{d}}$ is a subset of $\mathbb{Z}^{\mathrm{c}}$, then $\mathbb{Z}^{\mathrm{c}}$ is the admissible terminal set for the nominal states. Otherwise, the intersection of $\mathbb{Z}^{\mathrm{c}}$ and $\mathbb{Z}^{\mathrm{d}}$ is the new candidate admissible terminal set for the nominal states, and the procedure is repeated until the condition $\mathbb{Z}^{\mathrm{d}} \subseteq \mathbb{Z}^{\mathrm{c}}$ is satisfied.

Scenario 1: The mission time of the robot for the 3 MPC-based controllers is represented in Table 7: For MPC the mission time is strongly affected by the external disturbances (see the significant variations in the mission time in different cases), whereas TMPC and SDD-TMPC provide more consistent results independent of the realized disturbances. TMPC needs more time than SDD-TMPC to finish the mission. In fact, to keep the states within the tube of TMPC (determined for the worst disturbance case), the nominal acceleration and speed of the robot have to remain within 60% and 50% of their maximum allowed values.

Scenario 2: The results of the simulations for Scenario 2 are shown in Figure 4: MPC steers the robot close to the wall, and when the robot is pushed downwards due to external disturbances, it collides into the wall. TMPC provides a

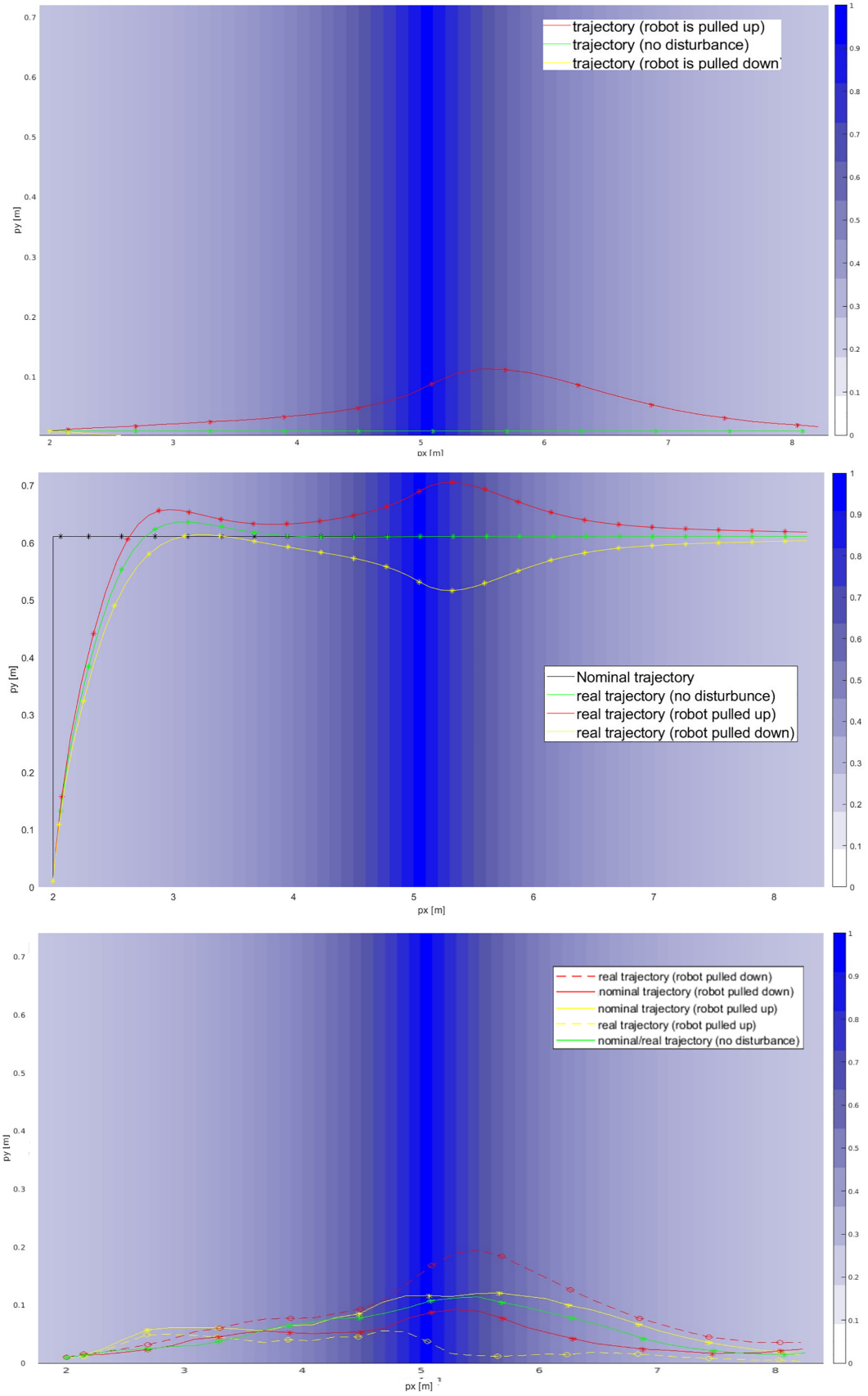

FIGURE 4 Scenario 2: Trajectories of the robot position when controlled by MPC (top plot), TMPC (middle plot) and SDD-TMPC (bottom plot), in case 1 (in green), 2 (in red), and 3 (in yellow). Note that the spectrum of blue corresponds to the values of $\beta(\boldsymbol{x}_k)$ for all realized states $\boldsymbol{x}_k$ of the robot during the simulation. In the middle plot, the black trajectory corresponds to the nominal case. For the bottom plot, the corresponding nominal and real trajectories are shown in the same color but with solid and dashed lines respectively.

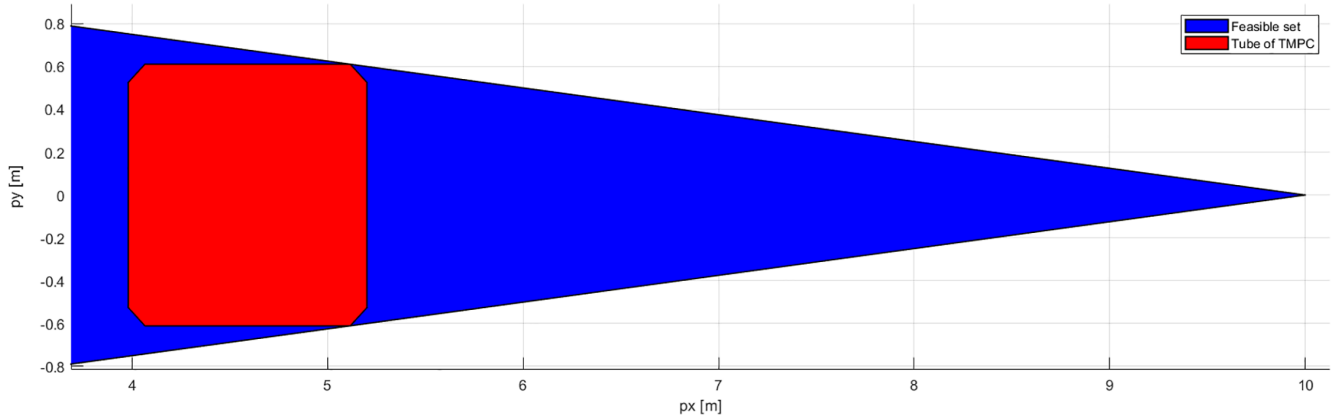

FIGURE 5 Scenario 3: The feasible state set $\mathbb{X}$ (in blue) and the tube of TMPC (in red).

safe, collision-free trajectory for the robot, moving relatively far from the wall. The initial nominal state of the robot with TMPC is considered far enough from the wall to avoid infeasibility (i.e., initial tube colliding with the wall). From the nominal and realized trajectories of the robot, when controlled via SDD-TMPC, it moves much closer to the wall compared to TMPC, but never crashes into it. Note that from (4), the nominal states of SDD-TMPC are determined based on an estimation of the external disturbances. Thus, the nominal trajectories vary with the disturbances (the more slippery the ground, the more careful the actions of SDD-TMPC).

Scenario 3: To reduce the risks of collision to the wall, the prediction horizon for this scenario is 6 (i.e., larger than other scenarios), since the target is outside of the feasible state set and there is no terminal constraint (Figure 5). In Figure 6, the evolution of the position and velocity of the robot when no external disturbances exist are shown, using SDD-TMPC to steer the robot. Since there is no wall in a close neighborhood of the robot, it first moves faster and then slows down in time (thanks to the larger prediction horizon) as the corridor narrows down. In fact, with SDD-TMPC the robot can move further through the corridor, compared to when TMPC is used. In case external disturbances push the robot upwards relative to its speed value, a similar pattern of behavior for the robot is observed, although the increase in the speed will be less significant than the case without external disturbances. Figure 7 shows the trajectories for the position of the robot for both cases, that is, with and without external disturbances: SDD-TMPC changes the nominal vertical position (in presence of external disturbances), which, in a vacuum, raises the cost (because of deviating from the reference trajectory). However, SDD-TMPC makes this decision for the nominal trajectory of the robot, because it foresees that significantly cancels out the impact of the external disturbances for the realized trajectory $p_{y,k}$ of the robot via the ancillary control input. Such behavior cannot be obtained via regular MPC and TMPC.

Scenario 4: When the robot moves upwards, the slipperiness coefficient is larger and thus, the robot is prone to larger disturbances, compared to when it moves downwards. The larger external disturbances will result in a longer realized path (due to an increased distance from the obstacle) for the robot. Since MPC and TMPC do not take into consideration the dynamics of the external disturbances, they estimate smaller costs when v_y is positive, whereas SDD-TMPC returns a lower cost (772) for the trajectory with smaller disturbances that corresponds to an initial vertical speed of $v_y = -1\ \frac{\text{m}}{\text{s}}$, compared to the cost (917) for the other trajectory (Figure 8).

5.3 | Case study 2: Transient response behavior

The leader and follower robots were simulated for 100 time steps, based on Reference 46, with $v^{\max} = 0.13\frac{\text{m}}{\text{s}}$ and $\rho = 0.0267\text{m}$. The aim was to maintain a constant distance $[-0.1, -0.1]^T$ (measured in the local coordinates of the leader robot) between the robots, in the local coordinate frame of the leader robot (the origin of this local frame coincides with the position of the center of the leader robot and the x-axis is parallel to the heading of the robot). The initial state of the leader and follower robots were, respectively, $[0, 0, \frac{\pi}{3}]$ and $[0.4, -0.2, -\frac{\pi}{2}]^T$. We compared the performance of SDD-TMPC and a nonlinear TMPC (see Appendix B for the formulation) for steering the follower robot. Since the control framework is discrete time, we used Matlab Ode45[47] to determine the evolved states per discrete time step, using the system of differential equations (23) and (24). To solve one optimization step, TMPC needed 0.2 s, while SDD-TMPC needed about 20 s. This is an improvement compared to the previous case study where it took 4 min for SDD-TMPC per optimization iteration for a twice smaller prediction horizon.

