Proceedings of Machine Learning Research vol 242:968–979, 2024

System-level Safety Guard: Safe Tracking Control through Uncertain Neural Network Dynamics Models

Xiao Li HSIAOLI@UMICH.EDU
Yutong Li YUTLI@UMICH.EDU
Anouck Girard ANOUCK@UMICH.EDU
Ilya Kolmanovsky ILYA@UMICH.EDU
University of Michigan, Ann Arbor, MI, USA

Abstract

The Neural Network (NN), as a black-box function approximator, has been considered in many control and robotics applications. However, difficulties in verifying the overall system safety in the presence of uncertainties hinder the deployment of NN modules in safety-critical systems. In this paper, we leverage the NNs as predictive models for trajectory tracking of unknown dynamical systems. We consider controller design in the presence of both intrinsic uncertainty and uncertainties from other system modules. In this setting, we formulate the constrained trajectory tracking problem and show that it can be solved using Mixed-integer Linear Programming (MILP). The proposed MILP-based approach is empirically demonstrated in robot navigation and obstacle avoidance through simulations. The demonstration videos are available at `https://xiaolisean.github.io/publication/2023-11-01-L4DC2024`.

Keywords: neural networks, system-level safety, uncertainties, trajectory tracking

1. Introduction

Robotic and autonomous driving systems are typically structured as a pipeline of individual modules, which are designed separately to satisfy corresponding performance requirements and are verified at a system level. With recent advances in machine learning, NNs have been utilized in individual modules, e.g., localization (Li et al. (2021)), mapping (Roddick and Cipolla (2020)), and path planning (Barnes et al. (2017)). However, the NNs approximate the desired functionalities as nonlinear mappings from data, thereby introducing (intrinsic) approximation errors. On a system level, the performance of an NN module can also be affected by extrinsic uncertainties from other modules. In safety-critical scenarios, considering both intrinsic and extrinsic uncertainties is crucial to the module design, yet vital to securing safety at the system level.

Several methods have been proposed to employ NNs in control development with safety guarantees. In particular, constrained nonlinear optimal control problems have been pursued to control complex dynamical systems leveraging NN-learned dynamic models, e.g., control of quadrotor motion (Bansal et al. (2016)) and reduction of diesel engine emission (Zhang et al. (2023)). To mitigate the computation effort, methods based on MILP have been employed to embed NNs with ReLU activation functions in Model Predictive Control formulations (Wei and Liu (2021)). However, the notion of uncertainties has not been considered in the literature above. In this work, we consider trajectory tracking controller design leveraging NNs as predictive models (see Figure 1). Apart from NN prediction errors, we consider the presence of uncertainties from other modules, which affect the NN predictions and, subsequently, impact the controller design and the system's safety.

. This research is supported in part by AFOSR grant number FA9550-20-1-0385 and by National Science Foundation Award CMMI-1904394.

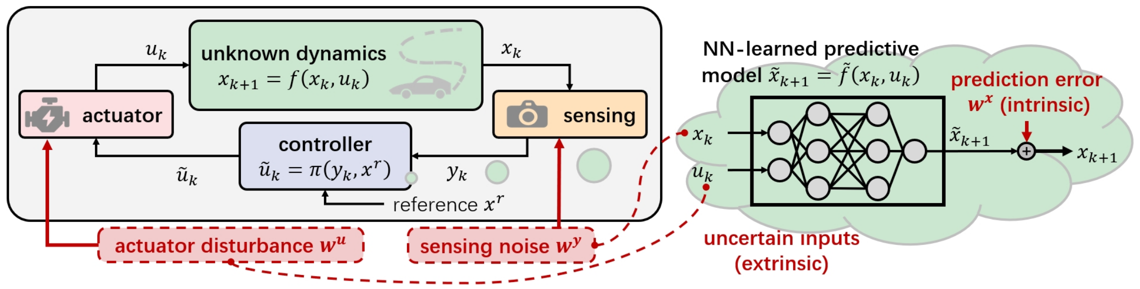
u_k
unknown dynamics
$x_{k+1} = f(x_k, u_k)$
x_k
actuator
sensing
controller
$\tilde{u}_k = \pi(y_k, x^r)$
$\tilde{u}_k$
y_k
reference x^r
actuator disturbance w^u
sensing noise w^y
NN-learned predictive model $\tilde{x}_{k+1} = \tilde{f}(x_k, u_k)$
prediction error w^x (intrinsic)
x_k
u_k
$\tilde{x}_{k+1}$
x_{k+1}
uncertain inputs (extrinsic)

Figure 1: Trajectory tracking through an NN-learned predictive model: The controller leverages an NN-learned model (with prediction errors) for predicting the unknown dynamics and tracks the reference trajectories. The NN predictions depend on the states x_k before the sensing module and the control u_k out of the actuator module, which are not directly accessible by the controller in this pipeline, and are uncertain quantities due to sensing noises and actuator disturbances.

In addition, several methods have been investigated for the safety verification of NNs, where safety constraints are imposed on NN outputs for a fixed set of inputs. The MILP-based methods have been utilized for evaluating NN classifiers with ReLU activation functions (Tjeng et al. (2017)). Approaches to reachability analysis based on Bernstein polynomials (Huang et al. (2019)) and Semi-Definite Programming (SDP) (Hu et al. (2020)) have been proposed to estimate an over-bound of the reachable set given an input domain and a NN representation. Differently from the existing literature, we employ NNs as predictive models for dynamical systems, and the NN inputs are affected by uncertainties that belong to a decision-variable-dependent set. This renders the activation status of NN neurons uncertain and dependent on the decision variables of the optimal control problem. We focus on exploiting the structure of NNs in accounting for uncertainty propagation.

The contributions of this paper are as follows: 1) We propose an approach to robust tracking controller design subject to system-level safety constraints, which leverages an NN-learned dynamic model of the system. The safety and dynamics constraints are informed by decision-variable-dependent uncertainty set propagation through NN and are handled using MILP. 2) We consider both intrinsic uncertainties from NN prediction errors and extrinsic uncertainties present in other modules and establish theoretical properties for the proposed method. 3) We illustrate the applications of the proposed approaches in simulations of collision-avoiding navigation for both an omnidirectional mobile robot and a conventional vehicle.

This paper is organized as follows: In Section 2, we introduce the assumptions on the actual dynamics of the system as well as the NN learned dynamics, and we formulate a robust tracking problem. In Section 3, we present our method to solve the robust tracking problem, using MILP and its theoretical properties. In Section 4, we use the proposed method, in combination with a Reachability-Guided RRT algorithm, to navigate an omnidirectional robot through a maze filled with obstacles. In Section 5, we leverage a set-theoretical localization algorithm that provides vehicle state measurements with uncertainty bounds, and we use our method to navigate a vehicle while avoiding collisions. Finally, conclusions are given in Section 6. Due to length constraints, the proofs of the theoretical properties are relegated to our full technical report (Li et al. (2023a)).

