# Training Classifiers For Feedback Control With Safety In Mind

Hasan A. Poonawala [a], Niklas Lauffer [b], Ufuk Topcu [b]

[a] *Department of Mechanical Engineering, University of Kentucky, 506 Administration Drive, Lexington, KY 40506, USA*

[b] *Department of Aerospace Engineering, University of Texas at Austin, 210 E 24th St, Austin, TX 78712, USA*

**Abstract**

The sensors of many autonomous systems provide high-dimensional and information-rich measurements. The state of the system is a part of this information, however it is challenging to extract it from such measurements. An autonomous system cannot use traditional feedback control algorithms without knowledge of the state. We propose computational algorithms for the analysis and synthesis of classifier-enabled control architectures. We show how to train classifiers based on criteria that relate to both learning from data *and* properties of the resulting closed-loop system. The approach to deriving these algorithms involves modeling the resulting closed-loop system as a piecewise affine differential inclusion. The training method is based on the projected gradient descent algorithm. An application of this method to a navigation problem for a mobile robot demonstrates the capabilities of this approach.

*Key words:* Machine Leaning; Feedback Control; Classifiers; Lyapunov-based methods.

## 1 Introduction

A common situation for autonomous systems – e.g. drones, self-driving vehicles, collaborative robots – involves using information-rich sensors, which provide high-dimensional measurements, to control the state of the system in different environments [1]. Optical cameras [2] and three-dimensional light detection and ranging (LIDAR) sensors [3] are examples of such sensors. The high-dimensional measurements depend on the state of the robot in the environment, and it may be difficult to extract the state from this measurement.

Instead of attempting to extract a state, the autonomous system may only need to identify the current *situation* it is in and use a control strategy associated with that situation. Obstacle avoidance using proximity sensors such as SONAR [4] is an example of such a strategy, where the situation is proximity to any object, as identified by the SONAR readings. The problem of learning how to act in response to any measurement, given a finite set of measurements labeled with the corresponding situation, falls under the scope of supervised machine learning [5]. When the possible situations are finite, this problem is more specifically one of *classification*. The set of situations form the set of class labels, and the goal of supervised learning algorithms is to learn a *classifier* that assigns a unique class label to every measurement.

Given a classifier that maps measurements to class labels, the autonomous system uses a control input associated with the class label. This control input may be a constant vector, or function of the measurement. We refer to such a feedback control system, depicted in Figure 1, as a *classifier-in-the-loop* system.

**Motivation.** The standard approach to evaluating the suitability of a classifier is to look at the predicted label it assigns to measurements not included in the training set [5, 6]. The error rate on these measurements, known as the generalization error, should be low. This approach to assessing classifiers is meaningful when each measurement is generated independently using the same process, which is known as the independent and identically distributed (IID) assumption.

Methods such as deep learning achieve extremely low

⋆ This research was partially supported by the National Science Foundation Grants 1652113, CNS-1836900, and 1646522, and the University of Kentucky. Corresponding author H. A. Poonawala Tel. +1-859-323-7436. Fax +1-859-257-3304.

*Email addresses:* hasan.poonawala@uky.edu (Hasan A. Poonawala), nlauffer@utexas.edu (Niklas Lauffer), utopcu@utexas.edu (Ufuk Topcu).

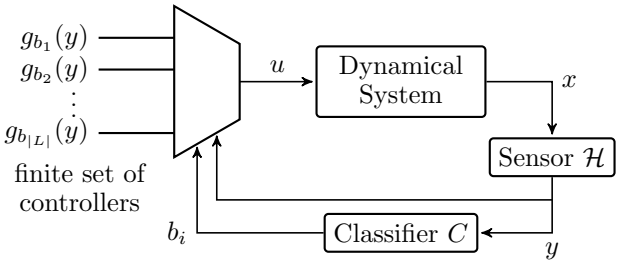


Fig. 1. A dynamical system with a classifier in the feedback loop. The measurement $y$ obtained by the sensor in state $x$ depends on an unknown map $\mathcal{H}$. The classifier converts the measurement into a label $b$ that determines which control law $g_b(y)$ is chosen for measurement-based feedback.

generalization error rates [7]. This success has led to significant increase in the use of classifier-in-the-loop control schemes [8–10]. Most classifier-based control solutions involve achieving good generalization error estimates and then empirically demonstrating suitable closed-loop behavior [10]. Recent work shows that low generalization error does not automatically imply safe or correct autonomous control [11–13]. When classifiers are used for control, the IID assumption does not hold. The classifier's predictions feed back into the evolution of the system, and the properties of the resulting closed-loop system need to be determined. Furthermore, methods for training classifiers – intended for use as feedback controllers – should preferably account for these closed-loop properties.