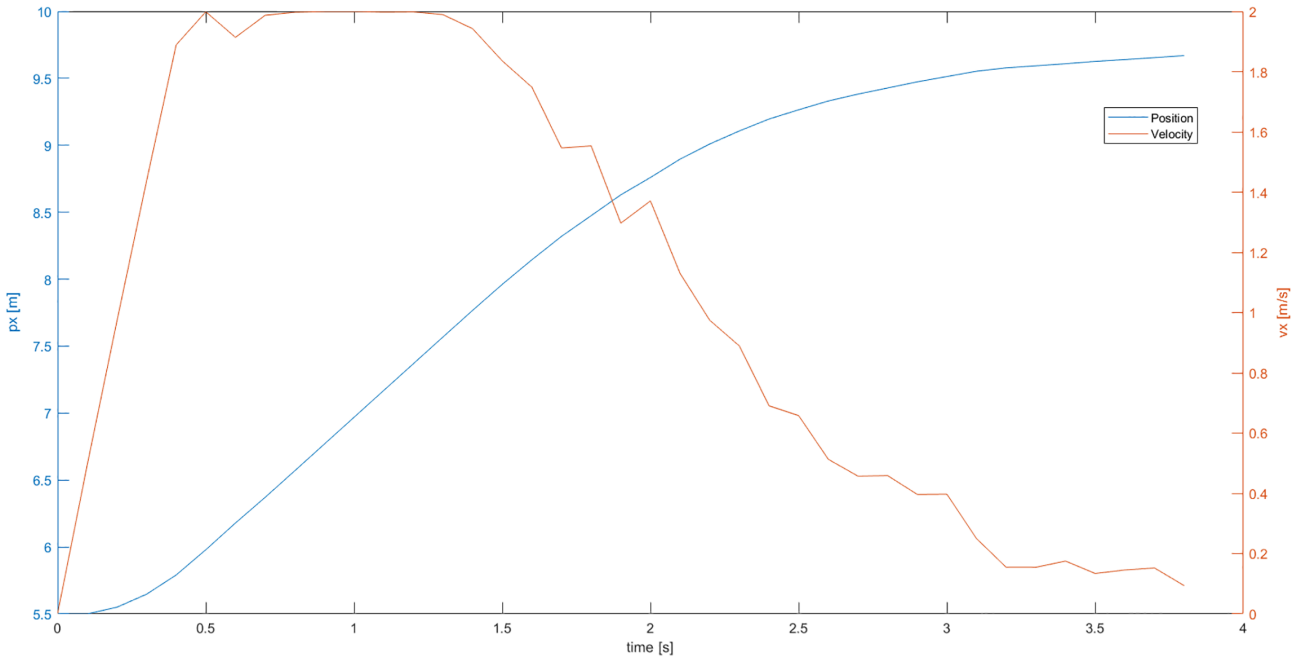

FIGURE 6 Scenario 3: The evolution of the position p_x (blue) and the velocity v_x (orange) of the robot, when SDD-TMPC is used and no external disturbances exist.

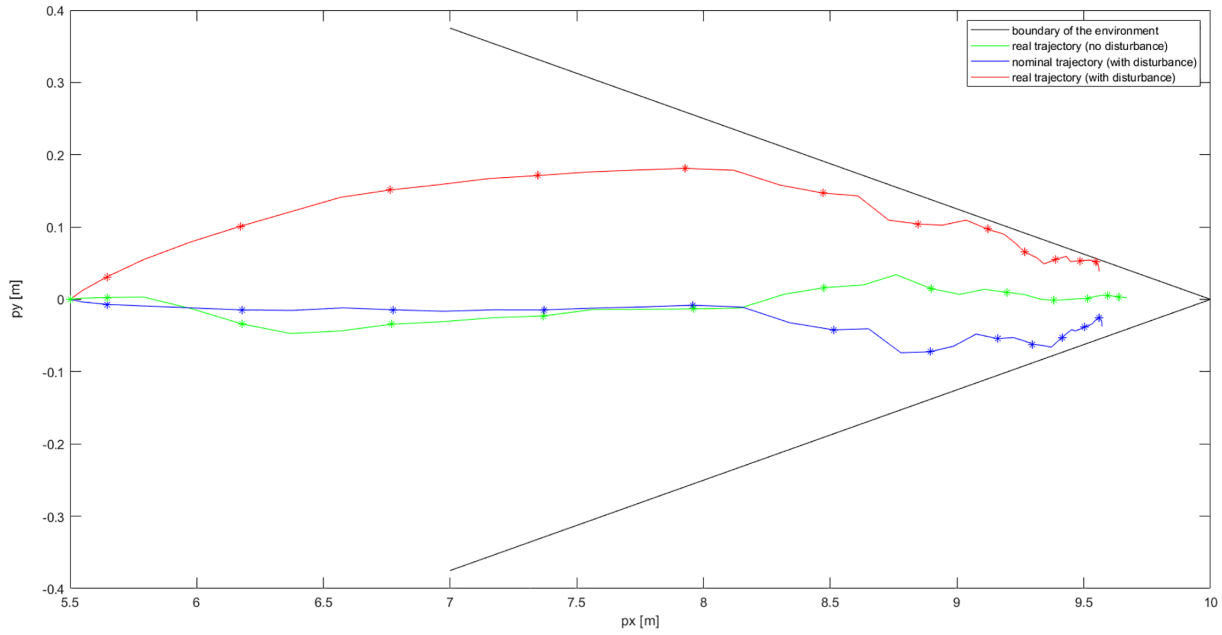

FIGURE 7 Scenario 3: Comparison of the trajectories of the position of the robot using SDD-TMPC, when no external disturbances affect the robot (green curve) and when external disturbances push the robot upwards relative to its speed (the blue curve for the nominal trajectory and the orange curve for the realized trajectory). Note that the black lines represent the boundaries of the corridor.

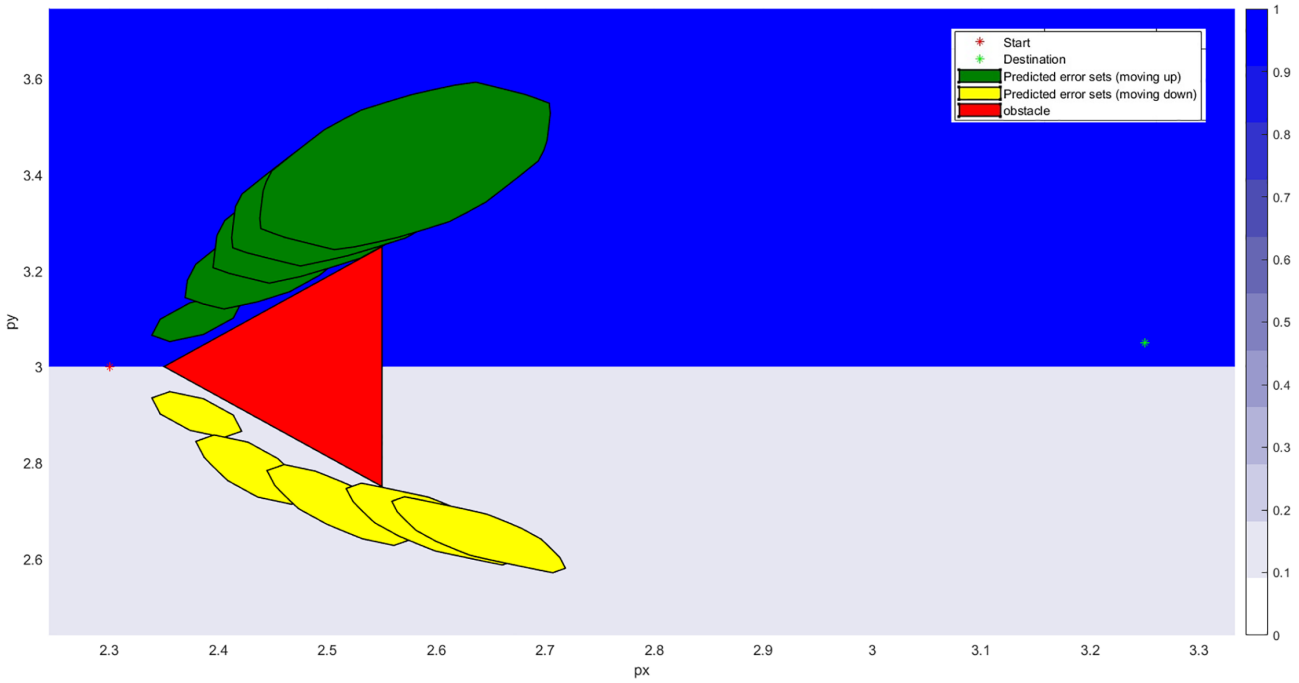

FIGURE 8 Scenario 4: The robot has to move from position $[2.3, 3]^\top$ to position $[3.25, 3.05]^\top$, while avoiding an obstacle (illustrated by the triangular red shape). The trajectories of the position projections of the tubes for two cases were shown in green (above the obstacle) and yellow (below the obstacle) colors.

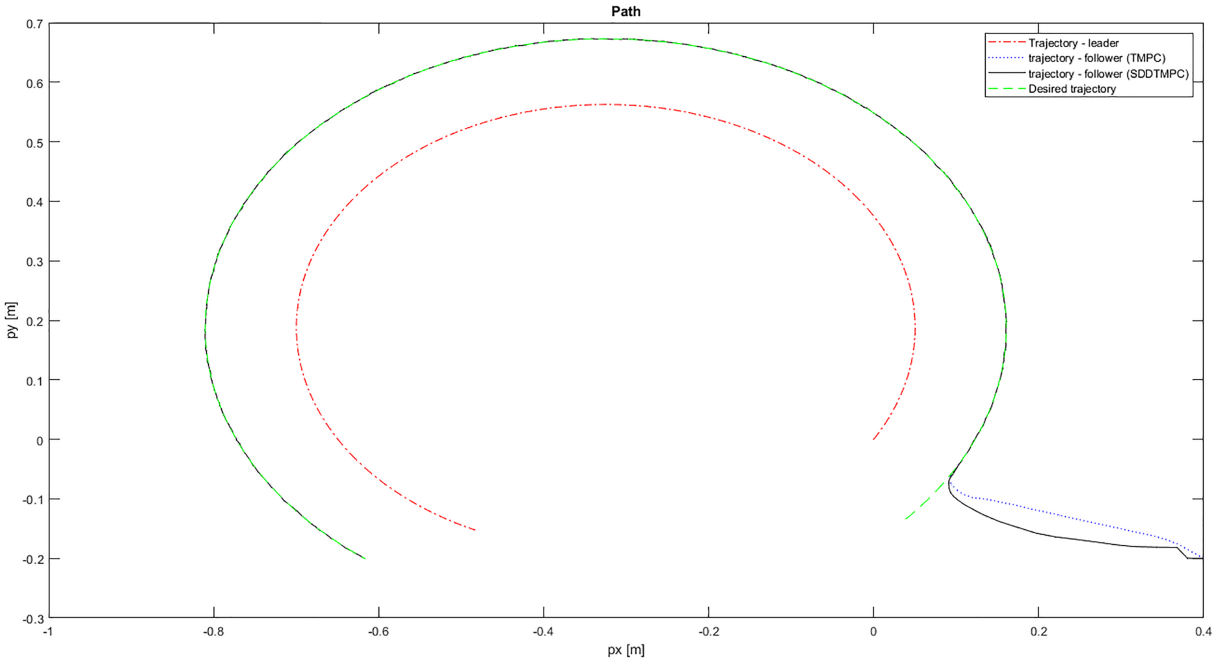

FIGURE 9 The leader trajectory, the desired trajectory for the follower, and the realized trajectories when the follower is steered via SDD-TMPC and via regular TMPC.

