Online Learning for Safe Model Predictive Control with the Compatible Models Approach ⋆

Anas Makdesi, Antoine Girard, Laurent Fribourg

Online Learning for Safe Model Predictive Control with the Compatible Models Approach ⋆

Anas Makdesi * **Antoine Girard** * **Laurent Fribourg** **

* *Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes, 91190, Gif-sur-Yvette, France* {*anas.makdesi,antoine.girard*}*@l2s.centralesupelec.fr*
** *Université Paris-Saclay, CNRS, ENS Paris-Saclay, Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France e-mail: fribourg@lsv.fr*

Abstract: In this paper, we introduce an online safe learning-based Model Predictive Control (MPC) approach. This approach, which we call the "compatible model approach", consists of building two models of the system. The first is a piecewise interval-valued over-approximation of the system, and the second is a single-valued piecewise multi-affine estimation of the system's dynamics. The first model is used to find the set of safe actions at each state, whereas the second is used to choose -out of those safe actions- the input that minimizes a given cost function. For the first model, we use the -assumed known- bounds on the derivative of the dynamics to update the model. The second model should be contained in the first to ensure the feasibility of the MPC scheme everywhere (Hence, the name compatible). Both models can be updated online. We are able to do that because each new transition updates the models locally. We present a test case where we train a mobile robot at low speeds, then navigate it in an environment while avoiding obstacles and collecting new data to learn its dynamics at high speeds.

Keywords: Bounded derivative systems, data-driven control, symbolic control, abstraction, online learning.

1. INTRODUCTION

In the rapidly evolving field of control theory, data-driven approaches are increasingly favored due to their adeptness at managing complex nonlinear dynamics, which are often challenging to model or completely unknown. However, a significant impediment to their broader adoption, particularly in safety-critical applications, is the lack of guaranteed safety. In order to facilitate their broader adoption, considerable efforts have been dedicated to ensuring the safety of these approaches, particularly in safety-critical applications (Hewing et al., 2020, and references therein). This paper presents our contribution to this ongoing endeavor: a learning-based Model Predictive Control (MPC) scheme that comes with guaranteed safety. A significant body of work has been dedicated to this endeavor. For instance, (Berberich et al. (2020)) and (Zhang et al. (2022)) have made notable contributions towards enhancing safety and robustness in learning-based MPC. Similar to this work, the study by (Aswani et al. (2013)) proposed the use of two models to represent the system, one for ensuring safety and the other for minimizing a cost function, with the latter being updated online. We also use two models, but with different structures, and we do not impose any linearity assumptions on the system.

In this paper, we introduce a safe learning-based MPC scheme that can be updated online. The scheme consists of building two models of the system. The first is a piecewise interval-valued over-approximation of the system, and the second is a single-valued piecewise multi-affine estimation of the system's dynamics. This estimation is compatible with the over-approximation, meaning that it is contained in the over-approximation. The first model is used to find the set of safe actions at each state, whereas the second is used to choose the input that minimizes a given cost function. To build the first model, we use the bounds on the derivatives of the dynamics to find -from data- a piecewise interval-valued over-approximation. By construction, the over-approximation contains the true unknown dynamics. This model is not but a step to finding a finite-state representation of the system (Abstraction). With this data-driven abstraction, we can use discrete control synthesis techniques to find control action fulfilling various complex requirements (Tabuada (2009)) and (Belta et al. (2017)). Essential to our case is the safety requirement. By implementing a fixed point algorithm, we can automatically reach the set of safe actions at each state of the system. In recent years, much work has been done to find data-driven abstractions. In (Kazemi et al. (2022)), abstractions are computed using the growth bound of the system calculated from a finite number of trajectories. Trajectories of unknown systems are collected in (Lavaei and Frazzoli (2022)), and a scenario optimization program is used to construct an alternating bisimulation function with probabilistic guarantees. Also, in (Lavaei et al. (2023)), stochastic bisimulation functions are used to capture the distance between trajectories of an unknown stochastic system and those of a finite Markov decision process. In (Devonport et al. (2021)), guarantees for the learned abstraction were provided using the Probably Approximately Correct (PAC) statistical framework. What distinguishes the introduced over-approximation model in this work from the

⋆ This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 725144).

previously mentioned works is that it guarantees to contain the system's dynamics, whereas the previous works offer probabilistic guarantees. In (Sadraddini and Belta (2018)), robust over-approximations are calculated using the system's Lipschitz constant, but the approach can only deal with a small number of points.

After finding the set of safe actions at each state, we use the single-valued estimation to choose the best control action out of the safe ones that minimize a given cost function. In previous work (Makdesi et al. (2023b)), we build a data-driven two-model approach to find a safe learning-based MPC controller. The two models were calculated offline after sampling a set of data from the system. This work introduces models that can be updated online while the system is working. To do that, we use the new data collected from the system to update the models locally. Other than the ability to update the models online, the introduced models are different from the ones in (Makdesi et al. (2023b)) in how we find the over-approximation and the compatible estimation. Instead of using the bounds on the derivatives of the dynamics to transform the data into a monotone-like case, we find the interval hull of the growth cones built from the data points and the bounds on the derivatives. Those growth cones were used by (Milanese and Novara (2004)) for the identification of nonlinear systems, and by Canale et al. (2014) to find data-driven MPC controllers. Because we find the interval-valued over-approximation on predefined partitions of the input and state spaces, we can deal with way more data points than (Milanese and Novara (2004)) and (Canale et al. (2014)). To find locally-computed compatible estimations, we implement a stochastic gradient descent-like algorithm.

The paper is organized as follows. In Section 2, we formulate the problem we are trying to solve. Section 3 deals with finding the system's over-approximation and using it to find the set of safe inputs. In Section 4, we introduce the safe learning-based scheme and the class of functions we use to estimate the true unknown dynamics. We showcase the validity of our approach with a test case about a mobile robot in Section 5.