2. Problem Formulation

We consider a discrete-time dynamical system represented by $x_{k+1} = f(x_k, u_k)$, where the state $x_k \in \mathcal{X}$ evolves within a specified feasible set $\mathcal{X} \subset \mathbb{R}^{n_x}$; $u_k \in \mathcal{U}$ is the control and $\mathcal{U} \subset \mathbb{R}^{n_u}$ is the control admissible set; and $f : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is an unknown nonlinear mapping. In the sequel, we use lowercase letters, e.g., x, y, u, w, to represent vectors, capital letters, e.g., W, to define matrices, and scripted letters, e.g., $\mathcal{X}, \mathcal{U}$, to denote sets. With a slight abuse of notation, we use $a \leq b$ to denote the element-wise order between two vectors $a, b \in \mathbb{R}^n$. In addition, given a lower bound $a \in \mathbb{R}^n$, an upper bound $b \in \mathbb{R}^n$ and $a \leq b$, we use $[a, b]$ to denote a hypercube in $\mathbb{R}^n$. We use $I_{n\times n}$ to represent the identity matrix of size n. We use $\mathbb{1}_{n\times m}$, and $\mathbb{0}_{n\times m}$ to denote matrices with all zeros and ones, respectively, of size $n \times m$. We neglect the subscript $n \times m$ in calculations assuming that the dimensions are appropriate. We first discuss the NN-learned dynamic model for controller design that is subject to both intrinsic and extrinsic uncertainties in Section 2.1. Then, we introduce the robust tracking problem ensuring safety under uncertainties in Section 2.2.

2.1. Model Preliminaries

As shown in Figure 1, we approximate the dynamics f with a pre-trained $\ell-$layer fully connected neural network (NN) $\tilde{f} : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ that admits the following form

$$\begin{aligned} z_i = \sigma^{(i)}(\hat{z}_i), \quad \hat{z}_i = W^{(i)} z_{i-1} + b^{(i)}, \quad i = 1, \dots, \ell - 1, \\ z_0 = [x_k^T, u_k^T]^T, \quad \tilde{x}_{k+1} = W^{(\ell)} z_{\ell-1} + b^{(\ell)}, \end{aligned} \tag{1}$$

where $W^{(i)} \in \mathbb{R}^{n_i \times n_{i-1}}$, $b^{(i)} \in \mathbb{R}^{n_i}$ and $\sigma^{(i)} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ are the weight matrix, the bias vector and an element-wise nonlinear activation function in the ith layer, respectively. The NN uses the inputs x_k, u_k to compute a prediction $\tilde{x}_{k+1} = \tilde{f}(x_k, u_k)$ of the actual state x_{k+1}. We consider the overall system to be subject to uncertainties (see Figure 1) of three types: Firstly, the NN prediction $\tilde{x}_{k+1}$ of the actual state x_{k+1} is subject to an unknown prediction error w_k^x, i.e.,

$$\tilde{x}_{k+1} = x_{k+1} + w_k^x, \; w_k^x \in \mathcal{W}_x, \tag{2}$$

where $\mathcal{W}_x \subset \mathbb{R}^{n_x}$ is a bounded set; moreover, the actual states x_k and the actual control input u_k are both unknown to the controller due to the presence of noises in other modules. Secondly, we assume that a state measurement y_k of the actual state x_k is available from the sensing module and admits the form,

$$y_k = x_k + w_k^y, \; w_k^y \in \mathcal{W}_y, \tag{3}$$

where w_k^y is an unknown measurement noise, and $\mathcal{W}_y \subset \mathbb{R}^{n_x}$ is a bounded set. Thirdly, we consider the use of a feedback controller $\pi : \mathcal{X} \times \mathcal{X} \to \mathcal{U}$ to track the reference state x^r (to be designed). Then the actual control signal u_k is subject to an unknown additive actuator disturbance w_k^u, i.e.,

$$u_k = \tilde{u}_k + w_k^u, \; \tilde{u}_k = \pi(y_k, x^r), \; w_k^u \in \mathcal{W}_u, \tag{4}$$

where $\mathcal{W}_u \subset \mathbb{R}^{n_u}$ is a bounded set.

2.2. Robust Constrained Tracking Problem

Assume that $\mathcal{X}_u \subset \mathcal{X}$ is an unsafe set, e.g., representing obstacles, that the system should avoid. Given a reference $x^r \in \mathcal{X}_s$ in the safe subset, $\mathcal{X}_s = \mathcal{X} \backslash \mathcal{X}_u$, and a state measurement y_k that obeys the assumption in (3), our task is to design a controller $\pi(y_k, x^r)$ to track the reference x^r while keeping the system in the safe subset $\mathcal{X}_s$, i.e., ensuring $x_{k+1} \in \mathcal{X}_s$ given $x_k \in \mathcal{X}_s$. These objectives can be accounted for in a constrained optimization problem,

$$\pi(y_k, x^r) = \underset{\tilde{u}_k}{\operatorname{argmin}} \left(\max_{x \in \mathcal{F}\left(\mathcal{X}_k, \, \mathcal{U}_k(\tilde{u}_k)\right)} \|x - x^r\|_p \right) \tag{5a}$$

$$\text{subject to:} \quad \tilde{u}_k \in \mathcal{U}, \; \mathcal{X}_k = \left(y_k \oplus -\mathcal{W}_y\right) \cap \mathcal{X}_s, \; \mathcal{U}_k = \mathcal{U}_k(\tilde{u}_k) = (\tilde{u}_k \oplus \mathcal{W}_u) \cap \mathcal{U}, \tag{5b}$$

$$\tilde{\mathcal{F}}(\mathcal{X}_k,\mathcal{U}_k) = \left\{\tilde{x}_{k+1} \in \mathcal{X} : \ \tilde{x}_{k+1} = \tilde{f}(x_k,u_k), \forall x_k \in \mathcal{X}_k, \forall u_k \in \mathcal{U}_k\right\}, \tag{5c}$$

$$\mathcal{F}(\mathcal{X}_k,\mathcal{U}_k) = \tilde{\mathcal{F}}(\mathcal{X}_k,\mathcal{U}_k) \oplus \mathcal{W}_x, \ \mathcal{F}(\mathcal{X}_k,\mathcal{U}_k) \subset \mathcal{X}_s, \tag{5d}$$

where $\tilde{u}_k$ is the decision variable, $\|\cdot\|_p$ represents the ℓ_p norm, and $\oplus$ denotes the Minkowski sum. The term $y_k \oplus -\mathcal{W}_y$ stands for $\{y_k\} \oplus \{w : -w \in \mathcal{W}_y\}$ where we omit the curly brackets for simplicity. Given the measurement y_k that satisfies (3), the set $\mathcal{X}_k$ in constraint (5b) represents an uncertainty set that contains the actual state x_k. Similarly, the set $\mathcal{U}_k$ in constraint (5b) depends on the decision variable $\tilde{u}_k$ and contains the actual control u_k based on (4). The set-valued function $\tilde{\mathcal{F}} : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ in (5c) calculates the one step ahead reachable set of the learned NN dynamic model. The reachable set $\mathcal{F}(\mathcal{X}_k,\mathcal{U}_k)$ in (5d) is derived from $\tilde{\mathcal{F}}(\mathcal{X}_k,\mathcal{U}_k)$ and contains the actual state x_{k+1} based on (2). Meanwhile, the constraints in (5d) also guarantee the safety of the next state x_{k+1} for all the possible uncertainties modeled by (2), (3), (4). The objective defined by (5a) minimizes the maximum distance between the states in the reachable set $\mathcal{F}(\mathcal{X}_k,\mathcal{U}_k)$ and the reference, thereby steering the system closer to x^r. Unlike reachability analysis Hu et al. (2020), the input set $\mathcal{U}_k = \mathcal{U}_k(\tilde{u}_k)$ is conditioned on the decision variable $\tilde{u}_k$.