Figure 9 shows the path generated by these controllers: Both controllers were able to reach and follow the reference trajectory despite the disturbances. Figure 10 shows that the absolute value of the directional error is actually less than 0.6 rad, which confirms our assumption of small angles. Figure 11 shows the transient responses, where SDD-TMPC reaches the desired position in 10% less time compared to nonlinear TMPC. From Figure 12, once both controllers achieve the steady state, their performance is nearly equivalent. This is due to the fact that SDD-TMPC is authorized to employ substan-

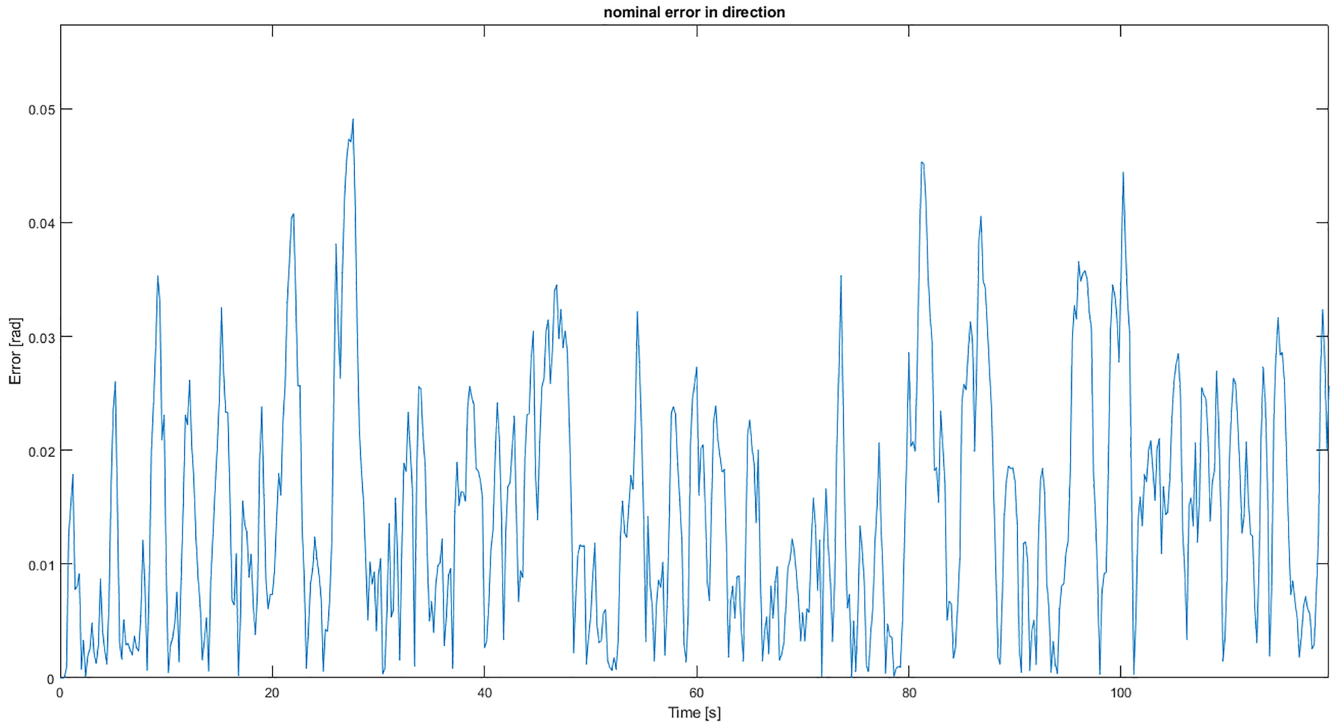

FIGURE 10 Directional error (i.e., difference between the nominal and actual heading for SDD-TMPC).

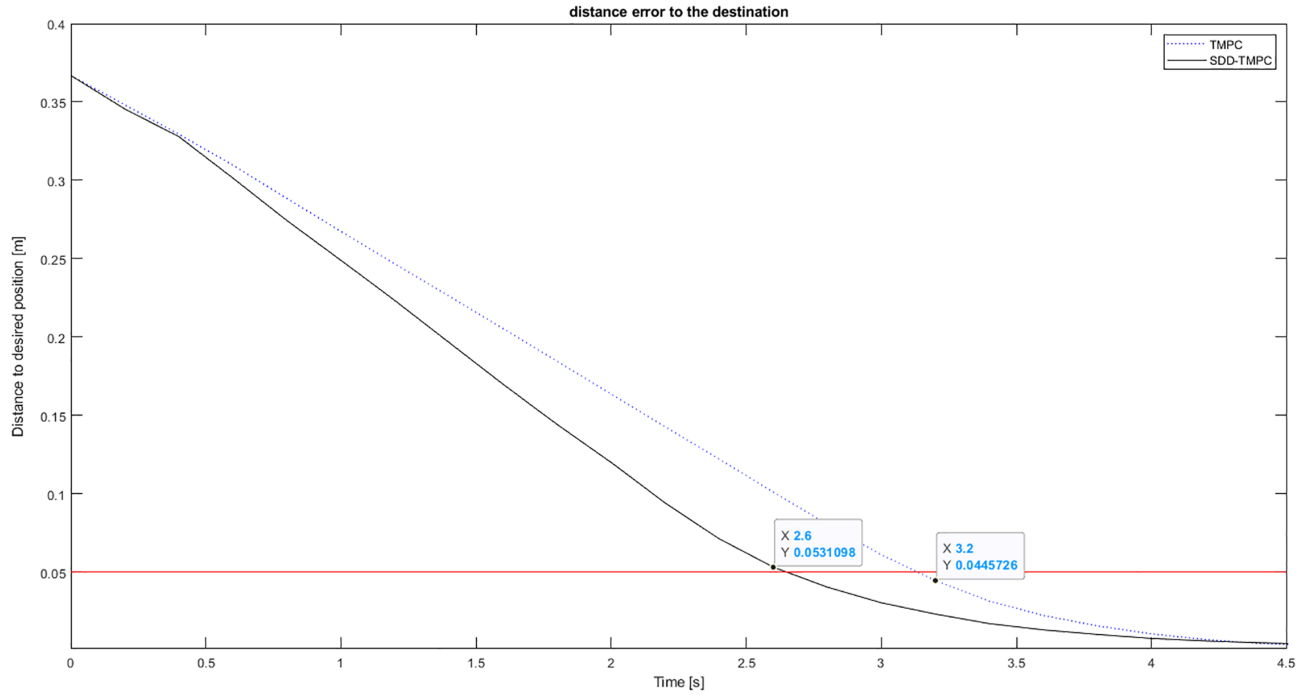

FIGURE 11 Distance between the desired and the transient positions using SDD-TMPC and regular TMPC.

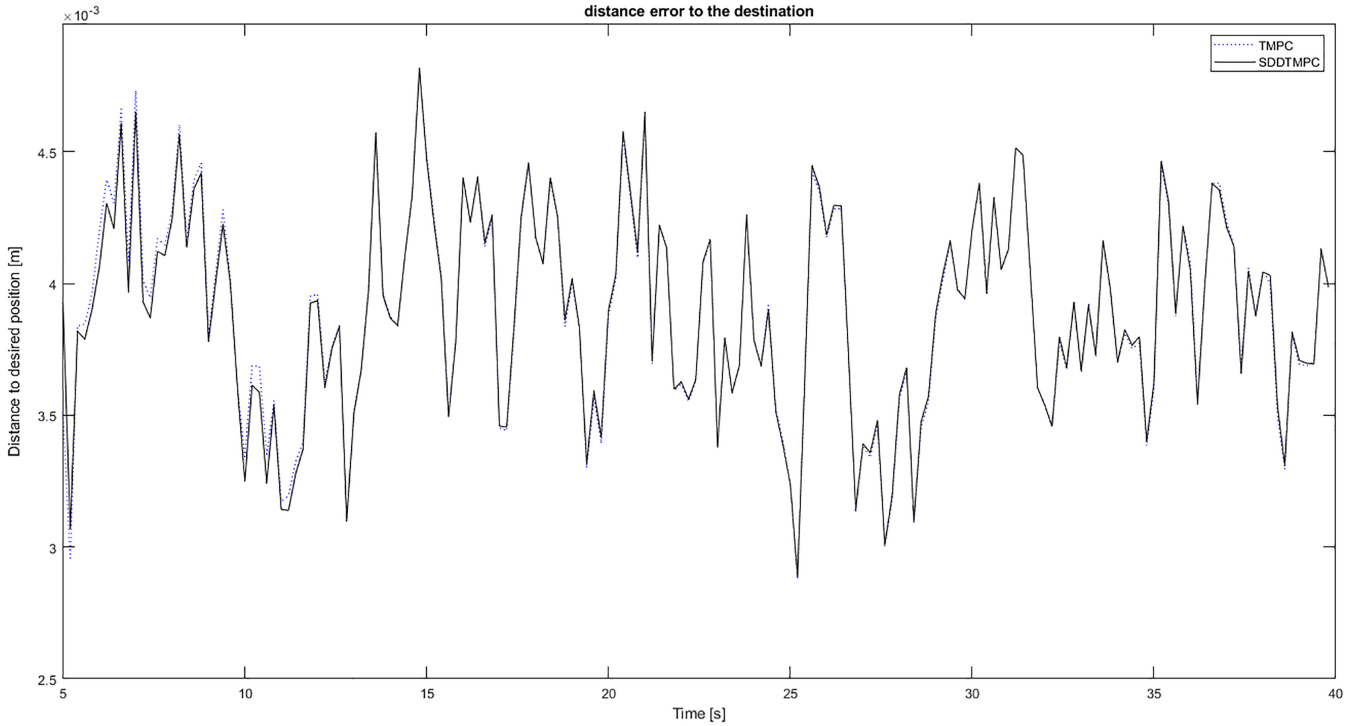

FIGURE 12 Difference between the desired and the steady-state positions using SDD-TMPC and regular TMPC.