Notations

Given two vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n$, we define the partial order $\preceq$ on $\mathbb{R}^n$ to be $\mathbf{z}_1 \preceq \mathbf{z}_2$ if and only if $z_1^i \leq z_2^i$ for all $i = 1, \ldots, n$. $[\mathbf{z}_1, \mathbf{z}_2] = \{\mathbf{z} \in \mathbb{R}^n | \ \mathbf{z}_1 \preceq \mathbf{z} \preceq \mathbf{x}_2\}$ defines a closed interval of $\mathbb{R}^n$. We define $\max(\mathbf{z}_1, \mathbf{z}_2)$, or $\min(\mathbf{z}_1, \mathbf{z}_2)$, to be the vector $\mathbf{z}$ whose components are $z^i = \max(z_1^i, z_2^i)$, or $z^i = \min(z_1^i, z_2^i)$ respectively. Minkowski addition is denoted by $\oplus$ and defined as follows: $A \oplus B = \{\mathbf{a} + \mathbf{b} \mid \mathbf{a} \in A, \mathbf{b} \in B\}$. Meanwhile, the Minkowski subtraction is denoted by $\ominus$ and defined as follows: $A \ominus B = (A^c \oplus (-B)^c)^c$ where A^c is the complement of A and $-B$ is the set of vectors $-\mathbf{b}$ for all $\mathbf{b} \in B$.

2. PROBLEM FORMULATION

Given $X \subseteq \mathbb{R}^{n_x}$, $U \subseteq \mathbb{R}^{n_u}$, $D \subseteq \mathbb{R}^{n_x}$, let us consider a discrete-time nonlinear system of the form:

$$\mathbf{x}(t+1) = f(\mathbf{x}(t), \mathbf{u}(t)) + \mathbf{d}(t) \tag{1}$$

where $\mathbf{x} \in X, \mathbf{u} \in U, \mathbf{d} \in D$ are the state, input, and disturbance. $f : X \times U \to X$ is an unknown nonlinear function. The unknown function f has bounded derivatives, and the set of disturbances $D = [\underline{\mathbf{d}}, \overline{\mathbf{d}}]$, is a bounded interval with known bounds $\underline{\mathbf{d}}, \overline{\mathbf{d}} \in \mathbb{R}^{n_x}$ and such that $0 \in W$.

Let us assume we are given a set of data generated from the dynamic system (1):

$$\mathscr{D} = \{(\mathbf{x}_k, \mathbf{u}_k, \mathbf{x}'_k) \mid \mathbf{x}'_k \in f(\mathbf{x}_k, \mathbf{u}_k) + D, k \in \mathbb{K}\}$$

where $\mathbb{K}$ is a finite set of indices. There are different ways to collect the data set $\mathscr{D}$. One way is to randomly sample the system's dynamics using independent samples. This can be done easily if one can use a black box model of (1) to generate independent simulations. However, our approach does not require the use of independent samples. Instead, we can collect the data by recording the evolution of the true system over a given period. In that case, we would have $\mathbf{x}'_k = \mathbf{x}_{k+1}$.

In (Makdesi et al. (2023b)), a learning-based MPC scheme was introduced. The scheme depends on building two models of the system offline, and it is capable of enforcing a strong safety requirement such that the system's state always stays safe $\mathbf{x}(t) \in X_s, \forall t \in \mathbb{N}$, where $X_s \subseteq X$ is a safe set.

First, a set of safe inputs is found using a data-driven over-approximation of the system's dynamics

Definition 1. An *over-approximation* of the dynamics defined in (1) is a set-valued map $F : X \times U \rightrightarrows X$ that satisfies

$$f(\mathbf{x}, \mathbf{u}) + D \subseteq F(\mathbf{x}, \mathbf{u}), \forall \mathbf{x} \in X, \forall \mathbf{u} \in U. \tag{2}$$

This set-valued over-approximation is used to find a safety controller C_F using either set-theoretic methods (Blanchini and Miani (2008)) or symbolic control (Tabuada (2009)).

Definition 2. A *safety controller* C_F for the safe set X_s and the map F is a set-valued map $C_F : X \rightrightarrows U$ satisfying

- $\mathrm{dom}(C_F) \subseteq X_s$,
- $\forall \mathbf{x} \in \mathrm{dom}(C_F), \forall \mathbf{u} \in C_F(\mathbf{x}), F(\mathbf{x}, \mathbf{u}) \subseteq \mathrm{dom}(C_F)$,

where $\mathrm{dom}(C_F) = \{\mathbf{x} \in X \mid C_F(\mathbf{x}) \neq \emptyset\}$ is the domain of C_F.

We use safety controllers to attribute a set of allowed inputs to each state $\mathbf{x} \in \mathrm{dom}(C_F)$. Then, a single-valued estimation of the true function is built to find the input that minimizes a receding horizon cost function out of all the possible safe inputs.

Definition 3. An estimation $\hat{f} : X \times U \to X$ of the true function f is said to be *compatible* with the over-approximation F if

$$\hat{f}(\mathbf{x}, \mathbf{u}) + D \subseteq F(\mathbf{x}, \mathbf{u}), \forall \mathbf{x} \in X, \forall \mathbf{u} \in U. \tag{3}$$

The following theorem shows how to use the set of safe inputs found using the data-driven over-approximation and a compatible estimation to implement a learning-based MPC program that meets the strict safety requirements while enforcing a more relaxed performance requirement.