Remark *We introduce our methodology in the simplest possible setting, i.e., horizon-one Model Predictive Control with no control cost. This is the same setting as considered in Wei and Liu (2021). The methodology can be extended to accommodate a multi-step prediction horizon, by augmenting* (5c) *and* (5d) *with the following constraints:*

$$\mathcal{F}^{(i+1)} = \tilde{\mathcal{F}}^{(i+1)} \oplus \mathcal{W}_x \subset \mathcal{X}_s, \ \tilde{\mathcal{F}}^{(i+1)} = \tilde{\mathcal{F}}(\mathcal{F}^{(i)},\mathcal{U}_k), \ i = 1,2,\dots,N-1,$$

where N is the prediction horizon length, and $\mathcal{F}^{(1)} = \mathcal{F}(\mathcal{X}_k,\mathcal{U}_k)$. We consider $p = 1$ in the sequel as it leads to linear constraints and MILP.

3. Mixed-integer Linear Programming

To simplify the exposition of the approach, we assume that the uncertainty sets in (2), (3), (4) are hyper-cubes defined according to

$$\mathcal{W}_x = \{w^x \in \mathbb{R}^{n_x} : |w^x| \le \epsilon^x, \ \epsilon^x \in \mathbb{R}^{n_x}, \ \epsilon^x \ge 0\}, \tag{6a}$$

$$\mathcal{W}_y = \{w^y \in \mathbb{R}^{n_x} : |w^y| \le \epsilon^y, \ \epsilon^y \in \mathbb{R}^{n_x}, \ \epsilon^y \ge 0\}, \tag{6b}$$

$$\mathcal{W}_u = \{w^u \in \mathbb{R}^{n_u} : |w^u| \le \epsilon^u, \ \epsilon^u \in \mathbb{R}^{n_u}, \ \epsilon^u \ge 0\}. \tag{6c}$$

Note that, in principle, one can always find a hypercube overbounding a bounded set. Moreover, we assume that the nonlinear functions in the NN model are ReLU activation functions

$$\sigma^{(i)}(x) = \texttt{ReLU}(x) = \max\{0,x\}, \ i = 1,\cdots,\ell, \tag{7}$$

that are commonly adopted in contemporary NN architectures and have demonstrated good empirical performance (LeCun et al. (2015)). We, furthermore, focus on the case when the state feasible set, control admissible set, and unsafe set are represented as

$$\mathcal{X} = [\underline{x},\overline{x}], \ \underline{x},\overline{x} \in \mathbb{R}^{n_x}, \ \underline{x} \le \overline{x}, \tag{8a}$$

$$\mathcal{U} = [\underline{u},\overline{u}], \ \underline{u},\overline{u} \in \mathbb{R}^{n_u}, \ \underline{u} \le \overline{u}, \tag{8b}$$

$$\mathcal{X}_u = \cup_i \mathcal{X}_u^{(i)}, \ \mathcal{X}_u^{(i)} = [\underline{x}_u^{(i)},\overline{x}_u^{(i)}], \ \underline{x}_u^{(i)},\overline{x}_u^{(i)} \in \mathbb{R}^{n_x}, \ \underline{x} \le \underline{x}_u^{(i)} \le \overline{x}_u^{(i)} \le \overline{x}. \tag{8c}$$

The definition of the unsafe set (8c) enables us to tightly over-bound obstacles of irregular shapes using unions of hypercubes in the optimization problem. Then, equations (5b), (5c), (5d) encode the constraints associated with the input feasibility, NN structural non-linearity, and system safety, respectively, and are realized using integer decision variables as described in the respective Sections 3.1, 3.2, and 3.3. In this setting and when $p = 1$ in (5a), i.e., the cost function is based on the ℓ_1 norm, we show that (5) reduces to a MILP in Section 3.4. The proofs of the theoretical properties in this section are provided in our full technical report (Li et al. (2023a)).

3.1. Constraints Embedding Input Feasibility

Given a measurement y_k and the decision variable $\tilde{u}_k$, the constraints (5b) define the sets $\mathcal{X}_k$, $\mathcal{U}_k$ that are guaranteed to contain the actual state x_k and the control u_k while the decision variable shall be admissible, i.e., $x_k \in \mathcal{X}_k$, $u_k \in \mathcal{U}_k$, $\tilde{u}_k \in \mathcal{U}_k$. We embed these feasibility conditions of the NN inputs using linear inequality and equality constraints in the following proposition.

Proposition 1 *Given state measurement y_k and a decision variable $\tilde{u}_k$, assume that the unknown actual quantities x_k and u_k obey (3), (4) with assumptions in (6b), (6c), (8a), (8b). Let the decision variables $\tilde{u}_k \in \mathbb{R}^{n_u}$, $a_0, b_0 \in \mathbb{R}^{n_x+n_u}$, and $\delta^a, \delta^b \in \mathbb{R}^{n_u}$ satisfy the following constraints*

$$a_{0,1:n_x} = \max\{\underline{x}, y_k - \epsilon^y\}, \quad b_{0,1:n_x} = \min\{\overline{x}, y_k + \epsilon^y\}, \quad a_0 \le b_0, \tag{9}$$

$$\underline{u} \le \tilde{u}_k \le \overline{u}, \quad \delta^a_j, \delta^b_j \in \{0,1\},\ j = 1, \dots, n_u,$$
$$\left\{\begin{array}{l} a_{0,(n_x+1):(n_x+n_u)} \ge \underline{u} \\ a_{0,(n_x+1):(n_x+n_u)} \ge \tilde{u}_k - \epsilon^u \\ a_{0,(n_x+1):(n_x+n_u)} \le \underline{u} + M(\mathbb{1} - \delta^a) \\ a_{0,(n_x+1):(n_x+n_u)} \le \tilde{u}_k - \epsilon^u + M\delta^a \end{array}\right., \quad \left\{\begin{array}{l} b_{0,(n_x+1):(n_x+n_u)} \le \overline{u} \\ b_{0,(n_x+1):(n_x+n_u)} \le \tilde{u}_k + \epsilon^u \\ b_{0,(n_x+1):(n_x+n_u)} \ge \overline{u} - M(\mathbb{1} - \delta^b) \\ b_{0,(n_x+1):(n_x+n_u)} \ge \tilde{u}_k + \epsilon^u - M\delta^b \end{array}\right., \tag{10}$$

where δ^a_j denotes the jth element in column vector δ^a, $a_{0,m:n}$ represents the vector containing elements between row n and row m in a_0, the constant matrix $M = \mathrm{Diag}\left(\max\{\epsilon^u, \overline{u} - \underline{u} - \epsilon^u\}\right)$, and $\mathrm{Diag}(x) \in \mathbb{R}^{n\times n}$ yields a square matrix with elements of x on the diagonal and zero anywhere else. Then, it is guaranteed that $z_0 = [x_k^T\ u_k^T]^T \in [a_0,\ b_0]$.