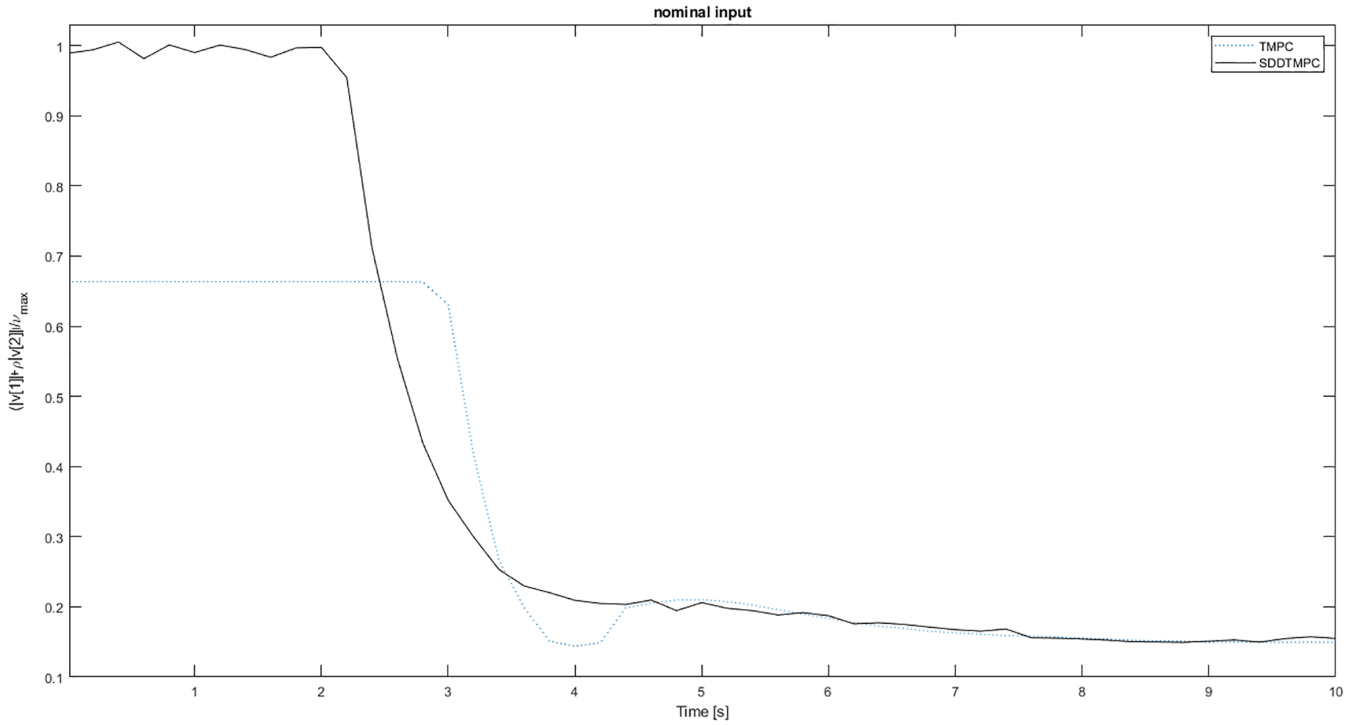

FIGURE 13 Nominal control input over time for SDD-TMPC and for regular TMPC.

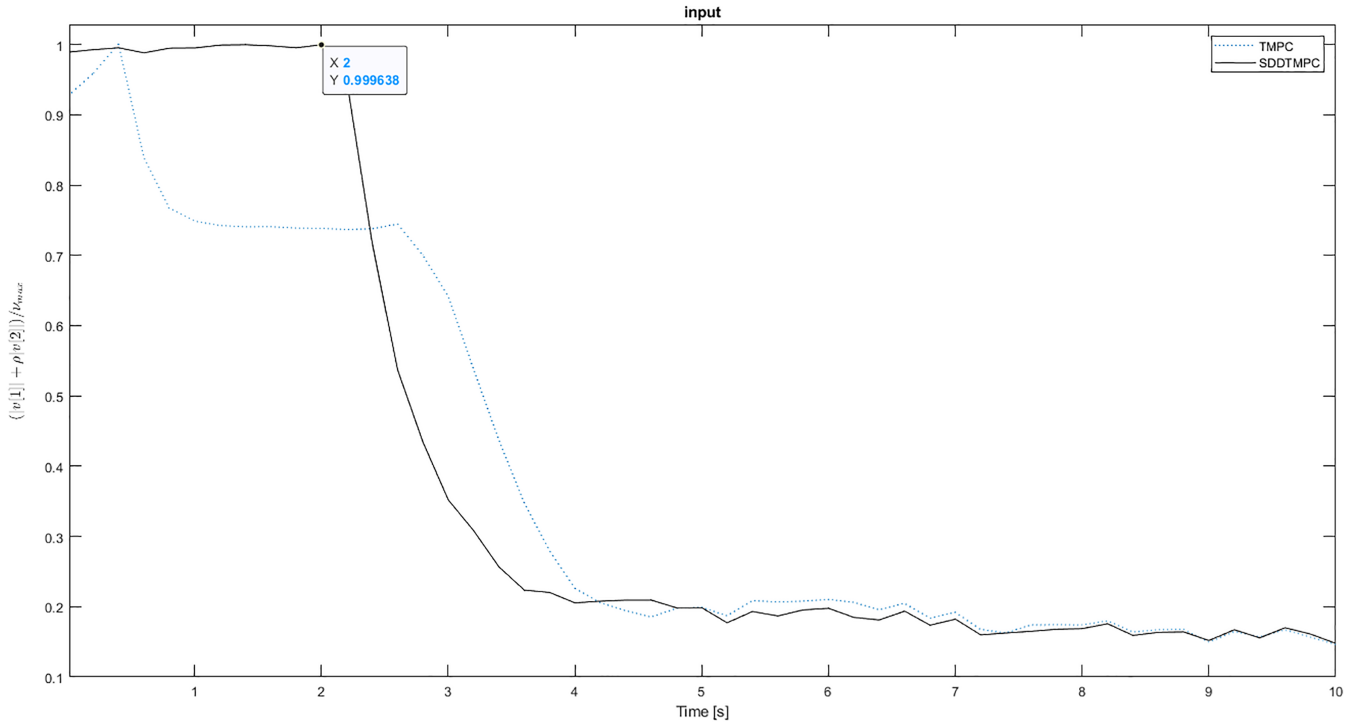

FIGURE 14 Control input over time for SDD-TMPC and for regular TMPC.

tially larger nominal inputs, as shown in Figure 13. Even though SDD-TMPC returns significantly larger inputs (above 30%), especially in the early stages, the actual input constraints are never violated (Figure 14) and, compared to nonlinear TMPC, the actual input is usually 20% larger, during the first 3 s of the mission, compared to the input of TMPC. This is particularly intriguing, because the nominal input of SDD-TMPC violates the original constraints $\mathbb{U}$. Nonetheless, SDD-TMPC anticipates the behavior of the ancillary controller and allows it to marginally violate the constraints, since the actual control input will not breach them. These results are noteworthy, since in Reference 40 nonlinear TMPC is compared to another robust MPC, where nonlinear TMPC exhibits an improved steady-state performance, but has a larger transient time. With SDD-TMPC however, the large rise time is successfully eliminated, whereas it maintains an equivalent steady-state performance.

6 | CONCLUSIONS AND FUTURE WORK

A dynamic version of tube model predictive control (TMPC), called state-dependent dynamic TMPC (SDD-TMPC), was proposed that uses a disturbance model (f^{dis}) to learn the dynamics of state-dependent external disturbances and to incorporate these dynamics into its future decisions. We used a fuzzy inference system (FIS) as an example of f^{dis}. It was generated offline based on a historical dataset. In the future, expert knowledge can be used to derive the rules and a reinforcement learning approach can be integrated into the FIS for online tuning. We also show an alternative approach by deriving state-dependent bounds by comparing real and simplified models in Appendix C.

We also proved the stability of SDD-TMPC. The performance of SDD-TMPC was evaluated compared to regular MPC and TMPC for steering an autonomous robot in various scenarios that include obstacles and external disturbances. SDD-TMPC and TMPC show robustness to state-dependent disturbances, whereas SDD-TMPC compromises the optimality less than TMPC. Thus, SDD-TMPC can reach states that are inaccessible for TMPC, resulting in reduced mission time (e.g., via a larger velocity or moving closer to the obstacles). In Table 8, we have included a summarized comparison between TMPC and SDD-TMPC, based on the theoretical discussions and results of the numerical experiments given in this article.

A main challenge of SDD-TMPC is the time required to solve online the optimization problem, which does not scale well with the number of states. Moreover, if polytopes are used to describe the error sets, the complexity of the optimization problem does not scale well with the size of the prediction horizon. In the future, the online computation time of

TABLE 8 Comparison between SDD-TMPC and TMPC based on the results of the case studies.

Criteria	Optimality	Computation time	Robustness
Outperforming controller	SDD-TMPC (reducing the final value of the cost function by up to 33% in case study 1 and reducing the time to reach steady state by 18% in case study 2)	TMPC (with TMPC operating in the range of milliseconds and SDD-TMPC in the range of a few minutes for Case study 1, whereas this computation time has already been reduced to seconds for SDD-TMPC in Case study 2)	Both controllers are the same

SDD-TMPC will be improved by providing approximate versions for SDD-TMPC, for example, by learning the nonlinear control policy using a neural network, as has been done for MPC in Reference 48 and for TMPC in Reference 49. The main research challenges for such an approximation will involve the proper choice of the learning method (e.g., deep versus spiking neural networks) and enhancing the guarantees of the trained system for satisfying the hard constraints and regarding stability. Another interesting topic concerns comparing SDD-TMPC and a similar controller that includes the dynamic tube within the cost, similarly as in Reference 27.

An additional interesting topic for future research is to replace the FIS with other approximators, including state-of-the-art state estimators for estimating an augmented state vector that includes the external disturbances. This allows to implement SDD-TMPC for various real-life applications, including systems for which no reliable intuitive knowledge is available, but such state estimators already exist.

Finally, SDD-TMPC will be implemented and assessed for large-scale search-and-rescue scenarios.

FUNDING INFORMATION

This research has been supported by the NWO Talent Program Veni project "Autonomous drones flocking for search-and-rescue" (18120), which has been financed by the Netherlands Organisation for Scientific Research (NWO).