Theorem 1. Makdesi et al. (2023b) Given a stage costs $J_k : X \times U \to \mathbb{R}$, $k \in \{1, \ldots, N-1\}$ and a terminal cost $J_N : X \to \mathbb{R}$, starting from $\mathbf{x}(0) \in \mathrm{dom}(C_F)$, consider the trajectory of (1) with $\mathbf{u}(t) = \mathbf{u}(0|t)$ where $\mathbf{u}(0|t)$ is obtained by solving the optimisation problem below:

$$\begin{aligned} \min_{\mathbf{u}(0|t), \ldots \mathbf{u}(N-1|t)} \ & \sum_{i=0}^{N-1} J_i(\mathbf{x}(i|t), \mathbf{u}(i|t)) + J_N(\mathbf{x}(N|t)) \\ \text{subject to} \quad & \mathbf{x}(i+1|t) = \hat{f}(\mathbf{x}(i|t), \mathbf{u}(i|t)), \\ & \qquad \forall i \in \{0, \ldots, N-1\} \\ & \mathbf{x}(i|t) \in X_s, \ \forall i \in \{0, \ldots, N\} \\ & \mathbf{u}(0|t) \in C_F(\mathbf{x}(t)) \\ & \mathbf{x}(0|t) = \mathbf{x}(t) \end{aligned} \tag{4}$$

Then, for all $t \in \mathbb{N}$, $\mathbf{x}(t) \in X_s$ and (4) admits a feasible solution, i.e. the closed-loop system is safe and well-posed.

In this paper, we are interested in finding a safe learning-based MPC scheme that can be updated online. We assume that we are given a set of data $\mathscr{D}$ sampled from the system (1). We also assume that we are given the bounds on the derivatives of the dynamics and the bounds on the disturbance. We want to find approaches that can be used to build the over-approximation and the compatible estimation offline to be used in implementing a safe learning-based MPC scheme according to Theorem 1. When the system is working, we want to be able to update the models online using the new data collected from the system to enlarge the set of safe inputs and improve the performance of the controller.

3. DATA-DRIVEN SAFETY CONTROLLERS

In this chapter, we will introduce an algorithm to build an over-approximation of an unknown function. The over-approximation we introduce is a piecewise interval-valued map defined on a predefined partition of the inputs and states spaces. This piecewise interval-valued map is well suited for building a finite-state representation of the system and, subsequently, a safety controller.

Let $Z \subseteq \mathbb{R}^{n_z}$, $Y \subseteq \mathbb{R}^{n_y}$, $D \subseteq \mathbb{R}^{n_y}$ and $f : Z \to Y$ is defined as

$$\mathbf{y} = f(\mathbf{z}) + \mathbf{d} \tag{5}$$

where $\mathbf{z} \in Z, \mathbf{y} \in Y$, and $\mathbf{d} \in D = [\underline{\mathbf{d}}, \overline{\mathbf{d}}]$. The unknown function f has bounded derivatives, i.e.

$$\frac{\partial f^i}{\partial z^j}(\mathbf{z}) \in [\underline{\gamma}_{ij}, \overline{\gamma}_{ij}], i \in \{1,\dots,n_y\},\ j \in \{1,\dots,n_z\}.$$

It is straightforward to implement the results of finding the over-approximation of the function in (5) to study systems defined in (1) by choosing $Z = X \times U$ and $\mathbf{z} = (\mathbf{x}, \mathbf{u})$. A set of data is sampled from the unknown function $\mathscr{D} = \{(\mathbf{z}_k, \mathbf{y}_k) \mid \mathbf{y}_k \in f(\mathbf{z}_k) + D, k \in \mathbb{K}\}$

Remark 1. In (Makdesi et al. (2023a)), a data set consisting of i.i.d. samples is utilized to estimate the bounds on the derivatives of the function f and the bounds on the disturbance D with probabilistic guarantees.

3.1 Building the over-approximation

In this section, we will introduce an algorithm to build an over-approximation of the function f defined in (5).

Let us start by defining the partition of the input space Z. For simplicity, we assume that $Z = [\underline{\mathbf{z}}, \overline{\mathbf{z}}]$. For each coordinate $i \in \{1,\dots,n_z\}$, let be given finite partition $(Z_{r^i})_{r^i \in R^i}$ of the interval $[\underline{z}^i, \overline{z}^i]$ where $R^i = \{0,\dots,K^i\}$ and

$$\begin{cases} z^i_0 &= [\underline{z}^i, \zeta^i_1), \\ z^i_{r^i} &= [\zeta^i_{r^i}, \zeta^i_{r^i+1}),\ r^i = 1,\dots,K^i-1, \\ z^i_{K^i} &= [\zeta^i_{K^i}, \overline{z}^i], \end{cases}$$

where $\underline{z}^i < \zeta^i_1 < \cdots < \zeta^i_{K^i} < \overline{z}^i$. We define $R = R^1 \times \cdots \times R^{n_z}$, and let the finite rectangular partition $(Z_{\mathbf{r}})_{\mathbf{r} \in R}$ of Z be given for $\mathbf{r} = (r^1,\dots,r^{n_z})$ by $Z_{\mathbf{r}} = Z^1_{r^1} \times \cdots \times Z^{n_z}_{r^{n_z}}$. We denote by $\underline{\mathbf{z}}_{\mathbf{r}}, \overline{\mathbf{z}}_{\mathbf{r}}$ the lower and upper bounds of the intervals $Z_{\mathbf{r}}$. When talking about the states and inputs separately, we will use $(X_{\mathbf{q}})_{\mathbf{q} \in Q}, (U_{\mathbf{p}})_{\mathbf{p} \in P}$ for the partition of the states and inputs spaces.