The constraints in Proposition 1 enforce input feasibility, i.e., $\tilde{u}_k \in \mathcal{U}_k$, and provide bounds a_0, b_0 on the NN input subject to extrinsic uncertainties, i.e., $a_0 \le [x^T,\ u^T]^T \le b_0$ for all $x \in \mathcal{X}_k,\ u \in \mathcal{U}_k$. Specifically, based on assumption (6b) and (8a), the constraints (9) imply $\mathcal{X}_k \subseteq [a_{0,1:n_x}, b_{0,1:n_x}]$. Based on assumption (6c) and (8b), the constraints (10) are equivalent to the following inequalities

$$\max\{\underline{u}, \tilde{u}_k - \epsilon^u\} = a_{0,(n_x+1)\cdots(n_x+n_u)} \le b_{0,(n_x+1)\cdots(n_x+n_u)} = \min\{\overline{u}, \tilde{u}_k + \epsilon^u\},$$

thereby $\mathcal{U}_k \subseteq [a_{0,(n_x+1):(n_x+n_u)}, b_{0,(n_x+1):(n_x+n_u)}]$. We use integer variables δ^a, δ^b to move the decision variable $\tilde{u}_k$ out of the nonlinear min/max function: $\delta^a_j = 1$ implies the jth element of $\max\{\underline{u}, \tilde{u}_k - \epsilon^u\}$ attains the value of the jth element of $\underline{u}$, otherwise $\delta^a_j = 0$; $\delta^b_j = 1$ indicates the jth element of $\min\{\overline{u}, \tilde{u}_k + \epsilon^u\}$ attains the value of the jth element of $\overline{u}$, otherwise $\delta^b_j = 0$.

3.2. Constraints Encoding NN Structural Non-Linearity

Using the bounded sets $\mathcal{X}$ in (8a) and $\mathcal{U}$ in (8b), we can numerically derive lower and upper bounds $\underline{\hat{z}}_i, \overline{\hat{z}}_i \in \mathbb{R}^{n_i}, i = 1, \dots, \ell$ on the neuron values $\hat{z}_i$ and the output $\tilde{x}_{k+1}$ using interval arithmetic, i.e., $\hat{z}_i \in [\underline{\hat{z}}_i, \overline{\hat{z}}_i]$ and $\tilde{x}_{k+1} \in [\underline{\hat{z}}_\ell, \overline{\hat{z}}_\ell]$. In the sequel, these derived bounds are used to tighten the constraints and limit the search region for the optimization solver. Given the decision variables a_0, b_0 in Proposition 1 as bounds on the NN input, we can encode the decision-variable-dependent uncertainty set propagation through the NN defined in (5c) using the following results:

Proposition 2 *Given $z_0 = [x_k^T \ u_k^T]^T \in [a_0, \ b_0]$, consider a NN defined by (1) and (7), and let the decision variables $a_{i-1}, b_{i-1} \in \mathbb{R}^{n_{i-1}}$, $\hat{a}_i, \hat{b}_i \in \mathbb{R}^{n_i}$, $\delta_i^{--}, \delta_i^{-+}, \delta_i^{++} \in \{0,1\}^{n_i}$, $i = 1, \ldots, \ell-1$, and $a_{k+1}, b_{k+1} \in \mathbb{R}^{n_x}$ satisfy the following constraints*

$$\begin{gathered} \hat{a}_{i,j} = w_j^{(i)} S_i\left((w_j^{(i)})^T\right) \begin{bmatrix} a_{i-1} \\ b_{i-1} \end{bmatrix} + b_j^{(i)}, \quad \hat{b}_{i,j} = w_j^{(i)} S_i\left((w_j^{(i)})^T\right) \begin{bmatrix} b_{i-1} \\ a_{i-1} \end{bmatrix} + b_j^{(i)}, \\ \underline{\hat{z}}_i \leq \hat{a}_i \leq \hat{b}_i \leq \overline{\hat{z}}_i, \quad \forall j = 1, \cdots, n_i, \quad \forall i = 1, \ldots, \ell-1, \end{gathered} \tag{11}$$

$$\begin{gathered} \begin{cases} a_i \geq \hat{a}_i \\ a_i \leq \hat{a}_i - \mathrm{Diag}(\underline{\hat{z}}_i)(\delta_i^{--} + \delta_i^{-+}) \\ a_i \leq \mathrm{Diag}(\overline{\hat{z}}_i)\delta_i^{++} \end{cases}, \quad \begin{cases} b_i \geq \hat{b}_i \\ b_i \leq \hat{b}_i - \mathrm{Diag}(\underline{\hat{z}}_i)\delta_i^{--} \\ b_i \leq \mathrm{Diag}(\overline{\hat{z}}_i)(\delta_i^{-+} + \delta_i^{++}) \end{cases}, \\ 0 \leq a_i \leq b_i, \quad \delta_{i,j}^{--}, \delta_{i,j}^{-+}, \delta_{i,j}^{++} \in \{0,1\}, \quad \delta_{i,j}^{--} + \delta_{i,j}^{-+} + \delta_{i,j}^{++} = 1, \\ \forall j = 1, \cdots, n_i, \quad \forall i = 1, \ldots, \ell-1, \end{gathered} \tag{12}$$

$$\begin{gathered} \underline{\hat{z}}_\ell \leq a_{k+1} \leq b_{k+1} \leq \overline{\hat{z}}_\ell, \quad a_{k+1,j} = w_j^{(\ell)} S_\ell\left((w_j^{(\ell)})^T\right) \begin{bmatrix} a_{\ell-1} \\ b_{\ell-1} \end{bmatrix} + b_j^{(\ell)}, \\ b_{k+1,j} = w_j^{(\ell)} S_\ell\left((w_j^{(\ell)})^T\right) \begin{bmatrix} b_{\ell-1} \\ a_{\ell-1} \end{bmatrix} + b_j^{(\ell)}, \quad \forall j = 1, \cdots, n_i, \end{gathered} \tag{13}$$

where $\hat{a}_{i,j}$ is the jth element of $\hat{a}_i$; $w_j^{(i)}$ is the jth row of $W^{(i)}$; $b_j^{(i)}$ is the jth element of $b^{(i)}$; $\delta_{i,j}^{--}$ denotes the jth element of δ_i^{--}; $a_{k+1,j}, b_{k+1,j}$ represent the jth element of a_{k+1}, b_{k+1}, respectively; The functions $S_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_{i-1} \times 2n_{i-1}}$ and $s : \mathbb{R} \to \mathbb{R}^{2n_{i-1}}$ are defined according to

$$S_i\left(\begin{bmatrix} \vdots \\ w_q \\ \vdots \end{bmatrix}\right) = \begin{bmatrix} \vdots \\ (s(w_q))^T \\ \vdots \end{bmatrix}, \quad s(w_q) := \begin{cases} \begin{bmatrix} e_q \\ \mathbb{0}_{n_{i-1} \times 1} \end{bmatrix} & \text{if } w_q \geq 0 \\ \begin{bmatrix} \mathbb{0}_{n_{i-1} \times 1} \\ e_q \end{bmatrix} & \text{if } w_q < 0 \end{cases},$$

and $e_q \in \mathbb{R}^{n_{i-1}}$ ($q = 1, \ldots, n_{i-1}$) has 1 as the qth element and other elements being zero. Then, the reachable set $\tilde{\mathcal{F}}(\mathcal{X}_k, \mathcal{U}_k)$ defined in (5c) is a subset of the hypercube $[a_{k+1}, b_{k+1}]$, i.e., $\tilde{\mathcal{F}}(\mathcal{X}_k, \mathcal{U}_k) \subseteq [a_{k+1}, b_{k+1}]$.