CONFLICT OF INTEREST STATEMENT

The authors have declared no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in Gitlab repositories at http://doi.org/10.4121/82150c4b-eea2-46f4-8a47-fcb6bf3d8e3d.v1, reference number[39] and at http://doi.org/10.4121/3f1b28ee-2eca-4bac-bc2e-5ae2d2db4a5f, reference number.[41]

ORCID

Filip Surma https://orcid.org/0000-0002-6756-9598

REFERENCES

1. Rawlings J, Mayne DQ, Diehl M. *Model Predictive Control: Theory, Computation, and Design*. 2nd ed. Nob Hill Publishing; 2017.
2. Forbes M, Patwardhan R, Hamadah H, Gopaluni B. Model predictive control in industry: challenges and opportunities. *IFAC-PapersOnLine*. 2015;48:531-538.
3. Liu Y, Nejat G. Robotic urban search and rescue: a survey from the control perspective. *J Intell Robot Syst*. 2013;72(2):147-165.
4. Queralta JP. Collaborative multi-robot search and rescue: planning, coordination, perception, and active vision. *IEEE Access*. 2020;8:191617-191643.
5. Mesbah A. Stochastic model predictive control: an overview and perspectives for future research. *IEEE Control Syst Mag*. 2016;36(6):30-44.
6. Raković SV, Mayne DQ. A simple tube controller for efficient robust model predictive control of constrained linear discrete time systems subject to bounded disturbances. *IFAC Proc Vol*. 2005;38(1):241-246.
7. Kosko B. Fuzzy systems as universal approximators. *IEEE Trans Comput*. 1994;43(11):1329-1333.
8. Wang LX, Mendel JM. Back-propagation fuzzy system as nonlinear dynamic system identifiers. *IEEE International Conference on Fuzzy Systems*. IEEE; 1992:1409-1418.
9. Wang LX, Mendel JM. Generating fuzzy rules by learning from examples. *IEEE Trans Syst Man Cybern*. 1992;22(6):1414-1427.
10. Herrera F, Lozano M, Verdegay JL. Tuning fuzzy logic controllers by genetic algorithms. *Int J Approx Reason*. 1995;12(3):299-315.

11. Fathinezhad F, Derhami V, Rezaeian M. Supervised fuzzy reinforcement learning for robot navigation. *Appl Soft Comput*. 2016;40:33-41.
12. Zhou C, Meng Q. Dynamic balance of a biped robot using fuzzy reinforcement learning agents. *Fuzzy Sets Syst*. 2003;134:169-187.
13. Goharimanesh M, Mehrkish A, Janabi-Sharifi F. A fuzzy reinforcement learning approach for continuum robot control. *J Intell Robot Syst*. 2020;100:809-826.
14. Morari M, Lee JH. Model predictive control: past, present and future. *Comput Chem Eng*. 1999;23(4):667-682.
15. Mayne DQ. Model predictive control: Recent developments and future promise. *Automatica*. 2014;50(12):2967-2986.
16. Hegrenæs Ø, Gravdahl JT, Tøndel P. Spacecraft attitude control using explicit model predictive control. *Automatica*. 2005;41(12):2107-2114. https://www.sciencedirect.com/science/article/pii/S0005109805002657
17. Falcone P, Borrelli F, Asgari J, Tseng HE, Hrovat D. Predictive active steering control for autonomous vehicle systems. *IEEE Trans Control Syst Technol*. 2007;15(3):566-580.
18. Eskandarpour A, Sharf I. A constrained error-based MPC for path following of quadrotor with stability analysis. *Nonlinear Dyn*. 2020;01:99.
19. Trodden P, Richards A. Multi-vehicle cooperative search using distributed model predictive control. *AIAA Guidance, Navigation and Control Conference and Exhibit*. AIAA; 2008:8.
20. Leung C, Huang S, Dissanayake G. Active SLAM using model predictive control and attractor based exploration. *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE; 2006:5026-5031.
21. Lopez BT, Slotine JJE, How JP. Dynamic tube MPC for nonlinear systems. *2019 American Control Conference*. IEEE; 2019:1655-1662.
22. Hang P, Huang S, Chen X, Tan KK. Path planning of collision avoidance for unmanned ground vehicles: A nonlinear model predictive control approach. *Proc Inst Mech Eng I J Syst Control Eng*. 2021;235(2):222-236. 10.1177/0959651820937844
23. Mayne DQ, Kerrigan EC, van Wyk EJ, Falugi P. Tube-based robust nonlinear model predictive control. *Int J Robust Nonlinear Control*. 2011;21(11):1341-1353. https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.1758
24. Gonzalez R, Fiacchini M, Alamo T, Guzman JL, Rodriguez F. Online robust tube-based MPC for time-varying systems: a practical approach. *Int J Control*. 2011;84(6):1157-1170.
25. Raković S, Kouvaritakis B, Cannon M, Panos C. Fully parameterized tube model predictive control. *Int J Robust Nonlinear Control*. 2012;22(12):1330-1361. https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.2825
26. Parsi A, Iannelli A, Smith RS. Scalable tube model predictive control of uncertain linear systems using ellipsoidal sets. *Int J Robust Nonlinear Control*. 2022;33(3). https://onlinelibrary.wiley.com/toc/10991239/2023/33/3
27. Fan D, Agha A, Theodorou E. Deep learning tubes for tube MPC. *Robotics Science and Systems*. 2020. https://www.roboticsproceedings.org/rss16/index.html
28. Soloperto R, Müller MA, Trimpe S, Allgöwer F. Learning-based robust model predictive control with state-dependent uncertainty. *IFAC-PapersOnLine*. 2018;51(20):442-447.
29. Bonzanini AD, Graves DB, Mesbah A. Learning-based SMPC for reference tracking under state-dependent uncertainty: an application to atmospheric pressure plasma jets for plasma medicine. *IEEE Trans Control Syst Technol*. 2022;30(2):611-624.
30. Pin G, Raimondo DM, Magni L, Parisini T. Robust model predictive control of nonlinear systems with bounded and state-dependent uncertainties. *IEEE Trans Autom Control*. 2009;54(7):1681-1687.
31. Malyuta D, Açikmeşe B, Cacan M. Robust model predictive control for linear systems with state and input dependent uncertainties. *American Control Conference*. IEEE; 2019:1145-1151.
32. Falugi P, Mayne DQ. Getting robustness against unstructured uncertainty: a tube-based MPC approach. *IEEE Trans Autom Control*. 2014;59(5):1290-1295.
33. Köhler J, Müller MA, Allgöwer F. A novel constraint tightening approach for nonlinear robust model predictive control. *Annual American Control Conference*. IEEE; 2018:728-734.
34. Schlüter H, Allgöwer F. A constraint-tightening approach to nonlinear stochastic model predictive control under general bounded disturbances. *IFAC-PapersOnLine*. 2020;53(2):7130-7135.
35. Rakovic SV, Kerrigan EC, Kouramas KI, Mayne DQ. Invariant approximations of the minimal robust positively Invariant set. *IEEE Trans Autom Control*. 2005;50(3):406-410.
36. Gao Y, Gray A, Tseng HE, Borrelli F. A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles. *Veh Syst Dyn Int J Veh Mech Mobility*. 2014;52(6):802-823.
37. Kennedy J, Eberhart R. Particle swarm optimization. *Proceedings of ICNN'95–International Conference on Neural Networks*. Vol 4; IEEE; 1995:1942-1948.
38. Nocedal J, Wright SJ. *Numerical Optimization*. Springer Series in Operations Research. 2nd ed. Springer; 2006.
39. Surma F. Implementation of State-Dependent dynamic tube model predective control. 2023 10.4121/82150c4b-eea2-46f4-8a47-fcb6bf3d8e3d
40. Sun Z, Dai L, Liu K, Xia Y, Johansson KH. Robust MPC for tracking constrained unicycle robots with additive disturbances. *Automatica*. 2018;90:172-184. https://www.sciencedirect.com/science/article/pii/S0005109817306350
41. Surma F. Application of SDDTMPC to control a unicycle. 2023 10.4121/3f1b28ee-2eca-4bac-bc2e-5ae2d2db4a5f
42. Solve discrete-time Lyapunov equations. https://nl.mathworks.com/help/control/ref/dlyap.html
43. Ross TJ. *Fuzzy Logic with Engineering Applications*. Wiley; 2017.
44. CLOCS2: Your one-stop-shop solution for optimization based control in Matlab/Simulink. http://www.ee.ic.ac.uk/ICLOCS/Overview.html
45. Continuous-Discrete Conversion Methods. https://nl.mathworks.com/help/ident/ug/continuous-discrete-conversion-methods.html#bs78nig-2

46. Gonçalves P, Torres P, Alves C, et al. The e-puck, a robot designed for education in engineering. *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*. Vol 01; Instituto Politécnico de Castelo Branco; 2009:1.
47. Solve nonstiff differential equations – medium order method. https://www.mathworks.com/help/matlab/ref/ode45.html
48. Cao Y, Gopaluni RB. Deep neural network approximation of nonlinear model predictive control. *IFAC-PapersOnLine*. 2020;53(2):11319-11324.
49. Tagliabue A, Kim DK, Everett M, How JP. Demonstration-efficient guided policy search via imitation of robust tube MPC. *International Conference on Robotics and Automation*. IEEE; 2022:462-468.

How to cite this article: Surma F, Jamshidnejad A. State-dependent dynamic tube MPC: A novel tube MPC method with a fuzzy model of model of disturbances. *Int J Robust Nonlinear Control*. 2024;1-36. doi: 10.1002/rnc.7558

APPENDIX A. PARAMETERS/VALUES USED IN THE CASE STUDIES

This appendix includes all the parameters and values that have been used in the case studies.

In particular, for case study 1 we have:

- Discretization sampling time: $T^{\mathrm{s}} = 0.1$ s,
- Prediction horizon: $N = 5$,
- Cost matrices used in (11): $Q = \mathrm{diag}(100,100,1,1)$, $R = I_{2\times 2}$, $F = \begin{bmatrix} 805 & 0 & -571 & 0 \\ 0 & 805 & 0 & -571 \\ -571 & 0 & 655 & 0 \\ 0 & -571 & 0 & 655 \end{bmatrix}$,
- Gain matrix used in (12): $K = \begin{bmatrix} 7,98 & 0 & 4.42 & 0 \\ 0 & 7,98 & 0 & 4.42 \end{bmatrix}$,
- Matrices used to determine matrix K in (12) via LQR: $Q_K = \mathrm{diag}(100,100,0.1,0.1)$, $R_K = I_{2\times 2}$,
- Matrices Q_κ and R_κ used in (17): $Q_\kappa = \mathrm{diag}(10,10,1,1)$, $R_\kappa = I_{2\times 1}$,
- Ground slipperiness coefficient: $\beta = 1$ (unless otherwise states).

We used different cost matrices for

- ancillary control law to more aggressively minimize the position state which more strongly reduces the influence of external position disturbances in then the original cost matrices.
- terminal control law to reduce the size of the terminal set.