**Related Work.** Much of the work on learning and control with safety guarantees focuses on learning continuous state-based control functions where the dynamics of the state are almost entirely unknown [14–16]. These problems assume that the state can be observed, perhaps with added noise. Safe reinforcement learning [17–19] is similar to the work just mentioned, except that the rewards from taking actions must be learned. By contrast, our work focuses on the case where the state is observed indirectly through a high-dimensional information-rich measurement, similar to [10].

Most work on robustness and verification of the behavior of classifiers focuses on showing that the output of the classifier does not change under small enough perturbations of the input. The closed-loop dynamics of a classifier-in-the-loop system may cause the measurement to change significantly, with the class label switching over time. Our work explicitly tackles the switching aspect of classification-based control of continuous-state dynamical systems.

Prior work by the authors in [20] first proposed applying tools from switched systems analysis to classifier-in-the-loop systems. The analysis was limited to qualitative robustness guarantees using the notion of structural stability of dynamical systems. Further work in [21] proposes methods to synthesize classifiers using switched systems tools, by incorporating stability conditions into existing optimization-based training algorithms for classifiers.

**Contributions.** The primary objective of this paper is to develop algorithms that train classifiers for information-rich-sensor-based control, without intermediate state estimation. It makes two contributions towards achieving this objective. First, we show that piecewise affine differential inclusions provide a framework for modeling classifier-in-the-loop systems that a) accounts for switching behavior [22] due to classification of measurements at classifier boundaries, and b) incorporates robustness to feedback uncertainties when using classifiers trained from limited data. Second, we formulate the training problem for classifiers used in control as a two-step process. The first step involves the design of switching surfaces in the state-space using Lyapunov methods. The second step involves supervised learning methods that convert the state-space surfaces into classifiers in the measurement-space. Finally, we demonstrate that the empirical behavior of classification-based control algorithms for path-following, such as in [10], can be explained using the proposed approach.

Note that prior work by the authors in [21] formulated stability conditions on the parameters of classifiers of measurements. By using a two-step procedure instead, we overcome limitations of the approach in [21]. First, classifiers in the measurement space are no longer restricted to be linear classifiers. Second, we avoid needing to estimate and then linearize a closed-form map from state-space to measurement-space.

## 2 Preliminaries

*Notation.* The set $B_\epsilon(x) = \{y \in \mathbb{R}^n | \ \|x - y\| < \epsilon\}$ denotes an open ball of size $\epsilon$ centered at $x$. For a set $S$, we denote its interior as $\text{Int}(S)$, its closure as $\overline{S}$, its boundary as $\partial S$, and its cardinality as $|S|$. We denote the set of indices of a countable set $S$ as $I(S)$. The vector $\mathbf{1}$ denotes a vector with all elements equal to unity.

### *2.1 Partitions And Labeled Partitions*

A partition $\mathcal{P}(X)$ of a set $X \subseteq \mathbb{R}^n$ is a collection of subsets $\{X_i\}_{i \in I(\mathcal{P})}$, where $I(\mathcal{P})$ is an index set, $n \in \mathbb{N}$, $X_i \subseteq X$ and $\text{Int}(X_i) \neq \emptyset$ for each $i \in I(\mathcal{P})$, $X = \cup_{i \in I(\mathcal{P})} X_i$, and $\text{Int}(X_i) \cap \text{Int}(X_j) = \emptyset$ for each pair $i, j \in I(\mathcal{P})$ such that $i \neq j$.