From the $(i-1)$th to ith layer of the NN, the uncertainty set propagation is realized through variables a_{i-1}, b_{i-1} and $\hat{a}_i, \hat{b}_i$ that are the lower and upper bounds of z_{i-1} and $\hat{z}_i$, respectively (i.e., $z_{i-1} \in [a_{i-1}, b_{i-1}]$ and $\hat{z}_i \in [\hat{a}_i, \hat{b}_i]$). The constraints (11) and (13) encode the uncertainty propagation through the fully connected layers, $\hat{z}_i = W^{(i)} z_{i-1} + b^{(i)}$ and $\tilde{x}_{k+1} = W^{(\ell)} z_{\ell-1} + b^{(\ell)}$, respectively. The uncertainty set propagation through the nonlinear ReLU function is enforced in constraints (12). In constraints (11) and (13), the qth row of matrix $S_i((w_j^{(i)})^T)$ switches the upper and lower bounds of z_{i-1} if the qth element in $w_j^{(i)}$ is negative. Meanwhile, since the values z_{i-1} in neurons fall into a bounded hypercube $[a_{i-1}, b_{i-1}]$, the activation status of each ReLU activation function is uncertain. Inspired by Tjeng et al. (2017), we introduce integer variables $\delta_i^{--}, \delta_i^{-+}, \delta_i^{++}$ in constraints (12) to encode the uncertainty in the ReLU activation status according to

$$\hat{a}_{i,j} \leq \hat{b}_{i,j} \leq 0 \text{ if } \delta_{i,j}^{--} = 1; \quad \hat{a}_{i,j} \leq 0 \leq \hat{b}_{i,j} \text{ if } \delta_{i,j}^{-+} = 1; \quad 0 \leq \hat{a}_{i,j} \leq \hat{b}_{i,j} \text{ if } \delta_{i,j}^{++} = 1.$$

Furthermore, as can be shown, the hypercube defined by lower bound a_{k+1} and upper bound b_{k+1} is an over-estimation of the actual reachable set $\tilde{\mathcal{F}}$, i.e., $\tilde{\mathcal{F}} \subseteq [a_{k+1}, b_{k+1}]$.

3.3. Constraints Enforcing System Safety

Given a hypercube $[a_{k+1}, b_{k+1}]$ as an over-estimation of the reachable set $\tilde{\mathcal{F}}$ in Proposition 2, we enforce the safety constrains (5d) of the system such that $x_{k+1} \in \mathcal{F}(\mathcal{X}_k, \mathcal{U}_k) \subset \mathcal{X}_s$. We consider the presence of a simple unsafe subset $\mathcal{X}_u = [\underline{x}_u, \overline{x}_u]$ in the following result and discuss the extension to a union of $\mathcal{X}_u^{(i)}$ defined in (8c) at the end of this section.

Proposition 3 *Given $\tilde{\mathcal{F}}(\mathcal{X}_k, \mathcal{U}_k) \subseteq [a_{k+1}, b_{k+1}]$ and state space defined in (8a), assume that the NN predictions are subject to bounded additive errors defined in (2) and (6a), and the decision variables $\underline{x}_{k+1}, \overline{x}_{k+1} \in \mathbb{R}^{n_x}$, $\delta_1^u, \delta_2^u \in \mathbb{R}^{n_x}$ satisfy the following constraints:*

$$\underline{x} \leq \underline{x}_{k+1} \leq \overline{x}_{k+1} \leq \overline{x}, \quad \underline{x}_{k+1} = a_{k+1} - \epsilon^x, \quad \overline{x}_{k+1} = b_{k+1} + \epsilon^x, \tag{14}$$

$$\begin{cases} \overline{x}_{k+1} \leq \overline{x} + \mathrm{Diag}\left(\underline{x}_u - \overline{x}\right) \delta_1^u \\ \overline{x}_{k+1} \geq \underline{x}_u - \mathrm{Diag}\left(\underline{x}_u - \underline{x}\right) \delta_1^u \\ \underline{x}_{k+1} \geq \underline{x} + \mathrm{Diag}\left(\overline{x}_u - \underline{x}\right) \delta_2^u \\ \underline{x}_{k+1} \leq \overline{x}_u - \mathrm{Diag}\left(\overline{x}_u - \overline{x}\right) \delta_2^u \end{cases}, \qquad \begin{array}{c} \delta_{1,j}^u, \delta_{2,j}^u \in \{0,1\}, \quad \forall j = 1, \dots, n_x, \\ \delta_{1,j}^u + \delta_{2,j}^u \leq 1, \quad \forall j = 1, \dots, n_x, \\ \sum_{j=1}^{n_x} \left(\delta_{1,j}^u + \delta_{2,j}^u\right) \geq 1, \end{array} \tag{15}$$

where $\delta_{1,j}^u$, $\delta_{2,j}^u$ are the jth element of δ_1^u, δ_2^u, respectively. Then, $x_{k+1} \in \mathcal{F}(\mathcal{X}_k, \mathcal{U}_k)$ where $\mathcal{F}(\mathcal{X}_k, \mathcal{U}_k)$ is the reachable set defined in (5d), and $\mathcal{F}(\mathcal{X}_k, \mathcal{U}_k) \subseteq [\underline{x}_{k+1}, \overline{x}_{k+1}] \subset \mathcal{X}_s$ with $\mathcal{X}_u = [\underline{x}_u, \overline{x}_u]$.

In Proposition 3, the constraints (14) are equivalent to $[\underline{x}_{k+1}, \overline{x}_{k+1}] = [a_{k+1}, b_{k+1}] \oplus \mathcal{W}_x$ and $[\underline{x}_{k+1}, \overline{x}_{k+1}] \subset \mathcal{X}$. Considering the result $\tilde{\mathcal{F}}(\mathcal{X}_k, \mathcal{U}_k) \subseteq [a_{k+1}, b_{k+1}]$ from Proposition 2, it is obvious that $\mathcal{F}(\mathcal{X}_k, \mathcal{U}_k) \subseteq [\underline{x}_{k+1}, \overline{x}_{k+1}]$ based on the definition of $\mathcal{F}(\mathcal{X}_k, \mathcal{U}_k)$ in (5d). Subsequently, the safety constraint of $\mathcal{F}(\mathcal{X}_k, \mathcal{U}_k) \subset \mathcal{X}_s$ can be enforced with $[\underline{x}_{k+1}, \overline{x}_{k+1}] \subset \mathcal{X}_s$, which is equivalent to $[\underline{x}_{k+1}, \overline{x}_{k+1}] \subset \mathcal{X}$ and $[\underline{x}_{k+1}, \overline{x}_{k+1}] \cap \mathcal{X}_u = \varnothing$. The constraints (15) enforce $[\underline{x}_{k+1}, \overline{x}_{k+1}] \cap \mathcal{X}_u = \varnothing$ using integer variables according to