For case study 2 we have used the following values:

- Discretization sampling time: $T^{\mathrm{s}} = 0.2$ s,
- Prediction horizon: $N = 10$,
- Maximum value of the velocity of the robot wheels: $v^{\max} = 0.13$ m/s,
- Maximum value of the position disturbance vector: $\eta = 0.004$ m,
- Radius of the leader and follower robots: $\rho = 0.0267$ m,
- velocity of the leader robot: $v^{\mathrm{R}} = 0.015\frac{\mathrm{m}}{\mathrm{s}}$, $\omega^{\mathrm{R}} = 0.04\frac{\mathrm{rad}}{\mathrm{s}}$,
- Cost matrices used in (B1a): Q=diag(0.2,0.2,0), R=diag(0.4,0.4), F=diag(0.5,0.5),
- Gain used in (26): K^{e}=2.3,
- Gain used in (B2): $K^{\mathrm{e,d}}$=0.63,
- Gain used in (B6): $\overline{K} = 0.12$,
- $\mathbb{E}^{\max} = \{\boldsymbol{e} \in \mathbb{R}^3 | \max(|\boldsymbol{e}[1:2]|) \leq 0.0022\}$,
- $\mathbb{V} = 0.6636\mathbb{U}$,
- $\mathbb{Z}^{\mathrm{f}} = \{\boldsymbol{z}^{\mathrm{r}} \in \mathbb{R}^3 | |z^{\mathrm{r}}[1]| + |z^{\mathrm{r}}[2]| \leq 0.542\}$.

APPENDIX B. NONLINEAR TMPC FORMULATION FOR CASE STUDY 2

To control the motion of the follower robot, TMPC solves the following problem per iteration:

$$V^*(\boldsymbol{x}_k) = \min_{\tilde{\boldsymbol{z}}_k, \tilde{\boldsymbol{v}}_k} \sum_{i=k}^{k+N-1} \left(\left\| \boldsymbol{z}^{\mathrm{r}}_{i|k} \right\|^2_Q + \left\| \boldsymbol{v}^{\mathrm{r}}_{i|k} \right\|^2_R \right) + \left\| \boldsymbol{z}^{\mathrm{r}}_{k+N|k} \right\|^2_F, \tag{B1a}$$

$$\text{s.t. for } i = k+1, \ \dots \ , k+N:$$

$$\boldsymbol{x}_k \in \{\boldsymbol{z}_k\} \oplus \mathbb{E}^{\max}, \tag{B1b}$$

$$\boldsymbol{z}_{i+1|k} = f^{\mathrm{d}}(\boldsymbol{z}_{i|k}, \boldsymbol{v}_{i|k}), \tag{B1c}$$

$$\boldsymbol{v}_{i|k} \in \mathbb{V}, \quad \boldsymbol{z}^{\mathrm{r}}_{i+N|k} \in \mathbb{Z}^{\mathrm{f}}, \tag{B1d}$$

$$\boldsymbol{z}^{r}_{i|k}[1,2] = \mathrm{Rot}(-\boldsymbol{z}_{i|k}[3])(\boldsymbol{x}^{\mathrm{R}}_i[1,2] - \boldsymbol{z}_{i|k}[1:2]) + \mathrm{Rot}(\boldsymbol{z}^{\mathrm{r}}_{i|k}[3])\boldsymbol{p}_{\mathrm{d}}, \tag{B1e}$$

$$\boldsymbol{z}^{\mathrm{r}}_{i|k}[3] = \boldsymbol{x}^{\mathrm{R}}_i[3] - \boldsymbol{z}_{i|k}[3], \tag{B1f}$$

$$\boldsymbol{v}^{\mathrm{r}}_{i|k} = \begin{bmatrix} -\boldsymbol{v}_{i|k}[1] + (\nu_{\mathrm{R}} - p^{\mathrm{d}}_x \omega^{\mathrm{R}}) \cos(\boldsymbol{z}^{\mathrm{r}}_{i|k}[3]) - p^{\mathrm{d}}_x \omega^{\mathrm{R}} \sin(\boldsymbol{z}^{\mathrm{r}}_{i|k}[3]) \\ -\rho \boldsymbol{v}_{i|k}[2] + (\nu_{\mathrm{R}} - p^{\mathrm{d}}_x \omega^{\mathrm{R}}) \sin(\boldsymbol{z}^{\mathrm{r}}_{i|k}[3]) - p^{\mathrm{d}}_y \omega^{\mathrm{R}} \cos(\boldsymbol{z}^{\mathrm{r}}_{i|k}[3]) \end{bmatrix}, \tag{B1g}$$

where (B1b) states that the initial nominal state $\boldsymbol{z}_k$ of the follower robot, which is a decision variable, should be determined such that the error between the measured state $\boldsymbol{x}_k$ and the nominal state $\boldsymbol{z}_k$ of the follower robot belongs to set $\mathbb{E}^{\max}$. Moreover, according to (B1c), the nominal states of the follower robot evolve according to the discretized function $f^{\mathrm{d}}(\cdot)$, which is determined after time-discretization of (24) and by putting $\boldsymbol{w}_k = 0$ for all time steps. Constraint (B1d) enforces all the calculated nominal inputs to fall within set $\mathbb{V}$, which should be constructed, such that if $\boldsymbol{v}_{i|k} \in \mathbb{V}$, then for the actual input $\boldsymbol{u}_{i|k} \in \mathbb{U}$ for a given ancillary control law. Finally, $\mathbb{Z}^{\mathrm{f}}$ is the terminal set of the states for the follower robot. The cost function $V(\cdot)$ is computed using $\boldsymbol{z}^{\mathrm{r}}_{i|k}$, which is the nominal state of the follower robot within the coordinate system of the leader robot, while the original nominal state $\boldsymbol{z}_{i|k}$ is given in the global coordinates. The transition from the global coordinate system to the local coordinate system of the leader robot is done via (B1e)-(B1g), where $\mathrm{Rot}(\cdot)$ is a function that returns a 2-dimensional rotation matrix with respect to the global coordinates, given an input angle. Whenever square brackets are used after a vector, if the bracket contains one scalar, the notation refers to the element corresponding to that scalar of the vector. In case the square brackets include 'scalar 1:scalar 2', the notation refers to elements 'scalar 1' to 'scalar 2' (including these elements) of the vector.

For the nonlinear ancillary control law given at time instant t via (25), the dynamics of the position error for the controlled system is described by the first equation in (26). After discretization, the dynamics of the position error for time step $i \in \{k, \ \dots \ , k+N-1\}$ is given by:

$$\boldsymbol{e}_{i+1}[1:2] = K^{\mathrm{e,d}} \boldsymbol{e}_i[1:2] + T^{\mathrm{s}}[w_x, w_y]^T, \tag{B2}$$

where T^{s} represents the sampling time and $K^{\mathrm{e,d}}$ is calculated based on the zero-order hold approach (ZOH),[45] in order to ensure equivalence between the continuous-time and discrete-time formulations.

The cross section $\mathbb{E}^{\max}$ of the tube for nonlinear TMPC, set $\mathbb{V}$ of admissible values for the nominal control inputs, and the terminal control law (i.e., the control law that is implemented beyond the prediction horizon) are defined by:

$$\mathbb{E}^{\max} = \left\{ \boldsymbol{e} \in \mathbb{R}^3 \middle| \max(|\boldsymbol{e}[1:2]|) < \frac{T^{\mathrm{s}} \eta}{1 - K^{\mathrm{e,d}}} \right\}, \tag{B3}$$

$$\mathbb{V} = \left\{ \boldsymbol{v} \in \mathbb{R}^2 \middle| \left[\frac{1}{\nu^{\max}}, \frac{\rho}{\nu^{\max}} \right] |\boldsymbol{v}| < \frac{\sqrt{2}}{2} - \frac{\eta \sqrt{2}}{\nu^{\max}} \right\}, \tag{B4}$$

$$\boldsymbol{u}_k^{\mathrm{T}} = \begin{bmatrix} \overline{K}\boldsymbol{z}^{\mathrm{r}}_{k+N}[1] + (v^{\mathrm{R}} - p^{\mathrm{d}}\omega^{\mathrm{R}})\cos(\boldsymbol{z}^{\mathrm{r}}_{k+N}[3]) - p^{\mathrm{d}}_x\omega^{\mathrm{R}}\sin(\boldsymbol{z}^{\mathrm{r}}_{k+N}[3]) \\ \frac{1}{\rho}(\overline{k}\boldsymbol{z}^{\mathrm{r}}_{k+N}[2] + (v^{\mathrm{R}} - p^{\mathrm{d}}_x\omega^{R})\sin(\boldsymbol{z}^{\mathrm{r}}_{k+N}[3]) - p^{\mathrm{d}}_y\omega^{R}\cos(\boldsymbol{z}^{\mathrm{r}}_{k+N}[3])) \end{bmatrix}, \tag{B5}$$

where $\overline{K}$ is a constant. Finally, the terminal admissible set for the position of the follower robots is given by:

$$\mathbb{Z}^{\mathrm{f}} =: \left\{ \boldsymbol{z}^{\mathrm{r}} \in \mathbb{R}^3 \middle| \overline{K}(|\boldsymbol{z}^{\mathrm{r}}[1]| + |\boldsymbol{z}^{\mathrm{r}}[2]|) < \frac{v^{\max}\sqrt{2}}{2} - \eta\sqrt{2} - \sqrt{2}\left\| \begin{bmatrix} 1 & -p^{\mathrm{d}}_x \\ 0 & p^{\mathrm{d}}_y \end{bmatrix} \right\| \right\}. \tag{B6}$$

APPENDIX C. DIRECTIONAL ERROR DYNAMICS AND LOWER AND UPPER BOUNDS

In case study 2, for SDD-TMPC we approximate the error sets via box sets, which may slightly increase the conservatism, but significantly reduces the computation time. Let $\mathbb{E}_i[\ell]$ denote the projection of the error set $\mathbb{E}_i$ onto the ℓ^{th} dimension of the state space (thus for this case study $\ell = 1, 2, 3$). Since we work with box sets, $\mathbb{E}_i$ is the cross product over the projections of this set on all the dimensions of the state space. Based on (B2), for $\mathbb{W} = \{w \in \mathbb{R} | w \leq T^{\mathrm{s}}\eta\}$ and $\ell = 1, 2$, for the projection of the error set we can write:

$$\mathbb{E}_{i+1}[\ell] = K^{\mathrm{e,d}}\mathbb{E}_i[\ell] \oplus \mathbb{W}. \tag{C1}$$

In order to obtain the error dynamics for the third dimension of the state space (i.e., the robot direction), from the definition of this error and (24) we have:

$$\dot{\boldsymbol{e}}_t[3] = \dot{\boldsymbol{x}}_t[3] - \dot{\boldsymbol{z}}_t[3] = \boldsymbol{u}_t[2] - \boldsymbol{v}_t[2] \tag{C2a}$$

which, together with (25), and after applying the trigonometric identities $\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta)$ and $\sin(\alpha + \beta) = \sin(\alpha)\cos(\beta) + \cos(\alpha)\sin(\beta)$, as well as substituting $\boldsymbol{x}_t[3] = \boldsymbol{z}_t[3] + \boldsymbol{e}_t[3]$, results in:

$$\dot{\boldsymbol{e}}_t[3] = -\frac{1}{\rho}\sin(\boldsymbol{e}_t[3])\boldsymbol{v}_t[1] + (\cos(\boldsymbol{e}_t[3]) - 1)\boldsymbol{v}_t[2] - \frac{K^{\mathrm{e}}}{\rho}\sin(\boldsymbol{x}_t[3])\boldsymbol{e}_t[1] - \frac{K^{\mathrm{e}}}{\rho}\cos(\boldsymbol{x}_t[3])\boldsymbol{e}_t[2]. \tag{C2b}$$