A labeled partition is a tuple $(\mathcal{P}(X), L, \pi)$, where $\mathcal{P}(X)$ is a partition, $L$ is a finite set of labels, and $\pi: I(\mathcal{P}) \to L$ assigns labels to the regions in $\mathcal{P}(X)$.

### *2.2 Classifiers*

A classifier $C_\Phi: \Phi \to L$ is a map that assigns a unique label $b \in L$ to an input $\phi \in \Phi \subset \mathbb{R}^m$, where $\Phi$ is the

space of inputs. The set $L$ is typically a finite set. Examples of input spaces $\Phi$ include sensor outputs such as pixel intensity values from a digital camera, or distances to objects obtained from an array of sensors that use SONAR or infra-red light to measure distance.

A classifier $C_\Phi$ is typically parametrized by weights $w \in \mathbb{R}^s$. The process of determining weights $w$ is known as training the classifier, and uses training data consisting of pairs $(\phi^k, b^k)$, where $k$ is the index of a datum. Several methods exist for training a classifier using data [5].

When $L = \{b_1, b_2\}$, so that $|L| = 2$, we may classify inputs using a binary classifier $C_\Phi$ given by

$$C_\Phi(\phi) = \begin{cases} b_1, & \text{if} \quad h(\phi) \geq 0, \\ b_2, & \text{otherwise.} \end{cases} \tag{1}$$

where $h\colon \Phi \to \mathbb{R}$ is a continuous function. Often, the set $\{\phi \in \Phi\colon h(\phi) = 0\}$ is a hypersurface, which is called the classifier boundary.

When $|L| > 2$, it is possible to construct a classifier $C_\Phi\colon \Phi \to L$ by combining multiple binary classifiers (1) in different ways. One way is to construct a decision tree, where every node is a binary classifier. Another way is to train multiple binary classifiers, where each classifier distinguishes between one of the $\binom{|L|}{2}$ possible pairs of labels from $L$. This multi-label classification scheme is known as one-vs-one classification. Instead, we can train $|L|$ binary classifiers that separate each label from all other labels. This classification scheme is known as one-vs-all classification.

### *2.3 Labeled Partition Of A Classifier*

Consider a space $\Phi$ and a classifier $C_\Phi\colon \Phi \to L$. This classifier corresponds to a labeled partition of $\Phi$ as follows. We define a relation $R$ on $\Phi$ through $C_\Phi$, given by

$$\phi_1 R \phi_2 \iff C_\Phi(\phi_1) = C_\Phi(\phi_2).$$

In words, two points are related if they possess the same class label under $C_\Phi$. This relation is easily shown to be an equivalence relation, where the quotient space corresponds to $L$. The equivalence classes are subsets of $\Phi$, and so the classifier defines a partition $\mathcal{P}(\Phi)$. Let the map $\pi$ provide the label associated with each such region in this partition. In this way, a classifier $C_\Phi$ creates a labeled partition which we denote as $(\mathcal{P}(\Phi), L, \pi)$.

## 3 Classifier-in-the-Loop Systems

Consider a dynamical system with state $x \in X \subseteq \mathbb{R}^n$, input $u \in U \subseteq \mathbb{R}^p$ and measurement $y \in Y \subseteq \mathbb{R}^m$. The sets $X$, $U$, and $Y$ may be compact subsets of their respective vector spaces. The measurement $y$ obtained is high-dimensional and depends on the low-dimensional state through a map $\mathcal{H}\colon X \to Y$ that is not explicitly known. Therefore,

$$y = \mathcal{H}(x). \tag{2}$$

This assumption is valid for robots operating in static or slowly changing environments, so that its sensor measurements depend on its state.

One approach to feedback control for this system is to use a classifier $C_Y$ to choose a control input, without explicitly estimating the state. The classifier $C_Y$ specifies a control input $u \in U \subseteq \mathbb{R}^p$ through a map $g_{b_i}\colon Y \to U$ associated with each label $b_i \in L$. For example, $g_{b_i}(y)$ may be a constant vector $g_{b_i}(y) = u_{b_i}$, or a linear feedback $g_{b_i}(y) = K_{b_i}(y - y_{b_i})$ for some constant $y_{b_i} \in \mathbb{R}^m$. The classifier-based control is therefore

$$u(y) = g_{C_Y(y)}(y)\,. \tag{3}$$

Figure 1 depicts this control approach, which we refer to as a *classifier-in-the-loop system.*