We also consider a quantization function $\phi : Z \to R$ associated to the finite partitions $(Z_{\mathbf{r}})_{\mathbf{r} \in R}$ and defined as

$$\forall \mathbf{z} \in Z\ \forall \mathbf{r} \in R,\ \phi(\mathbf{z}) = \mathbf{r} \iff \mathbf{z} \in Z_{\mathbf{r}}. \tag{6}$$

The quantization function ϕ only goal is to aggregate the input space points inside a set $Z_{\mathbf{r}}$ to a single symbol $\mathbf{r}$. This will allow us to define a finite-state representation of the unknown function.

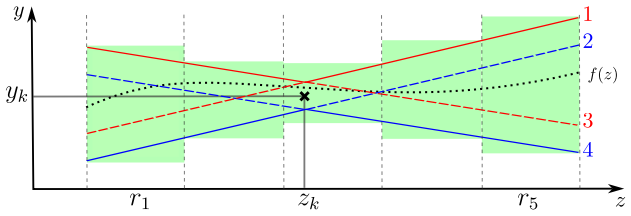


Fig. 1. The figure shows the over-approximation of f on the cells $Z_{\mathbf{r}}$. The over-approximation is done by finding the value of the growth cone at the vertices of the cells. Lines 1 through 4 are found using the bounds on the derivatives of the function f and the bounds on the disturbance D.

Let $V_{\mathbf{r}}$ be the set of vertices of the interval $Z_{\mathbf{r}}$, we define the interval-valued over-approximation map $F : Z \rightrightarrows \mathbb{R}^{n_y}$ to be a map $F(\mathbf{r}) = [\underline{F}(\mathbf{r}), \overline{F}(\mathbf{r})]$, where the upper and lower values are defined by their components $\underline{F}^i, \overline{F}^i$, $i \in \{1,\dots,n_y\}$ as follows

$$\overline{F}^i(\mathbf{r}) = \min_{k \in \mathbb{K}} \left(\max_{\mathbf{v} \in V_{\mathbf{r}}} \left(y^i_k + \overline{d}^i - \underline{d}^i + \overline{\mathbf{m}}_i{}^T (\mathbf{z}_k, \mathbf{v}) \cdot (\mathbf{v} - \mathbf{z}_k) \right) \right) \tag{7}$$

$$\underline{F}^i(\mathbf{r}) = \max_{k \in \mathbb{K}} \left(\min_{\mathbf{v} \in V_{\mathbf{r}}} \left(y^i_k + \underline{d}^i - \overline{d}^i + \underline{\mathbf{m}}_i{}^T (\mathbf{z}_k, \mathbf{v}) \cdot (\mathbf{v} - \mathbf{z}_k) \right) \right) \tag{8}$$

$$\overline{m}^j_i = \begin{cases} \underline{\gamma}_{ij} & \text{if } z^j_k > v^j, \\ \overline{\gamma}_{ij} & \text{if } z^j_k < v^j. \end{cases} \quad \underline{m}^j_i = \begin{cases} \overline{\gamma}_{ij} & \text{if } z^j_k > v^j, \\ \underline{\gamma}_{ij} & \text{if } z^j_k < v^j. \end{cases}$$

Proposition 1. The map $F \circ \phi : Z \rightrightarrows \mathbb{R}^{n_y}$ over-approximate the unknown function (5).

The computation of the over-approximation can be done one data point at a time, making it suitable for online computation. The main limitation of applying this method online is the partition size. If the partition is too large, the computation of the over-approximation will be too slow. To overcome this limitation, we introduce next a locally updated over-approximation that offers a trade-off between the speed of computation and the conservatism of the over-approximation.

Let $R_w(\mathbf{r}) = \{\mathbf{r}' \in R \mid max(\mathbf{r} - \underline{\mathbf{r}}_w, \mathbf{0}_n) \preceq \mathbf{r}' \preceq \min(\mathbf{r} + \overline{\mathbf{r}}_w, K), \underline{\mathbf{r}}_w, \overline{\mathbf{r}}_w \in R\}$ define the window where we want to update the over-approximation. We define the locally-computed over-approximation $F_w : Z \rightrightarrows \mathbb{R}^{n_y}$ as follows: For all $i \in \{1,\dots,n_y\}$

$$\overline{F}^i_w(\mathbf{r}) = \min_{k \in \mathscr{K}(\mathbf{r})} \left(\max_{\mathbf{v}_l \in V_{\mathbf{r}}} \left(y^j_k + \overline{d}^j - \underline{d}^j + \overline{\mathbf{m}}_i \cdot (\mathbf{z}_k - \mathbf{v}_l) \right) \right) \tag{9}$$

$$\underline{F}^i_w(\mathbf{r}) = \max_{k \in \mathscr{K}(\mathbf{r})} \left(\max_{\mathbf{v}_l \in V_{\mathbf{r}}} \left(y^j_k + \underline{d}^j - \overline{d}^j + \underline{\mathbf{m}}_i \cdot (\mathbf{z}_k - \mathbf{v}_l) \right) \right) \tag{10}$$

$$\mathscr{K}(\mathbf{r}) = \{k \in \mathbb{K} \mid \mathbf{z}_k \in Z_{\mathbf{r}^\star}, \mathbf{r}^\star \in R_w(\mathbf{r})\}$$

Proposition 2. The map $F_w \circ \phi : Z \rightrightarrows \mathbb{R}^{n_x}$ over-approximate the system (1)

3.2 The safety controller

As mentioned earlier, the over-approximation map is used to build a finite-state model. This model is built using the predefined partition and the calculated over-approximation. We begin by defining a tool to describe the finite-state model.

Definition 4. A *transition system* T is a tuple $T = (X, U, \Delta, Y, H)$, where X is a set of states, U is a set of inputs, $\Delta : X \times U \rightrightarrows$

X is a transition relation, Y is a set of outputs, and $H : X \to Y$ is an output map.