Note that (C2b) is nonlinear and thus, computing all possible errors per time instant via (C2b) is in general computationally demanding. Moreover, we have chosen to work with polytopic error sets, while (C2b) does not necessarily result in a polytopic error set per time instant. Thus, as explained in Reference 36, we rewrite the dynamic equation in (C2b) as a linear evolutionary equation for $\boldsymbol{e}_t[3]$, where the nonlinear terms are treated as state-dependent disturbances, that is,

$$\dot{\boldsymbol{e}}_t[3] = -\frac{1}{\rho}\boldsymbol{e}_t[3]\boldsymbol{v}_t[1] + w^{\mathrm{e}}_t, \tag{C2c}$$

where $w^{\mathrm{e}}_t = \sum_{i=1}^{4} w^{\mathrm{e},i}_t$, with $w^{\mathrm{e},1}_t = -\frac{1}{\rho}(\sin(\boldsymbol{e}_t[3]) - \boldsymbol{e}_t[3])\boldsymbol{v}_t[1]$ and $w^{\mathrm{e},2}_t = (\cos(\boldsymbol{e}_t[3]) - 1)\boldsymbol{v}_t[2]$ and $w^{\mathrm{e},3}_t = -\frac{K^{\mathrm{e}}}{\rho}\sin(\boldsymbol{x}_t[3])\boldsymbol{e}_t[1]$ and $w^{\mathrm{e},4}_t = -\frac{K^{\mathrm{e}}}{\rho}\cos(\boldsymbol{x}_t[3])\boldsymbol{e}_t[2]$. This approach has proven to be accurate for small deviations in the third dimension of the state space (in this case the heading/direction of the robot). Henceforth, we assume that the absolute values of $\boldsymbol{e}_t[3]$ remain relatively low (i.e., $|\boldsymbol{e}_t[3]| < \pi/4$).

For determining the lower and upper bounds of the external disturbances, we find a lower bound and an upper bound per term $w^{\mathrm{e},i}_t$ for $i = 1, 2, 3, 4$. Since $\boldsymbol{e}_t[3] < \pi/4$, we can write:

$$\begin{aligned} w^{\mathrm{e},1,\min}_t &= -\tfrac{1}{\rho}(\sin(\min(\mathbb{E}_t[3])) - \min(\mathbb{E}_t[3]))\boldsymbol{v}_t[1] \\ w^{\mathrm{e},1,\max}_t &= -\tfrac{1}{\rho}(\sin(\max(\mathbb{E}_t[3])) - \max(\mathbb{E}_t[3]))\boldsymbol{v}_t[1] \end{aligned} \quad \text{for } \boldsymbol{v}_t[1] \geq 0, \tag{C3a}$$

$$\begin{aligned} w^{\mathrm{e},1,\min}_t &= -\tfrac{1}{\rho}(\sin(\max(\mathbb{E}_t[3])) - \max(\mathbb{E}_t[3]))\boldsymbol{v}_t[1] \\ w^{\mathrm{e},1,\max}_t &= -\tfrac{1}{\rho}(\sin(\min(\mathbb{E}_t[3])) - \min(\mathbb{E}_t[3]))\boldsymbol{v}_t[1] \end{aligned} \quad \text{for } \boldsymbol{v}_t[1] \leq 0. \tag{C3b}$$

We can bound $w_t^{e,2}$ considering that $\cos(\boldsymbol{e}_t[3]) - 1$ is a descending function for the given range of $\boldsymbol{e}_t[3]$:

$$\begin{aligned} w_t^{e,2,\min} &= (\cos(|\max(\mathbb{E}_t[3])|) - 1)\max(\boldsymbol{v}_t[2], 0), \\ w_t^{e,2,\max} &= (\cos(|\min(\mathbb{E}_t[3])|) - 1)\min(\boldsymbol{v}_t[2], 0). \end{aligned} \tag{C3c}$$

Finally, for $w_t^{e,3}$ and $w_t^{e,4}$ we can write:

$$\begin{aligned} w_t^{e,3,\min} &= \min_{(\boldsymbol{e}_t[1],\boldsymbol{e}_t[3])\in\mathbb{E}_t[1]\times\mathbb{E}_t[3]} \left(-\frac{K^e}{\rho}\sin(\boldsymbol{z}_t[3] + \boldsymbol{e}_t[3])\boldsymbol{e}_t[1]\right), \\ w_t^{e,3,\max} &= \max_{(\boldsymbol{e}_t[1],\boldsymbol{e}_t[3])\in\mathbb{E}_t[1]\times\mathbb{E}_t[3]} \left(-\frac{K^e}{\rho}\sin(\boldsymbol{z}_t[3] + \boldsymbol{e}_t[3])\boldsymbol{e}_t[1]\right), \end{aligned} \tag{C3d}$$

$$\begin{aligned} w_t^{e,4,\min} &= \min_{(\boldsymbol{e}_t[2],\boldsymbol{e}_t[3])\in\mathbb{E}_t[2]\times\mathbb{E}_t[3]} \left(-\frac{K^e}{\rho}\cos(\boldsymbol{z}_t[3] + \boldsymbol{e}_t[3])\boldsymbol{e}_t[2]\right), \\ w_t^{e,4,\max} &= \max_{(\boldsymbol{e}_t[2],\boldsymbol{e}_t[3])\in\mathbb{E}_t[2]\times\mathbb{E}_t[3]} \left(-\frac{K^e}{\rho}\cos(\boldsymbol{z}_t[3] + \boldsymbol{e}_t[3])\boldsymbol{e}_t[2]\right). \end{aligned} \tag{C3e}$$

Therefore, the admissible set of the external disturbances for (C2c) is defined via:

$$\mathbb{W}_t[3] := \left\{ w_t^e \in \mathbb{R} \Big| \sum_{i=1}^{4} w_t^{e,i,\min} \le w_t^e \le \sum_{i=1}^{4} w_t^{e,i,\min} \right\}. \tag{C3f}$$

For the discrete-time framework of the problem, (C3f) is estimated for the discrete time steps.

APPENDIX D. INPUT CONSTRAINT TIGHTENING

Here we explain the approach that has been used to tighten the input constraints of SDD-TMPC online for case study 2. Considering the ancillary control law that is formulated by (25), imposing the hard constraint $\boldsymbol{u}_t \in \mathbb{U}$, using the equality $\boldsymbol{x}_t[3] = \boldsymbol{z}_t[3] + \boldsymbol{e}_t[3]$, multiplying both sides of the hard constraints by $\begin{bmatrix} \cos(\boldsymbol{e}_t[3]) & -\rho\sin(\boldsymbol{e}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{e}_t[3]) & \cos(\boldsymbol{e}_t[3]) \end{bmatrix}$ from left, and considering all possible combinations of $\boldsymbol{e}_t[1]$, $\boldsymbol{e}_t[2]$, and $\boldsymbol{e}_t[3]$, the following condition is obtained:

$$\{\boldsymbol{v}_t\} \oplus \begin{bmatrix} -\cos(\boldsymbol{z}_t[3]) & -\sin(\boldsymbol{z}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{z}_t[3]) & -\frac{1}{\rho}\cos(\boldsymbol{z}_t[3]) \end{bmatrix} K^e(\mathbb{E}_t[1]\times\mathbb{E}_t[2]) \subseteq \bigcap_{\boldsymbol{e}[3]\in\mathbb{E}_t[3]} \left(\begin{bmatrix} \cos(\boldsymbol{e}_t[3]) & -\rho\sin(\boldsymbol{e}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{e}_t[3]) & \cos(\boldsymbol{e}_t[3]) \end{bmatrix} \mathbb{U} \right) = \mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3]). \tag{D1}$$

Note that multiplication of a matrix by a set that contains vector elements means that the mapping corresponding to that matrix is implemented on each vector element of the matrix. On the left-hand side of (D1), the Minkowski addition of the nominal control input and a linear transformation of the error set is given, whereas on the right-hand side a nonlinear transformation of the admissible set of control inputs, that is, $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$, should be computed. The resulting set $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$ does not necessarily correspond to a polytope. Moreover, since set $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$ is time-varying and based on (D1) is in general rotated per time instant, the exact computation of (D1) per time instant is complex or may become computationally intractable.

Picking an arbitrary element of $\mathbb{U}$, for example, $[u[1], u[2]]^T$, after the mapping $\begin{bmatrix} \cos(\boldsymbol{e}_t[3]) & -\rho\sin(\boldsymbol{e}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{e}_t[3]) & \cos(\boldsymbol{e}_t[3]) \end{bmatrix}$ is performed on this vector, we obtain a new vector $\left[\cos(\boldsymbol{e}_t[3])u[1] - \rho\sin(\boldsymbol{e}_t[3])u[2], \frac{1}{\rho}\sin(\boldsymbol{e}_t[3])u[1] + \cos(\boldsymbol{e}_t[3])u[2]\right]^T$, which belongs to the admissible set $\mathbb{U}$ of inputs in case based on (22), it satisfies the following condition:

$$\frac{|\cos(\boldsymbol{e}_t[3])u[1] - \rho\sin(\boldsymbol{e}_t[3])u[2]|}{\nu^{\max}} + \frac{|\sin(\boldsymbol{e}_t[3])u[1] + \rho\cos(\boldsymbol{e}_t[3])u[2]|}{\nu^{\max}} \le 1.$$