$$\begin{cases} \underline{x}_{k+1,j} \leq \overline{x}_{k+1,j} \leq \underline{x}_{u,j} & \text{if } \delta_{1,j}^u = 1,\ \delta_{2,j}^u = 0 \\ \overline{x}_{k+1,j} \geq \underline{x}_{k+1,j} \geq \overline{x}_{u,j} & \text{if } \delta_{1,j}^u = 0,\ \delta_{2,j}^u = 1 \\ \underline{x}_{u,j} \leq \overline{x}_{k+1,j} \leq \overline{x},\ \underline{x} \leq \underline{x}_{k+1,j} \leq \overline{x}_{u,j} & \text{if } \delta_{1,j}^u = 0,\ \delta_{2,j}^u = 0 \end{cases},$$

where $\overline{x}_{k+1,j}, \underline{x}_{k+1,j}, \overline{x}_{u,j}, \underline{x}_{u,j}$ are the jth element of $\overline{x}_{k+1}, \underline{x}_{k+1}, \overline{x}_u, \underline{x}_u$, respectively. The integer constraints imply that there exists at least one dimension j where $[\underline{x}_{k+1,j}, \overline{x}_{k+1,j}] \cap [\underline{x}_{u,j}, \overline{x}_{u,j}] = \varnothing$. Subsequently, the hypercube $[\underline{x}_{k+1}, \overline{x}_{k+1}]$ has zero overlaps with the unsafe subset $[\underline{x}_u, \overline{x}_u]$. Notably, in the case of a complex unsafe region $\mathcal{X}_u$, we can derive a union of hypercubes $\cup_i \mathcal{X}_u^{(i)}$ as in (8c) that over-bounds $\mathcal{X}_u$. Thereafter, to ensure safety, we can formulate similar constraints (15) with each individual hypercube $\mathcal{X}_u^{(i)}$ in the union.

3.4. Safe Tracking Control using MILP

The optimization objective in (5a) is designed to minimize the maximum distance between the points in the reachable set and the reference state x^r. Focusing on the case when $p = 1$, i.e., the cost is defined using ℓ_1 norm, we can introduce a vector of slack variables $\lambda \in \mathbb{R}^{n_x}$, and reformulate (5a) as

$$\begin{aligned} &\operatorname{argmin}_{\tilde{u}_k} \textstyle\sum_{q=1}^{n_x} \lambda_q, \\ &\text{subject to:} \quad \lambda \geq 0,\ -\lambda \leq \underline{x}_{k+1} - x^r \leq \lambda,\ -\lambda \leq \overline{x}_{k+1} - x^r \leq \lambda, \end{aligned} \tag{16}$$

where λ_q designates the qth element of vector λ. This objective function relies on the fact that the maximum distance, between a reference x^r and points in the hypercube $[\underline{x}_{k+1}, \overline{x}_{k+1}]$, is attained at the points located at the boundary of the hypercube. Then, the optimization problem (5) for tracking the reference state x^r safely can be rewritten into a MILP according to

Robust Constrained Tracking Control Problem:

$$\underset{\substack{\tilde{u}_k,\, \delta^a,\, \delta^b,\, a_0,\, b_0,\, a_{k+1},\, b_{k+1},\, \underline{x}_{k+1},\, \overline{x}_{k+1},\, \delta_1^u,\, \delta_2^u,\, \lambda,\\ a_i,\, b_i,\, \hat{a}_i,\, \hat{b}_i,\, \delta_i^{--},\, \delta_i^{-+},\, \delta_i^{++},\, i=1,\ldots,\ell-1,}}{\operatorname{argmin}} \sum_{q=1}^{n_x} \lambda_q, \quad \text{subject to: (9), (10), (11), (12), (13), (14), (15), (16).} \tag{17}$$

We note that the number of decision variables in the MILP problem (17) scales linearly with the number of neurons in the NN. In the subsequent examples, we use the *YALMIP* toolbox (Lofberg (2004)) for MATLAB to solve the optimization (17). The code is available at `https://github.com/XiaoLiSean/MILPSafetyGuard`. It also has the following property.

Proposition 4 *Consider a NN-learned dynamical system defined by (1), (7) that takes control u_k and state x_k as inputs and yields $\tilde{x}_{k+1}$ as a prediction of the next state x_{k+1} and satisfies the bounded additive error assumption in (2), (6a). We assume that $x_k \in \mathcal{X}_s$ is unknown but belongs to a safe set $\mathcal{X}_s = \mathcal{X} \backslash \mathcal{X}_u$ that is the complement set of the unsafe region $\mathcal{X}_u$ given by (8c) in the state space $\mathcal{X}$ defined by (8a). Also assume that a measurement y_k of x_k is given that satisfies the assumptions in (3), (6b). If there exists a solution of the Problem (17) such that the corresponding $\tilde{u}_k$ is in the admissible set $\mathcal{U}$ defined by (8b), then for all actuator disturbances w_k^u in set $\mathcal{W}_u$ defined by (6c), the actual control u_k subject to this additive disturbance w_k^u according to (4) renders the actual next system state safe, i.e., $x_{k+1} \in \mathcal{X}_s$.*

4. Obstacle Avoidance and Reachability-Guided RRT

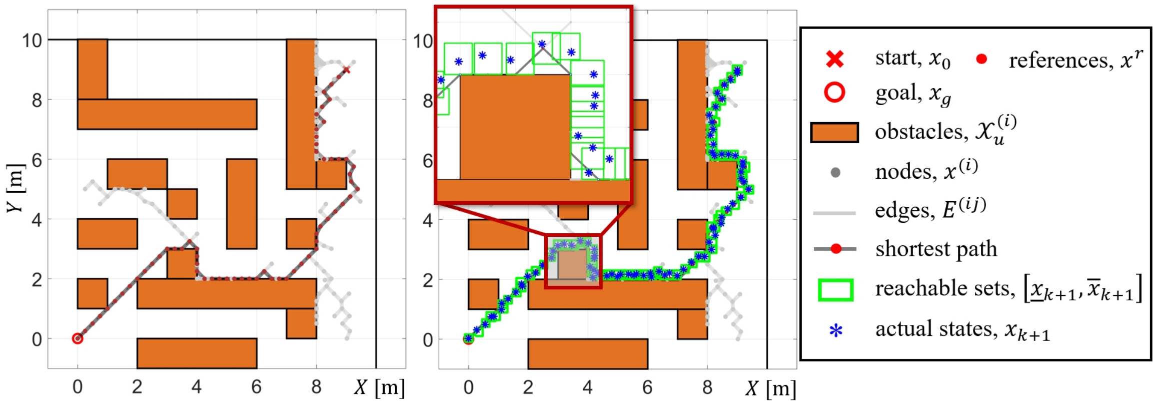

Figure 2: Schematic of obstacle avoidance using an omnidirectional robot: (Left) The reachability-guided RRT algorithm expands the tree from the start x_0 to the goal x_g over the safe state space $\mathcal{X}_s$. Then, we use the Dijkstra planning algorithm to find a reference path (black lines with red dots) from x_0 to x_g that has the shortest distance defined by ℓ_1 norm. (Middle) We use the proposed method in (17) to track the reference states (red dots). Our method can guarantee that the robot motion is collision-free under uncertainties, i.e., the unknown actual states in blue asterisks are in the safe set $\mathcal{X}_s$, even with most of the reference points located near the obstacles.