Transition systems are useful for describing the behavior of both continuous and discrete systems, for example, we define transition systems $T_{sys} = (X, U, \Delta_{sys}, Y, H)$ associated to (1) where the set of states X and inputs U are the same as in (1). The transition relation Δ_{sys} is defined for all $\mathbf{x} \in X$, for all $\mathbf{u} \in U$ by $\Delta_{sys}(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}, \mathbf{u}) + D$. The set of outputs Y, and the output map H can be defined in different ways depending on the application. We define a symbolic transition system $T_{symb} = (Q, P, \Delta_{symb}, Y, H_{symb})$ where the set of states and inputs are given by the partitions index sets Q and P, and the transition relation Δ_{symb} is defined as follows, for all $\mathbf{q} \in Q$, for all $\mathbf{p} \in P$

$$\Delta_{symb}(\mathbf{q}, \mathbf{p}) = \left\{ \mathbf{q}' \in Q | \ (F_w(\mathbf{q}, \mathbf{p})) \cap X_{\mathbf{q}'} \neq \emptyset \right\}. \tag{11}$$

We define the set of outputs to be $Y = Q$ and the output map H_{symb} to be the identity map. In (Makdesi et al. (2023a)), it is shown that a symbolic transition system T_{symb} (the finite-state model) built using an interval-valued over-approximation like F_w and quantized like in (11) alternatingly simulates the true system, or in other words, it is an abstraction of the transition system T_{sys}.

Proposition 3. (see (Makdesi et al., 2023a, Theorem 3)).
Let T_{sys} be defined for the set of outputs $Y = Q$ and the output map $H : X \to Q$, given by

$$\forall \mathbf{x} \in X, \ \forall \mathbf{q} \in Q, \ H(\mathbf{x}) = \mathbf{q} \iff \mathbf{x} \in X_{\mathbf{q}}. \tag{12}$$

Then, $T_{symb} \preceq_{AS} T_{sys}$.

This means that any controller synthesized for the finite-state model can be refined to work on the true system. Most relevant to our work is safety; we can find the safety controller $C_{\Delta_{symb}} : Q \rightrightarrows P$ and the control invariant set X_{inv} for the finite-state model using a fixed point algorithm by iteratively removing unsafe actions. Although this type of algorithm can automatically find the safety controller, it is meant to be carried out offline as it is computationally prohibitive to do online. Algorithm 1 shows how to update the control invariant set and the safety controller. This is done by checking all the cells in the neighboring window of the data point and adding the cells that become safe to the control invariant set. All the safe actions in the neighboring cells are also added to the safety controller. Algorithm 1 will not be able to find all the safe actions and cells based on the new data point. To do that, we need to go through all the cells and actions in the partition. However, this algorithm is computationally less expensive than the fixed-point algorithm and can be done online.

Algorithm 1: Update the control invariant set and the safety controller

Input : F_w, X_{inv}, $C_{\Delta_{symb}}$
Output: X_{inv}, $C_{\Delta_{symb}}$
foreach $\mathbf{q} \in Q_w$ **do**
  foreach $\mathbf{p} \in P_w$ **do**
    if $F_w(\mathbf{q}, \mathbf{p}) \subseteq X_{inv}$ **then**
      $X_{inv} \leftarrow X_{inv} \cup \mathbf{q}$;
      $C_{\Delta_{symb}}(\mathbf{q}) \leftarrow C_F(\mathbf{q}) \cup \mathbf{p}$;
    end
  end
end

4. SAFE LEARNING-BASED MPC

The previous section introduced a data-driven approach to synthesizing safety controllers. This section shows how to build a single-valued estimation of dynamics. First, let us define the type of estimation we will work with.

4.1 Piecewise multi-affine functions

We use the class of piecewise multi-affine functions to build compatible estimations.

Definition 5. A multi-affine function $g : Z \to \mathbb{R}^m$, $Z \subseteq \mathbb{R}^n$ is a function of the form

$$g(z^1, \ldots, z^n) = \sum_{i_1, \ldots, i_n \in \{0,1\}} c_{i_1, \ldots, i_n} (z^1)^{i_1} \ldots (z^n)^{i_n} \tag{13}$$

where $c_{i_1, \ldots, i_n} \in \mathbb{R}^m$ for all $i_1, \ldots, i \in \{0, 1\}$

In the case where Z is an interval; $Z = [\underline{\mathbf{z}}, \overline{\mathbf{z}}]$, $\underline{\mathbf{z}}, \overline{\mathbf{z}} \in \mathbb{R}^n$, we denote the set of vertices of this interval by

$$V = \prod_{i=1}^{n} \{\underline{z}^i, \overline{z}^i\}.$$

Let $\xi_i : \{\underline{z}^i, \overline{z}^i\} \to \{0, 1\}$ for all $i \in \{1, \ldots, n\}$ denote the indicator function

$$\xi_i(\underline{z}^i) = 0 \quad \xi_i(\overline{z}^i) = 1 \quad \forall i \in \{1, \ldots, n\}.$$

Proposition 4. (see (Belta and Habets, 2006, Proposition 1)).
Let $Z \subseteq \tilde{Z}$ be an n-dimensional interval, $g : \tilde{Z} \to \mathbb{R}^m$ a multi-affine function such that, for all the vertices $\mathbf{v} = (v^1, \ldots, v^n) \in V$ of Z, we have $g(v^1, \ldots, v^n) = \mathbf{y}_{\mathbf{v}}$. Then, for all $\mathbf{z} = (z^1, \ldots, z^n) \in \tilde{Z}$ the function g is uniquely given by

$$g(\mathbf{z}) = \sum_{\mathbf{v} \in V} \prod_{i=1}^{n} \left(\frac{z^i - \underline{z}^i}{\overline{z}^i - \underline{z}^i} \right)^{\xi_i(v^i)} \left(\frac{\overline{z}^i - z^i}{\overline{z}^i - \underline{z}^i} \right)^{1 - \xi_i(v^i)} \mathbf{y}_{\mathbf{v}}. \tag{14}$$

Although the claim in (Belta and Habets, 2006, Proposition 1) is only for the case when $Z = \tilde{Z}$, nothing in the proof prevents making this generalization. Proposition 4 allows us to estimate a multi-affine function by estimating the function's values on the vertices of a given interval included in its domain.