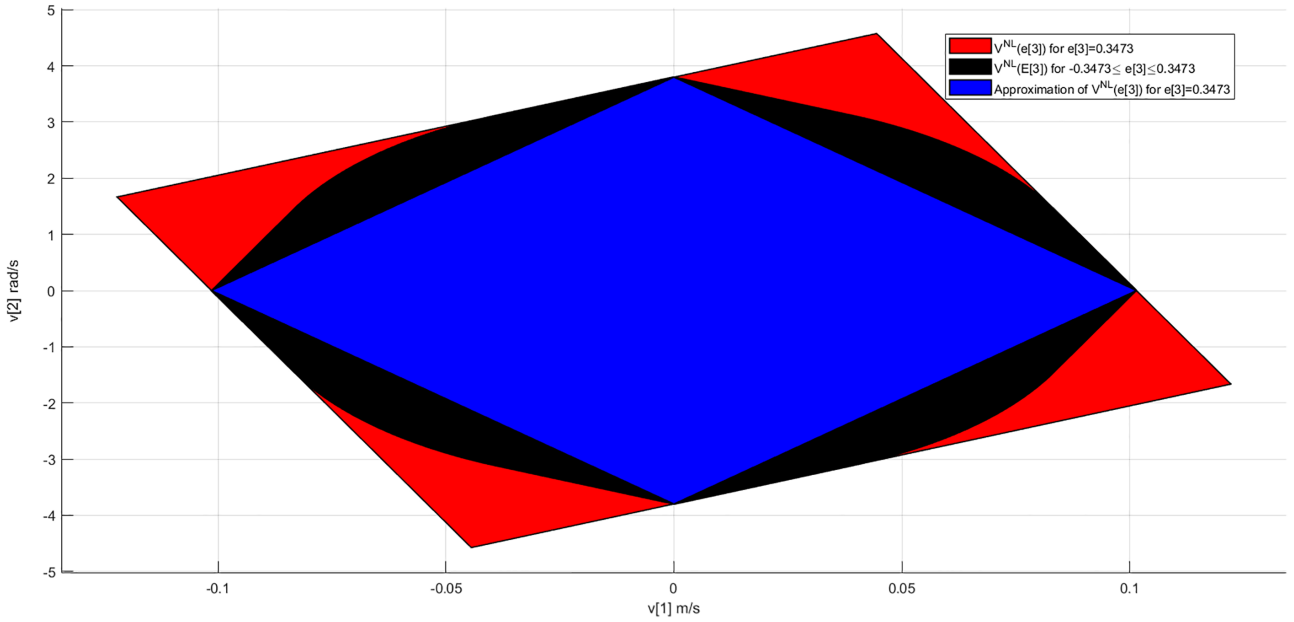

FIGURE D1 Illustration of an example quadrangle (the red background quadrangle) that represents $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$ when $\boldsymbol{e}_t[3] = 0.3473$. The intersection of all such quadrangles for the entire range of $\boldsymbol{e}_t[3]$, that is, for $-0.3473 \leq \boldsymbol{e}_t[3] \leq 0.3473$, generates $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$ (the black middle-ground polygon). The simplified version of $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$, that is, the largest subset of $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$ with its diameters parallel to the x and y axes, is shown via the blue foreground quadrangle.

Therefore, in general for each $\boldsymbol{e}_t[3] \in \mathbb{E}_t[3]$ the right-hand side term of (D1), that is, $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$, can be defined by the following quadrangle:

$$\begin{bmatrix} \frac{\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3])}{\nu^{\max}} & \frac{-\rho\sin(\boldsymbol{e}_t[3]) + \rho\cos(\boldsymbol{e}_t[3])}{\nu^{\max}} \\ \frac{-\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3])}{\nu^{\max}} & \frac{\rho\sin(\boldsymbol{e}_t[3]) + \rho\cos(\boldsymbol{e}_t[3])}{\nu^{\max}} \\ \frac{-\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3])}{\nu^{\max}} & \frac{\rho\sin(\boldsymbol{e}_t[3]) - \rho\cos(\boldsymbol{e}_t[3])}{\nu^{\max}} \\ \frac{\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3])}{\nu^{\max}} & \frac{-\rho\sin(\boldsymbol{e}_t[3]) - \rho\cos(\boldsymbol{e}_t[3])}{\nu^{\max}} \end{bmatrix} \boldsymbol{u}_t \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \tag{D2}$$

Figure D1 (see the red background quadrangle) illustrates an example of such a quadrangle, when $\boldsymbol{e}_t[3] = 0.3473$. The intersection of all such quadrangles for the entire range of $\boldsymbol{e}_t[3]$, in this case when $-0.3473 \leq \boldsymbol{e}_t[3] \leq 0.3473$, generates $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$ according to (D1). This set is illustrated via the black middle-ground polygon in Figure D1. In order to tackle the computational complexity associated with (D1) and to ensure that the resulting $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t[3])$ is always represented via a polytope, we replace the quadrangles corresponding to $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$ (i.e., the red background quadrangle in Figure D1) with the largest subset of $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$, for which the diameters are parallel to the x and y axes (see the blue foreground quadrangle in Figure D1). In other words, we replace $\mathbb{V}^{\mathrm{NL}}(\cdot)$ with the multiplication of the original set $\mathbb{U}$ and a positive, state-dependent scalar function $\lambda(\cdot)$, that is,

$$\mathbb{V}^{\mathrm{NL}}(\cdot) = \lambda(\cdot)\mathbb{U}. \tag{D3}$$

This approach is similar to the constraint tightening method used in Reference 40, although there a constant value is used instead of a scalar function $\lambda(\cdot)$.

To determine this subset, we first put $\boldsymbol{u}_t[1] = 0$, which based on (D2) results in:

$$\begin{bmatrix} \boldsymbol{u}_t[2] \\ \boldsymbol{u}_t[2] \\ -\boldsymbol{u}_t[2] \\ -\boldsymbol{u}_t[2] \end{bmatrix} \leq \begin{bmatrix} \frac{\nu^{\max}}{\rho(\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3]))} \\ \frac{\nu^{\max}}{\rho(\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3]))} \\ \frac{\nu^{\max}}{\rho(\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3]))} \\ \frac{\nu^{\max}}{\rho(\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3]))} \end{bmatrix}. \tag{D4}$$

Since $|\boldsymbol{e}_t[3]| < \pi/4$, both '$\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3])$' and '$\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3])$' result in positive values. Thus, (D4) can be reformulated via:

$$|\boldsymbol{u}_t[2]| \leq \frac{\nu^{\max}}{\rho} \lambda(\boldsymbol{e}_t[3]), \tag{D5}$$

where we define $\lambda(\cdot)$ via:

$$\lambda(\cdot) = \frac{1}{\cos(\cdot) + |\sin(\cdot)|}. \tag{D6}$$

Similarly, if we put $\boldsymbol{u}_t[2] = 0$, from (D2) we obtain:

$$\begin{bmatrix} \boldsymbol{u}_t[1] \\ \boldsymbol{u}_t[1] \\ -\boldsymbol{u}_t[1] \\ -\boldsymbol{u}_t[1] \end{bmatrix} \leq \begin{bmatrix} \frac{\nu^{\max}}{\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3])} \\ \frac{\nu^{\max}}{\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3])} \\ \frac{\nu^{\max}}{\cos(\boldsymbol{e}_t[3]) - \sin(\boldsymbol{e}_t[3])} \\ \frac{\nu^{\max}}{\cos(\boldsymbol{e}_t[3]) + \sin(\boldsymbol{e}_t[3])} \end{bmatrix}. \tag{D7}$$

which is equivalent to the following equation:

$$|\boldsymbol{u}_t[1]| \leq \nu^{\max} \lambda(\boldsymbol{e}_t[3]). \tag{D8}$$

From (D5) and (D8), vectors $\lambda(\boldsymbol{e}_t[3])\left[0, \frac{\nu^{\max}}{\rho}\right]^T$, $\lambda(\boldsymbol{e}_t[3])\left[0, -\frac{\nu^{\max}}{\rho}\right]^T$, $\lambda(\boldsymbol{e}_t[3])[\nu^{\max}, 0]^T$, and $\lambda(\boldsymbol{e}_t[3])[-\nu^{\max}, 0]^T$ are obtained as the corners of the polytope that represents the largest subset of $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3])$ (see, e.g., the blue foreground quadrangle in Figure D1). All other points of this polytope are obtained by a convex combination of these vectors. In this case, the original set $\mathbb{U}$ is simplified to a polytope with it corners given by $[\nu^{\max}, 0]^T$, $\left[0, -\frac{\nu^{\max}}{\rho}\right]^T$, $[-\nu^{\max}, 0]$, and $\left[0, \frac{\nu^{\max}}{\rho}\right]^T$, which satisfies $\mathbb{V}^{\mathrm{NL}}(\boldsymbol{e}_t[3]) = \lambda(\boldsymbol{e}_t[3])\mathbb{U}$, that is, satisfies (D3).

Figure D2 shows the curve corresponding to function $\lambda(\cdot)$. From the expression of $\lambda(\cdot)$ given by (D6) and as is shown in Figure D2, for negative values of $\boldsymbol{e}_t[3]$, $\lambda(\cdot)$ shows an ascending behavior, where the maximum of $\lambda(\cdot)$ is unity, which occurs for $\boldsymbol{e}_t[3] = 0$. For positive values of $\boldsymbol{e}_t[3]$, $\lambda(\cdot)$ shows a descending behavior. Moreover, $\lambda(\cdot)$ is an even function, which implies that its representative curve is symmetric with respect to the vertical axis (see Figure D2). Thus, for tightening the constraints online, from (D1) and (D3), we define the admissible set $\mathbb{V}(\mathbb{E}_t)$ for the nominal control inputs as a function of the error set via:

$$\mathbb{V}(\mathbb{E}_t) := \min_{\boldsymbol{e}_t[3] \in \mathbb{E}_t[3]} (\lambda(\boldsymbol{e}_t[3])\mathbb{U} \ominus \begin{bmatrix} -\cos(\boldsymbol{z}_t[3]) & -\sin(\boldsymbol{z}_t[3]) \\ \frac{1}{\rho}\sin(\boldsymbol{z}_t[3]) & -\frac{1}{\rho}\cos(\boldsymbol{z}_t[3]) \end{bmatrix} K^{\mathrm{e}}(\mathbb{E}_t[1] \times \mathbb{E}_t[2]). \tag{D9}$$

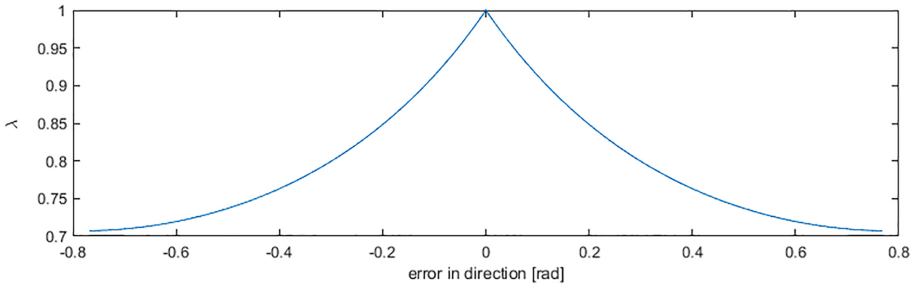

FIGURE D2 Illustration of the curve representing function $\lambda(\cdot)$ versus the error in the third dimension of the state space.

For the design of TMPC in Reference 40, the admissible set of the nominal control inputs were generated by scaling set $\mathbb{U}$ using a constant value 0.6636. As shown in Figure D2, set $\mathbb{V}(\mathbb{E}_t)$ defined via (D9) is consistently larger. Consequently, the original design is more conservative than our proposed approach. Note that in order to further reduce the conservatism, $\mathbb{V}^{\mathrm{NL}}(\mathbb{E}_t)$ and thus $\mathbb{V}(\mathbb{E}_t)$ may be approximated via polytopes that in general have more vertices than 4.