We consider an omnidirectional robot as a point mass and use the following equation to represent its kinematics: $x_{k+1} = f(x_k, u_k) = x_k + u_k + w_k$, where $x_k \in \mathbb{R}^2$ is the robot coordinates in the $X - Y$ plane, $u_k \in \mathbb{R}^2$ contains the displacements, and the values of the disturbance $w_k \sim U(-\epsilon^x, \epsilon^x)$ are sampled uniformly from the interval $[-\epsilon^x, \epsilon^x]$. We consider the sets $\mathcal{U} = \{[u_1\ u_2]^T \in \mathbb{R}^2 : -0.25 \le u_1, u_2 \le -0.25\}$, $\mathcal{X} = \{[x_1\ x_2]^T \in \mathbb{R}^2 : -1 \le x_1, x_2 \le 10\}$ together with obstacles $\mathcal{X}_u^{(i)}$ visualized as orange boxes in Figure 2. The NN is manually constructed and admits the following form

$$\tilde{x}_{k+1} = \begin{bmatrix} -I_{2\times 2} & -I_{2\times 2} \end{bmatrix} \texttt{ReLU}\left(\begin{bmatrix} I_{2\times 2} & \mathbb{0}_{2\times 2} \\ \mathbb{0}_{2\times 2} & I_{2\times 2} \end{bmatrix}\begin{bmatrix} x_k \\ u_k \end{bmatrix} + \begin{bmatrix} -50\cdot \mathbb{1}_{2\times 1} \\ -50\cdot \mathbb{1}_{2\times 1} \end{bmatrix}\right) + 100\cdot \mathbb{1}_{2\times 1},$$

which is equivalent to $\tilde{x}_{k+1} = \tilde{f}(x_k, u_k) = x_k + u_k$ given $x_k \in \mathcal{X}$ and $u_k \in \mathcal{U}$. Finally, the uncertainty bounds are set to $\epsilon^x = \epsilon^u = \epsilon^y = [0.05,\ 0.05]^T$. We develop a reachability-guided RRT planner similar to Shkolnik et al. (2009) using the CORA toolbox (Althoff (2015)), combined with the Dijkstra algorithm, to generate a path of reference states x^r. Different from the classic RRT, the reachability-guided RRT incorporates dynamics as a constraint to extend the edges of the tree. This tree $\mathcal{T}(\{x^{(i)}\}, \{E^{(ij)}\})$ comprises nodes $x^{(i)} \in \mathcal{X}_s$ and directed edges $E^{(ij)}$. The edge $E^{(ij)}$ connects node $x^{(i)}$ to node $x^{(j)}$ and implies that there exists a control $u_k \in \mathcal{U}$ such that $x^{(j)} = \tilde{f}(x^{(i)}, u_k) \in \mathcal{X}_s$, i.e., $x^{(j)} \in \tilde{\mathcal{F}}(\{x^{(i)}\}, \mathcal{U}) \cap \mathcal{X}_s$. As shown in Figure 2, the algorithm is initialized with an initial node x_0 and is terminated if there exists a node $x^{(i)} \in \mathcal{T}$ such that $x_g \in \tilde{\mathcal{F}}(\{x^{(i)}\}, \mathcal{U})$. At each time step, we take the first node $x^{(i)}$ in the shortest path as the reference state x^r in (17). We solve the optimization (17) and apply the resulting control $\tilde{u}_k$ to the actual dynamic model f. Then, we remove $x^{(i)}$ from the path if $x^{(i)} \in [\underline{x}_{k+1}, \overline{x}_{k+1}]$ and the navigation terminates when $x_g \in [\underline{x}_{k+1}, \overline{x}_{k+1}]$. The solution of (17) takes 0.16 ± 0.08 sec on a laptop with an Intel i7-8550U CPU and 8 GB memory. As shown in Figure 2, after taking control $\tilde{u}_k$, the resulting next states x_{k+1} in blue asterisks are within the reachable set $[\underline{x}_{k+1}, \overline{x}_{k+1}]$ in green boxes and the boxes $[\underline{x}_{k+1}, \overline{x}_{k+1}]$ are collision-free, which empirically validates Proposition 4.

5. Vehicle Navigation and Set-theoretical Localization

We next consider a front-wheel drive vehicle of 2 m width shown in Figure 3. The length of the vehicle wheelbase is $l = 5$ m. We adopt the vehicle kinematics model from Li et al. (2023b), which admits the form,

$$x_{k+1} = f(x_k, u_k) = \begin{bmatrix} p_{x,k} + v_k dt \cos\theta_k \cos\delta_k \\ p_{x,k} + v_k dt \sin\theta_k \cos\delta_k \\ \theta_k + v_k dt/l \sin\delta_k \end{bmatrix},$$

where $x_k = [p_{x,k}\ p_{y,k}\ \theta_k]^T$ is the state vector, $(p_{x,k},\ p_{y,k})$ in meters are the coordinates of the center of the vehicle rear wheel axis, and $\theta_k \in [-\pi, \pi]$ is the vehicle orientation; $u_k = [v_k\ \delta_k]^T$ is the control vector, v_k in m/s is the vehicle longitudinal speed, and δ_k in rad is the vehicle steering angle; $dt = 0.1$ sec is the sampling period. We use the sets $\mathcal{U} = \{u_k : v_k \in [2, 5],\ \delta_k \in [-0.6, 0.6]\}$, $\mathcal{X} = \{x_k : p_{x,k}, p_{y,k} \in [-50, 50],\ \theta_k \in [-\pi, \pi]\}$ together with obstacles $\mathcal{X}_u^{(i)}$ visualized in grey boxes in Figure 3. For the uncertainties, we assume the actuator disturbance $\epsilon^u = [0.01\ \mathrm{m/s},\ 0.5\ \mathrm{deg}]^{\mathrm{T}}$. We densely sample a dataset $\mathcal{D} = \{(x_k^{(i)}, u_k^{(i)}, x_{k+1}^{(i)})\}_i$ from $\mathcal{X} \times \mathcal{U}$ for NN training and quantifying the NN prediction error. The NN has two hidden layers of 8 and 4 neurons, respectively. Based on Pytorch library (Paszke et al. (2019)), we train an NN $\tilde{f}$ using the Stochastic Gradient Descent algorithm and dataset $\mathcal{D}$ to minimize the mean-squared error $\left\| x_{k+1}^{(i)} - \tilde{f}(x_k^{(i)}, u_k^{(i)}) \right\|_2^2$, and the prediction error is equal to $\epsilon^x = [0.02\ \mathrm{m},\ 0.02\ \mathrm{m},\ 1.5\ \mathrm{deg}]^{\mathrm{T}}$. We quantify the prediction error as the maximum value of the empirical absolute error, i.e.,

$$\epsilon^x = \max\left\{\epsilon \in \mathbb{R}^3 : \epsilon = \left| x_{k+1}^{(i)} - \tilde{f}(x_k^{(i)}, u_k^{(i)}) \right|,\ (x_k^{(i)}, u_k^{(i)}, x_{k+1}^{(i)}) \in \mathcal{D}\right\},$$