As we describe in Section 6, we assume that the closed-loop dynamics under a classifier-based control (3) may be modeled as a state-dependent differential inclusion:

$$\dot{x}(t) \in \mathcal{A}(x).$$

For dynamical systems, we have three kinds of paired data sets, which we denote $D_{xb}$, $D_{yb}$, and $D_{xy}$. Let superscript $k$ denote the $k^{\text{th}}$ element of a data set. Then,

$$\begin{aligned} D_{xb} &= \{(x^k, b^k)\}, \text{for } k \in \{1, \ldots, |D_{xb}|\}, \\ D_{yb} &= \{(y^k, b^k)\}, \text{for } k \in \{1, \ldots, |D_{yb}|\}, \text{ and} \\ D_{xy} &= \{(x^k, y^k)\}, \text{for } k \in \{1, \ldots, |D_{xy}|\}. \end{aligned}$$

The data sets $D_{xb}$ and $D_{yb}$ correspond to labeled states and measurements respectively. The data set $D_{xy}$ corresponds to measurements observed in states. Most classifier-in-the-loop systems use data of the form $D_{yb}$. In this paper, we require availability of $D_{xb}$ and $D_{xy}$. The next section describes the control design problem we wish to solve for such systems.

## 4 Problem Statement

Training of a classifier usually focuses on classifying the data, and not on the implied closed-loop behavior. In this section, we specify control-oriented training problems.

We consider closed-loop behaviors related to *safety*, which we characterize as invariance of a set $S_{inv} \subseteq X$. This property depends on the behavior of solutions $x(t)$ of the system. Formally,

**Definition 1 (Positive Invariance)** *A set $S_{inv} \subseteq X$ is positively invariant if*

$$x(t_0) \in S_{inv} \implies x(t) \in S_{inv}\ \forall t > t_0.$$

This definition leads to the following problem.

**Problem 2** *Design a classifier $C_Y$ such that a given set $S_{inv} \subseteq X$ is invariant.*

In the rest of this paper, we develop a solution to Problem (2) when $S_{inv}$ is a convex polyhedral set containing the origin.

## 5 Training Measurement-Space Classifiers

In this section, we provide an overview of our approach to solving Problem 2. A fundamental idea in our approach to control-oriented training of classifiers $C_Y$ is that these classifiers define the boundaries of a piecewise dynamics model in the state space [20, 21]. Training a classifier, therefore, corresponds to designing these boundaries. These boundaries represent discontinuities in the control, or switching surfaces.

We refer to classifiers with domain $Y$ as *measurement-space* classifiers. Similarly, we refer to classifiers with domain $X$ as *state-space* classifiers. In the rest of this section, we provide details about the state-space classifier, and how this classifier is used to train measurement-space classifiers.

### *5.1 State-Space Classifiers*

We use the idea of an arrangement $W$ of hyperplanes in $\mathbb{R}^n$ to design a partition of $X \subseteq \mathbb{R}^n$. The arrangement $W$ consists of $N$ hyperplanes:

$$W = \{(w_1^i, w_0^i)\}_{i \in \{1,\dots,N\}}, \tag{4}$$

where $w_1^i \in \mathbb{R}^n$, $w_0^i \in \mathbb{R}$ for each $i \in \{1, \dots, N\}$. These hyperplanes create a partition $\mathcal{P}(X)$. The regions of $\mathcal{P}(X)$ are polytopes when $X$ is a either a polytope or $\mathbb{R}^n$.

By assigning labels from $L$ to regions in $\mathcal{P}(X)$ through a map $\pi$, we obtain a labeled partition. This labeled partition corresponds to a classifier $C_X$ composed of multiple binary classifiers, each classifier corresponding to one of the hyperplanes in $W$. The parameters of $C_X$ are therefore $W$. Each polytopic region $X_i$ of the partition $\mathcal{P}(X)$ will be defined by affine inequalities corresponding to the hyperplanes in $W$. We collect the inequalities defining $X_i$ into a matrix $E_i(W)$ and vector $e_i(W)$, so that

$$X_i = \{x \in X \colon E_i(W)x + e_i(W) \geq 0\}. \tag{5}$$

By construction, $E_i(W)$ and $e_i(W)$ are affine functions of $W$. We represent the classifier $C_X$ as

$$C_X(x) = \pi(i), \quad \text{if } E_i(W)x + e_i(W) \geq 0, \tag{6}$$

where $\pi \colon I(\mathcal{P}) \to L$ labels each region in $\mathcal{P}(X)$.