Lemma 1. (see (Belta and Habets, 2006, Lemma 2)).
Let $\mathbf{s} \in \mathbb{R}^m$ and $b \in \mathbb{R}$. Then, $\mathbf{s}^T g(\mathbf{z}) \bowtie b$ for all $\mathbf{z} \in Z$ if and only if $\mathbf{s}^T g(\mathbf{v}) \bowtie b$, for all $\mathbf{v} \in V$, where $\bowtie$ stands for any of $<, \leq, =, \geq, >$.

Given a partition $(Z_{\mathbf{r}})_{\mathbf{r} \in R}$ of the interval $Z \subseteq \mathbb{R}^n$, we denote the vertices of an interval $Z_{\mathbf{r}}$ by $V_{\mathbf{r}}$. A function $g : Z \to \mathbb{R}^m$ is piecewise multi-affine if for all $\mathbf{r} \in R$ the function is multi-affine on $Z_{\mathbf{r}}$.

Proposition 5. (see (Makdesi et al., 2023b, Proposition 4)). If a piecewise multi-affine function $g : Z \to \mathbb{R}^m$ is continuous on the grid points of the partition $(Z_{\mathbf{r}})_{\mathbf{r} \in R}$:

$$\lim_{\mathbf{z} \to \mathbf{v}} g(\mathbf{z}) = g(\mathbf{v}) \quad \forall \mathbf{r} \in R, \mathbf{v} \in V_{\mathbf{r}}$$

then g is continuous for all $\mathbf{z} \in Z$.

As a consequence of Proposition 5, estimating the values of a piecewise multi-affine function on the vertices of a given partition will result in a continuous function.

4.2 Compatible estimation

After introducing the piecewise multi-affine functions, we will now discuss how to use them to find estimations of the true

functions on the predefined partition. A regression problem to find such a compatible piecewise multi-affine function offline using all the collected data points was introduced in (Makdesi et al. (2023b)). Like the over-approximation, the same argument about the invalidity of this approach online is still relevant here. Therefore, we developed an online scheme to update the estimation locally.

Let $\hat{f} : Z \to \mathbb{R}^{n_y}$ be a piecewise multi-affine function. Given a data transition $(\mathbf{z}_k, \mathbf{y}_k) \in \mathscr{D}$, we found in the previous step, using binary search, that $\mathbf{z}_k \in Z_{\mathbf{r}(k)}$. We want to update the value of this function on the vertices of all the cells in the neighborhood that contain the data point, namely inside the window R_w introduced earlier. For all $\mathbf{r}' \in R_w(\mathbf{r}(k))$ we have

$$\mathbf{y}_k = \hat{f}_{\mathbf{r}'}(\mathbf{z}_k) + \mathbf{e}(\mathbf{z}_k) \tag{15}$$

Where $\hat{f}_{\mathbf{r}'}$ is the multi-affine function defined on the interval $Z_{\mathbf{r}'}$, and $\mathbf{e}(\mathbf{z}_k)$ is the residual of the estimation. Proposition 4 states that function $\hat{f}_{\mathbf{r}'}$ can be written as a linear combination of its value on the vertices of the interval $Z_{\mathbf{r}'}$.

$$\hat{f}_{\mathbf{r}'}(\mathbf{z}_k) = \sum_{\mathbf{v} \in V_{\mathbf{r}'}} \beta_{\mathbf{v},\mathbf{r}'}(\mathbf{z}_k)\hat{\mathbf{y}}_{\mathbf{v}}.$$

where $\beta_{\mathbf{v},\mathbf{r}'}$ represent the linear coefficients, and $\hat{\mathbf{y}}_{\mathbf{v}}$ is the value of the estimation function on the vertex $\mathbf{v}$. Let us denote the set of all vertices of all the cells $\mathbf{r}' \in R_w(\mathbf{r})$ by

$$\mathscr{V}(k) = \bigcup_{\mathbf{r}' \in R_w(\mathbf{r})} V_{\mathbf{r}'}.$$

The set $\mathscr{V}(k)$ is finite and thus can be numbered $\mathscr{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{n_w}\}$, where n_w is the number of all vertices of all the cells $\mathbf{r}' \in R_w(\mathbf{r}(k))$. For every $j \in \{1, \ldots, n_x\}$, We aggregate all the components of the function's estimation on the vertices of the window (i.e. $\hat{y}^j_{\mathbf{v}}$, for all $\mathbf{v} \in \mathscr{V}(k)$) in a single vector $\Psi_j(k) \in \mathbb{R}^{n_w}$. Then, the set of equations in (15) can be written in the matrix form as follows

$$\chi_j(k) = A(k) \cdot \Psi_j(k) + \mathbf{E}_j(k)$$

where $\chi_j(k) \in \mathbb{R}^{|R_w|}$ is a vector containing $|R_w|$ replications of y^j_k, $A(k) \in \mathbb{R}^{n_w \times |R_w|}$ is coefficients matrix, and $\mathbf{E}_j(k) \in \mathbb{R}^{|R_w|}$ the vector of residuals.