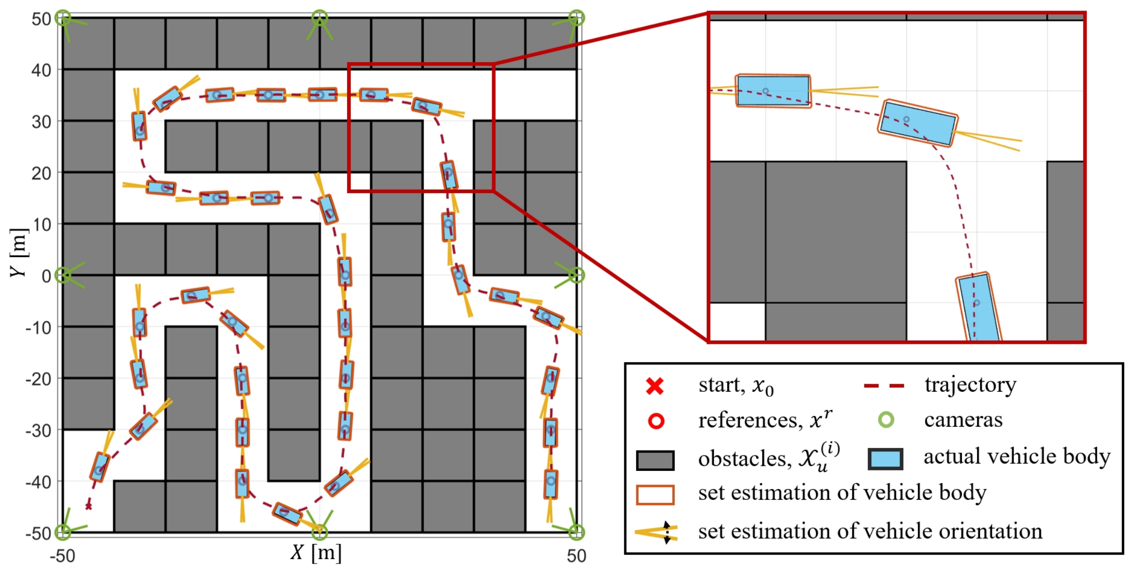

Figure 3: Schematic of navigating a vehicle through a maze with tracking controller (17) and with a set-theoretic localization algorithm. The zoom-in view, at the top-left, demonstrates that the set-theoretic localization algorithm provides estimates of the vehicle body and orientation that are guaranteed to contain the actual ones. The tracking controller (17) leverages this information, together with an NN-learned vehicle dynamics model, to avoid obstacles.

and the function $\max$ is applied element-wise. The theoretical properties of uncertainty-bound quantification from samples are discussed in Dean et al. (2020) and are beyond the scope of this work. At each time step k, we apply the set-theoretic localization algorithm presented in Li et al. (2023b) to generate an uncertainty polygon P_{xy} that contains the actual vehicle position $[p_{x,k},\ p_{y,k}]^T$ and an uncertainty interval P_θ that contains the actual vehicle orientation θ_k, i.e., $[p_{x,k},\ p_{y,k}]^T \in P_{xy}$ and $\theta_k \in P_\theta$. Subsequently, we can derive the smallest hypercube P that over-bounds $P_{xy} \times P_\theta$, and the measurement error ϵ^y is quantified as half of the sizes of P. The algorithm can also produce a polytope estimation of the vehicle body as demonstrated in Figure 3. Then, akin to the process in Section 4, we solve the optimization problem (17) and use the resulting control to navigate the vehicle through the maze. The solution of (17) takes 0.14 ± 0.06 sec. As shown in Figure 3, we can also observe that the vehicle motion is collision-free, which is consistent with Proposition 4.

6. Conclusion and Future Work

In this paper, we developed an approach for robust reference tracking that leveraged a learned NN model to control the actual dynamics. We considered both bounded intrinsic and extrinsic uncertainties from the controller and other system modules, respectively. We transcripted the resulting decision-variable-dependent uncertainty set propagation through NN using a MILP. We provided results which ensure that the proposed MILP can render the overall system safe considering all possible actuator disturbances, measurement noise, and prediction errors within their corresponding bounded sets. We tested the proposed method in navigation and obstacle avoidance scenarios for an omnidirectional robot and a vehicle in simulations. We also note that the recursive feasibility is not guaranteed under the current horizon-one MPC setting; this will be addressed in future work.

References

Matthias Althoff. An introduction to cora 2015. In *Proc. of the workshop on applied verification for continuous and hybrid systems*, pages 120–151, 2015.

Somil Bansal, Anayo K Akametalu, Frank J Jiang, Forrest Laine, and Claire J Tomlin. Learning quadrotor dynamics using neural network for flight control. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 4653–4660. IEEE, 2016.

Dan Barnes, Will Maddern, and Ingmar Posner. Find your own way: Weakly-supervised segmentation of path proposals for urban autonomy. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 203–210. IEEE, 2017.

Sarah Dean, Nikolai Matni, Benjamin Recht, and Vickie Ye. Robust guarantees for perception-based control. In *Learning for Dynamics and Control*, pages 350–360. PMLR, 2020.

Haimin Hu, Mahyar Fazlyab, Manfred Morari, and George J Pappas. Reach-sdp: Reachability analysis of closed-loop systems with neural network controllers via semidefinite programming. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 5929–5934. IEEE, 2020.

Chao Huang, Jiameng Fan, Wenchao Li, Xin Chen, and Qi Zhu. Reachnn: Reachability analysis of neural-network controlled systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(5s):1–22, 2019.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Xiao Li, Yidong Du, Zhen Zeng, and Odest Chadwicke Jenkins. Seannet: Semantic understanding network for localization under object dynamics. *arXiv preprint arXiv:2110.02276*, 2021.

Xiao Li, Yutong Li, Anouck Girard, and Ilya Kolmanovsky. System-level safety guard: Safe tracking control through uncertain neural network dynamics models. *arXiv preprint arXiv:2312.06810*, 2023a.

Xiao Li, Yutong Li, Nan Li, Anouck Girard, and Ilya Kolmanovsky. Set-theoretic localization for mobile robots with infrastructure-based sensing. *Advanced Control for Applications: Engineering and Industrial Systems*, 5(1):e117, 2023b.

Johan Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In *2004 IEEE international conference on robotics and automation (IEEE Cat. No. 04CH37508)*, pages 284–289. IEEE, 2004.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11138–11147, 2020.

Alexander Shkolnik, Matthew Walter, and Russ Tedrake. Reachability-guided sampling for planning under differential constraints. In *2009 IEEE International Conference on Robotics and Automation*, pages 2859–2865. IEEE, 2009.

Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Tianhao Wei and Changliu Liu. Safe control with neural network dynamic models. *arXiv preprint arXiv:2110.01110*, 2021.

Jiadi Zhang, Xiao Li, Ilya Kolmanovsky, Munechika Tsutsumi, and Hayato Nakada. Model predictive control of diesel engine emissions based on neural network modeling. *arXiv preprint arXiv:2311.03555*, 2023.