### *5.2 Measurement Space Classifiers*

In this section, we describe how to train a measurement-space classifier $C_Y$ *given* a state-space classifier $C_X$. We achieve this training by creating a new labeled data set $D_{yb}^X$, defined as

$$D_{yb}^X = \cup_{(x^k, y^k) \in D_{xy}} \left(y^k, C_X(x^k)\right).$$

This process can be seen as approximating the relationship $y = \mathcal{H}(x)$ using available paired data in $D_{xy}$. This approach enables control behaviors, associated with a few states, to be used for measurement-based feedback.

Any sufficiently complex classifier $C_Y$ may be used to classify the data set $D_{yb}^X$ [5]. One way to define $C_Y$ is to use use binary classifiers that mimic those defining $C_X$, but this approach is not required. For example, convolutional neural networks [7] may be used for optical images. Note that the work in [21] allows use of only binary linear classifiers to define $C_Y$.

This section introduced the idea of training measurement-space classifiers by first training state-space classifiers. Sections 6, 7 and 8 describe how we train these state-space classifiers.

## 6 Robust State Space Models Of Classifier-In-The-Loop Systems

In this section we derive a piecewise affine differential inclusion model corresponding to a classifier $C_X$ through the labeled partition $(\mathcal{P}(X), L, \pi)$ it creates. We then describe how we make this model robust to uncertainties.

For each label $b_i \in L$, we assume that the measurement-based control input $g_{b_i}(y)$ associated with $b_i$ creates closed-loop dynamics in state space $X$ that can be embedded in a piecewise affine differential inclusion [23]

$$\dot{x}(t) \in \mathcal{A}_{b_i}(x) = \operatorname{co}\left(\{A_k x + a_k\}_{k \in I_{\mathcal{A}_{b_i}}}\right), \tag{7}$$

where $\operatorname{co}(\cdot)$ is the convex hull operation, and $I_{\mathcal{A}_{b_i}}$ is the (finite) index set of the affine functions that define $\mathcal{A}_{b_i}(x)$. Affine differential inclusions (7) are often used to represent uncertainty in dynamics [24, 25] or approximate nonlinear dynamics. For example, let the dynamics in the state space be given by a model

$$\dot{x}(t) = f(x(t), u_{b_i}),$$

where $g_{b_i}(y) = u_{b_i}$ is a constant control input associated with label $b_i$. We embed the vector field $f(x, u_{b_i})$ in an affine differential inclusion $\mathcal{A}_{b_i}(x)$ of the form (7), such that $f(x, u_{b_i}) \in \mathcal{A}_{b_i}(x)$ for each $x \in S$, where $S$ is region of interest in the state space. Procedures for computing such an embedding are beyond the scope of this paper (see [26, 27], for example).

A classifier $C_X$ given by (5) and (6), and the dynamics models (7) for each label $b_i \in L$, together create the piecewise differential inclusion

$$\Omega(C_X)\colon \quad \dot{x}(t) \in \mathcal{A}(x) = \mathcal{A}_{\pi(i)}(x(t)), \text{ if } x \in X_i. \tag{8}$$

A classifier $C_Y$, trained by applying the methods in Section 5.2 to $C_X$, induces a classifier $C_X^Y$ through composition with $\mathcal{H}(x)$:

$$C_X^Y(x) = C_Y \circ \mathcal{H}(x). \tag{9}$$

In general, the boundaries of the partition due to $C_X^Y$ will not match those due to the designed $C_X$. Figure 2a depicts such a mismatch.