Given the diagonal weights' matrix H. We want to minimize the sum of the weighted square of errors

$$\begin{aligned} S(\Psi_j(k)) = \mathbf{E}_j^T(k) \cdot H \cdot \mathbf{E}_j(k) &= (\chi_j(k) - A(k) \cdot \Psi_j(k))^T \cdot \\ & H \cdot (\chi_j(k) - A(k) \cdot \Psi_j(k)) \\ &= \chi_j^T(k) \cdot H \cdot \chi_j(k) - 2\,(A^T(k) \cdot H \cdot \chi_j(k))^T \cdot \Psi_j(k) + \\ & \Psi_j^T(k) \cdot A^T(k) \cdot H \cdot A(k) \cdot \Psi_j(k) \end{aligned}$$

We will minimize S by implementing a stochastic gradient descent-like method

$$\Psi_j(k) \coloneqq \Psi_j(k) - \eta(k) \nabla S(\Psi_j(k))$$

Where

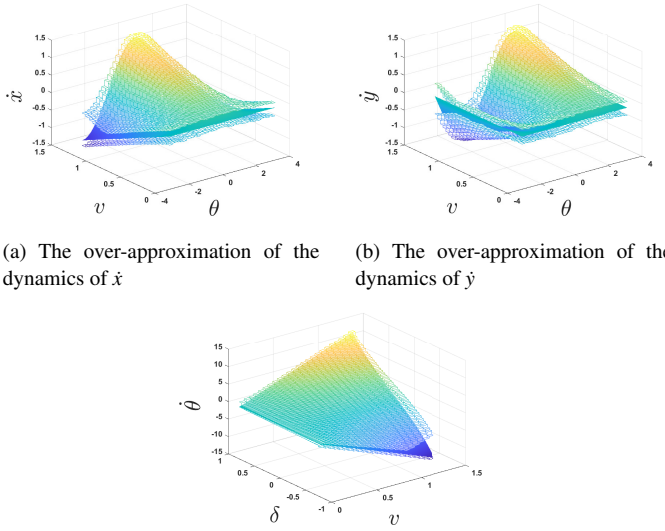
$$\nabla S(\Phi_j(k)) = 2A^T(k) \cdot H \cdot A(k) \cdot \Psi_j(k) - 2\,A^T(k) \cdot H \cdot \chi_j(k)$$

To ensure that the estimation $\hat{f}$ is compatible with the over-approximation, a final step is carried out to project the estimation onto the over-approximation. For all $i \in \{1, \ldots, n_w\}$

$$\Psi^i_j(k) \coloneqq P_{F_w(\mathbf{r}) \ominus D}\left(\Psi_j(k)\right), \mathbf{r} \in R_{\mathbf{v}_i} \tag{16}$$

Where $R_{\mathbf{v}_i}$ is the set of cells that contain $\mathbf{v}_i$ i.e. for all $\mathbf{r} \in R_{\mathbf{v}_i}, \mathbf{v}_i \in V_{\mathbf{r}}$.

Proposition 6. Given the partitions $(X_{\mathbf{q}})_{\mathbf{q} \in Q}$, $(U_{\mathbf{p}})_{\mathbf{p} \in P}$ of X and U, the piecewise multi-affine estimation function $\hat{f}$ updated locally as in (16) is compatible with the over-approximation F_w



(a) The over-approximation of the dynamics of $\dot{x}$

(b) The over-approximation of the dynamics of $\dot{y}$

(c) The over-approximation of the dynamics of $\dot{\theta}$

Fig. 2. Over-approximation of the bicycle dynamics. The true unknown functions are shown in solid

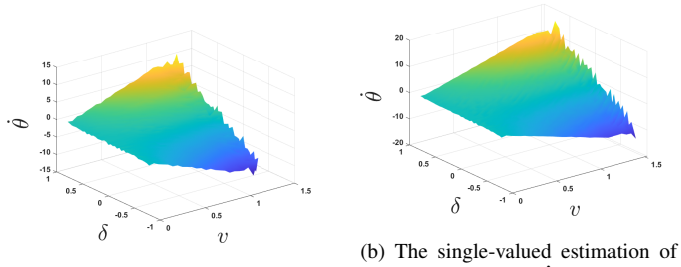
5. CASE STUDY

In this section, we present a path-planning problem where a mobile robot needs to navigate a given environment and reach four distinct regions while avoiding obstacles. We sampled a set of data from the vehicle dynamics at low speed. After that, We found the set of safe controllers and the estimation of dynamics. We then ran the system at those settings and collected new data points online. We chose a cost function that favors moving as fast as possible to learn the robot's dynamics at high speed. We do all of that while always ensuring safety. We consider the unicycle models defined by the following equations

$$\begin{aligned} \dot{x} &= v \cdot \cos(\theta) + d_1 \\ \dot{y} &= v \cdot \sin(\theta) + d_2 \\ \dot{\theta} &= \frac{v}{L} \tan(\delta) + d_3 \end{aligned} \tag{17}$$

Where x, y are the vehicle coordinates, θ is the heading angle, and L is the length of the vehicle. The system is discretized using the Euler method with a discretization time $dT = 0.2s$. We chose the value $L = 0.1$ m. Velocity is denoted v, and the steering angle δ. Velocity and steering angle are the control inputs. We chose for the states, inputs, and disturbance intervals the following $\theta \in [-\pi, \pi], \delta \in [-\frac{\pi}{4}, \frac{\pi}{4}], v \in (0, 2], \mathbf{d} \in [-0.05, 0.05]^3$, and partitioned those intervals uniformly into 40 cells each.