To make the design and analysis of classifiers robust, we derive a robust piecewise differential inclusion $\Omega_\Delta(C_X)$ from $\Omega(C_X)$ in (8) as follows. Figure 2b depicts an example of this procedure for one linear boundary. We define a new arrangement $W_\Delta$ from $W$ in (4), given by

$$\begin{aligned} W_\Delta = &\{(w_1^i, w_0^i + \Delta)\}_{i\in\{1,\dots,N\}} \\ &\cup \{(w_1^i, w_0^i - \Delta)\}_{i\in\{1,\dots,N\}}, \end{aligned} \tag{10}$$

where $\Delta > 0$ is a parameter to be chosen.

Let $\mathcal{P}_\Delta(X)$ represent the polytopic partition of $X$ associated with $W_\Delta$. The partition $\mathcal{P}_\Delta(X)$ has more regions than $\mathcal{P}(X)$. Let $X_j^\Delta \in \mathcal{P}_\Delta(X)$ represent one of these regions, and consider the index set

$$I(X_j^\Delta) = \left\{ i \in I(\mathcal{P})\colon X_j^\Delta \cap X_i \neq \emptyset \right\}.$$

Let the index set of the regions in $\mathcal{P}_\Delta(X)$ be $I(\mathcal{P}_\Delta)$. We define a differential inclusion

$$\mathcal{A}_j^\Delta(x) = \text{co}\left( \{\mathcal{A}_{\pi(i)}(x)\}_{i\in I(X_j^\Delta)} \right), \tag{11}$$

for each $j \in I(\mathcal{P}_\Delta)$. The regions $X_j^\Delta$ defined by the arrangement $W_\Delta$ in (10) and the dynamics (11) lead to the piecewise affine differential inclusion

$$\Omega_\Delta(C_X)\colon \quad \dot{x}(t) \in \mathcal{A}^\Delta(x) = \mathcal{A}_j^\Delta(x), \quad \text{if } x \in X_j^\Delta. \tag{12}$$

The set of trajectories of the robust model $\Omega_\Delta(C_X)$ will contain all those of the nominal model $\Omega(C_X)$, including sliding solutions [22], allowing analysis of the former to apply to the latter. The caveat to this approach is that $\Omega_\Delta(C_X)$ may exhibit far too many trajectories relative to $\Omega(C_X)$. Some of these trajectories may not satisfy the closed-loop properties we wish to certify, while a less conservative model may satisfy these control properties.

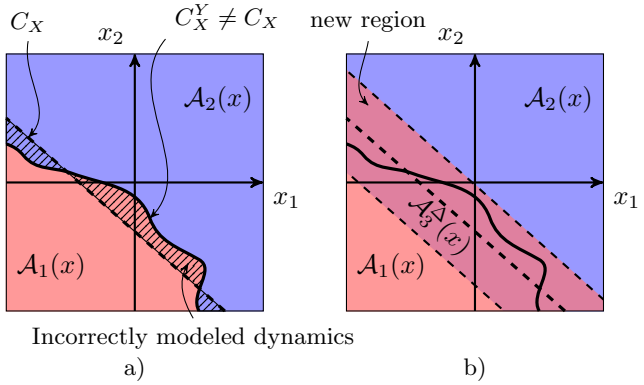


Fig. 2. [Figure best viewed in color.] Classifiers define switching surfaces and piecewise dynamics. a) The switching surface due to $C_Y$ in state space $X$ may not match the switching hyperplane of the designed classifier $C_X$, leading to modeling errors (shaded region). b) We model the uncertainty in switching surface by creating a new region with differential inclusion $\mathcal{A}_3^\Delta(x) = \text{co}\left(\mathcal{A}_1(x), \mathcal{A}_2(x)\right)$.

## 7 Control-Oriented Constraints On Classifier Parameters

In this section, we present the theoretical results that allow us to train $C_X$ in Section 8. These results are in the form of conditions on the set-valued Lie derivatives (defined below) of a polytopic Lyapunov function [28] along the solutions of differential inclusion $\Omega_\Delta(C_X)$ that guarantee closed-loop behaviors of $\Omega(C_X)$. These conditions are a modification of the approach in [23, 29].

### *7.1 Representing a Polytopic Lyapunov Function*

Consider a partition $\mathcal{Q}(\mathbb{R}^n) = \{Z_i\}_{i\in I(\mathcal{Q})}$ given by

$$Z_i = \{x \in \mathbb{R}^n\colon F_i x \geq 0\}, \tag{13}$$

where $F_i \in \mathbb{R}^{n\times n}$ is full rank. Each region $Z_i$ is an unbounded polyhedral cone with apex at the origin. The adjacent regions of $\mathcal{Q}$ are characterized by the set

$$I_{cont}(\mathcal{Q}) = \{(i,j) \in I(\mathcal{Q}) \times I(\mathcal{Q})\colon Z_i \cap Z_j \neq \emptyset\}. \tag{14}$$

We parametrize the common boundary between adjacent regions using the vector $\eta_{ij} \in \mathbb{R}^n$, so that $x \in Z_i \cap Z_j \implies \eta_{ij}^T x = 0$. Furthermore, we assume that $Z_i \subseteq \{x \in X\colon \eta_{ij}^T x \geq 0\}$.

We define a candidate Lyapunov function $V_{\mathcal{Q}}(x)$ using partition $\mathcal{Q}(\mathbb{R}^n)$ and a collection of vectors $\{p_i\}_{i\in I(\mathcal{Q})}$ as

$$V_{\mathcal{Q}}(x) = p_i^T x, \text{ if } x \in Z_i. \tag{15}$$

By construction $V_{\mathcal{Q}}(0) = 0$, however $V_{\mathcal{Q}}(x)$ is possibly multi-valued at the boundaries of regions in $\mathcal{Q}(\mathbb{R}^n)$. We need $V_{\mathcal{Q}}(x)$ to be positive, locally Lipschitz, and regular [23]. The next two Lemmas describe conditions under which $V_{\mathcal{Q}}(x)$ possesses these properties.

**Lemma 3 (Lemma** 4.7 **[29])** *The following are equivalent*

*(1)* $Fx \geq 0, x \neq 0 \implies p^T x > 0$.
*(2)* $\exists v \in \mathbb{R}^n$ *where* $v > 0$ *such that* $F^T v = p$.

**Lemma 4** *Consider a function* $V_{\mathcal{Q}}(x)$ *as in equations* (13), (14) *and* (15). *If there exist variables* $\mu_i$ *for* $i \in I(\mathcal{Q})$, $\lambda_{ij}$ *for* $(i,j) \in I_{cont}(\mathcal{Q})$, *and* $\epsilon > 0$ *that satisfy*

$$\begin{aligned} p_i = F_i^T \mu_i, &\quad \forall i \in I(\mathcal{Q}), &(16)\\ \mu_i \geq \epsilon \mathbf{1}, &\quad \forall i \in I(\mathcal{Q}), &(17)\\ p_i - p_j = \lambda_{ij}\eta_{ij}, &\quad \forall (i,j) \in I_{cont}(\mathcal{Q}), \text{ and} &(18)\\ \lambda_{ij} \geq 0, &\quad \forall (i,j) \in I_{cont}(\mathcal{Q}), &(19) \end{aligned}$$

*then* $V_{\mathcal{Q}}(x)$ *is positive definite, locally Lipschitz, and regular.*

**PROOF.** $V_{\mathcal{Q}}(x)$ is piecewise linear. Assume that variables satisfying (16)-(19) exist. When $x \in Z_i \cap Z_j$, then $\eta_{ij}^T x = 0$ by definition. Therefore, condition (18) implies that $V_{\mathcal{Q}}(x)$ is continuous at its boundaries, so that $V_{\mathcal{Q}}(x)$ is locally Lipschitz. By construction, $V_{\mathcal{Q}}(0) = 0$. By Lemma 3, if conditions (16) and (17) hold, then $V_{\mathcal{Q}}(x) > 0$ when $x \neq 0$. Therefore, $V_{\mathcal{Q}}(x)$ is positive definite. If $\lambda_{ij} \geq 0$ for all $(i,j) \in I_{cont}(\mathcal{Q})$, and $V_{\mathcal{Q}}(x)$ is continuous, then $V_{\mathcal{Q}}(x)$ is convex. Since convex functions are regular [23], we conclude that $V_{\mathcal{Q}}(x)$ is regular. □

### *7.2 Lyapunov-Based Conditions For Set Invariance*

The closed-loop model $\Omega_\Delta(C_X)$ in (12) we derive for a classifier-in-the-loop system in Section 3 is inherently discontinuous. We follow the approach in [23] for analyzing such models, beginning with some definitions below.