To find the over-approximation, we sampled $|\mathbb{K}| = 10^4$ data transitions. Only low speeds from the interval $v \in (0, 1]$ were chosen. We chose a window of size 11×11 (several window sizes were tested, and this one made the trade-off between the speed of computation from one side and the conservatism of the over-approximation and the "smoothness" of the estimation from the other side). The average time to update the over-approximation of each component was $t = 1.7ms$. Figure 2 shows the learned over-approximation of the dynamics. Although we sampled data points with speeds $v \leq 1$, the proposed technique allows for finding over-approximation for speeds $v > 1$, albeit more conservative, as seen in Figure 2. The environment where the robot operates, which can be seen in Figure 4, is a 5×5 m area. It was partitioned into 50×50 cells



(a) The single-valued estimation of the dynamics of $\dot{\theta}$ ($\hat{\theta}$) from the data collected offline

(b) The single-valued estimation of the dynamics of $\dot{\theta}$ ($\hat{\theta}$) After it was updated using data points collected online

Fig. 3. Estimation with piecewise multi-affine functions

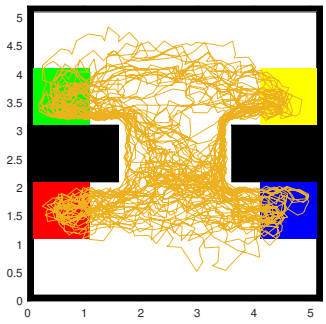


Fig. 4. The trajectory of the mobile robot visiting all four areas ten times

(i.e., x and y are partitioned into 50 cells each). The calculated over-approximation was used to find the maximal control invariant, and the set of safe controllers at each cell. The time it took to find the finite-state model and calculate the safety controller was $t = 424s$. Also, compatible estimations were calculated using the data collected offline, as seen in Figure 3a. The average time to update the estimation of each component was $t = 80ms$. A Gaussian weights matrix H was used, and the decreasing learning rate $\eta(k) = \frac{\text{step}}{k}$ was used. (The best results were obtained using step $= 10^{-3}$). Everything till now was done offline. We then ran the experiment, which consisted of making the robot visit the four colored areas in the order (red, yellow, green, blue) as fast as possible. This was done using the cost function $J = \sum_{i=1}^{N} ||\mathbf{x}(i|t) - \mathbf{x}_f|| + \frac{1}{v(i|t)}$, where $\mathbf{x}_f$ is the center of the colored areas, and it is changed to the next destination the moment the robot touches a given area. We collected the data online and updated the models, as can be seen in Figure 3. We were able to learn the dynamics of the system for higher speed that we did not sample for the offline stage. Most importantly, we did all of that while always ensuring safety.

6. CONCLUSION

In this paper, we introduced a novel approach to safe online learning-based MPC. Two models are learned online. The first is an over-approximation of the system and is meant to find the set of safe controllers. The second is a single-valued estimation of the dynamics and is used to minimize a given cost function. A case study showcasing the ability of the approach to learn the system's dynamics safely was carried out. Although promising results, more work should be done to improve the learning process. For example, we want to examine the question of exploration vs. exploitation.

REFERENCES

Aswani, A., Gonzalez, H., Sastry, S.S., and Tomlin, C. (2013). Provably safe and robust learning-based model predictive control. *Automatica*, 49(5), 1216–1226.

Belta, C. and Habets, L.C.G.J.M. (2006). Controlling a Class of Nonlinear Systems on Rectangles. *IEEE Transactions on Automatic Control*, 51(11), 1749–1759. doi:10.1109/TAC.2006.884957. Conference Name: IEEE Transactions on Automatic Control.

Belta, C., Yordanov, B., and Gol, E.A. (2017). *Formal methods for discrete-time dynamical systems*. Springer.

Berberich, J., Köhler, J., Müller, M.A., and Allgöwer, F. (2020). Data-driven model predictive control with stability and robustness guarantees. *IEEE Transactions on Automatic Control*, 66(4), 1702–1717.

Blanchini, F. and Miani, S. (2008). *Set-theoretic methods in control*, volume 78. Springer.

Canale, M., Fagiano, L., and Signorile, M. (2014). Nonlinear model predictive control from data: a set membership approach. *International Journal of Robust and Nonlinear Control*, 24(1), 123–139.

Devonport, A., Saoud, A., and Arcak, M. (2021). Symbolic abstractions from data: A pac learning approach. In *2021 60th IEEE Conference on Decision and Control (CDC)*, 599–604. doi:10.1109/CDC45484.2021.9683316.

Hewing, L., Wabersich, K.P., Menner, M., and Zeilinger, M.N. (2020). Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 269–296.

Kazemi, M., Majumdar, R., Salamati, M., Soudjani, S., and Wooding, B. (2022). Data-driven abstraction-based control synthesis. *arXiv preprint arXiv:2206.08069*.

Lavaei, A. and Frazzoli, E. (2022). Data-driven synthesis of symbolic abstractions with guaranteed confidence. *IEEE Control Systems Letters*, 7, 253–258.

Lavaei, A., Soudjani, S., Frazzoli, E., and Zamani, M. (2023). Constructing mdp abstractions using data with formal guarantees. *IEEE Control Systems Letters*, 7, 460–465. doi:10.1109/LCSYS.2022.3188535.

Makdesi, A., Girard, A., and Fribourg, L. (2023a). Data-driven models of monotone systems. *IEEE Transactions on Automatic Control*.

Makdesi, A., Girard, A., and Fribourg, L. (2023b). Safe learning-based model predictive control using the compatible models approach. *European Journal of Control*, 100849.

Milanese, M. and Novara, C. (2004). Set membership identification of nonlinear systems. *Automatica*, 40(6), 957–975.

Sadraddini, S. and Belta, C. (2018). Formal guarantees in data-driven model identification and control synthesis. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (part of CPS Week)*, 147–156.

Tabuada, P. (2009). *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer Publishing Company, Incorporated, 1st edition.

Zhang, X., Liu, J., Xu, X., Yu, S., and Chen, H. (2022). Robust learning-based predictive control for discrete-time nonlinear systems with unknown dynamics and state constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.