**Definition 5 (Generalized gradient [23])** *Let* $V$ *be a locally Lipschitz function, and let* $Z$ *be the set of points where* $V$ *fails to be differentiable. The generalized gradient* $DV(x)$ *at* $x$ *is defined by*

$$DV(x) = \text{co}\left(\{\lim_{i\to\infty} \nabla V(x_i) : x_i \to x, x_i \notin S \cup Z\}\right),$$

*where* $S$ *is any set of measure zero that can be arbitrarily chosen to simplify the computation. The resulting set* $DV(x)$ *is independent of the choice of* $S$.

**Definition 6 (Set-valued Lie Derivative [23])** *Given a locally Lipschitz function* $V : \mathbb{R}^n \to \mathbb{R}$ *and a set-valued map* $\mathcal{A}: \mathbb{R}^n \to 2^{\mathbb{R}^n}$, *the set-valued Lie derivative* $\mathcal{L}_{\mathcal{A}} V(x)$ *of* $V$ *with respect to* $\mathcal{A}$ *at* $x \in \mathbb{R}^n$ *is given by*

$$\begin{aligned} &\mathcal{L}_{\mathcal{A}} V(x)\\ &= \{a \in \mathbb{R} : \exists v \in \mathcal{A}(x) \text{ and } \zeta \in DV(x) \text{ such that } \zeta^T v = a \,\forall\}. \end{aligned}$$

**Definition 7 (Caratheodory solution)** *A Caratheodory solution of* $\dot{x}(t) \in \mathcal{A}(x)$ *defined on* $[t_0, t_1] \subset [0, \infty)$ *is an absolutely continuous map* $x: [t_0, t_1] \to \mathbb{R}^n$ *such that* $\dot{x}(t) \in \mathcal{A}(x)$ *for almost every* $t \in [t_0, t_1]$.

Note that Caratheodory solutions of differential inclusions are identical to Filippov solutions of discontinuous systems with the typical convex relaxation of the dynamics [22, 23]. These definitions enable us to describe, in the next sections, how we derive conditions on the closed-loop system that correspond to desired behavior.

One way to determine if the system $\Omega_\Delta(C_X)$ possesses desired properties is by finding a candidate Lyapunov function $V_{\mathcal{Q}}(x)$ whose set-valued Lie derivative along solutions of $\Omega_\Delta(C_X)$ satisfies

$$\max \mathcal{L}_{\mathcal{A}} V_{\mathcal{Q}}(x) \leq 0, \forall x \in X. \tag{20}$$

Due to the piecewise nature of $\Omega_\Delta(C_X)$, we must verify (20) on multiple regions formed by the intersection of regions in $\mathcal{P}_\Delta(X)$ and $\mathcal{Q}(\mathbb{R}^n)$. Let $\mathcal{R}(X)$ be the partition whose elements are these intersections. Then

$$\begin{aligned} \mathcal{R}(X) &= \{R_{ij}\}_{(i,j)\in I(\mathcal{R})}, \text{ where} &(21)\\ I(\mathcal{R}) &= \{(i,j) \in I(\mathcal{Q}) \times I(\mathcal{P}_\Delta) : Z_i \cap X_j^\Delta \neq \emptyset\}. &(22) \end{aligned}$$

For each $(i,j) \in I(\mathcal{R})$, let

$$R_{ij} = Z_i \cap X_j^\Delta = \{x \in X : G_{ij}x + g_{ij} \geq 0\}, \tag{23}$$

where $G_{ij}, g_{ij}$ depend on $W$, $\Delta$, $\{p_i\}_{i\in I(\mathcal{P})}$, and $\{F_i\}_{i\in I(\mathcal{P})}$. To check if (20) holds for each region in $\mathcal{R}(X)$, we use the following results.

**Lemma 8** *Let the set* $\{x \in \mathbb{R}^n | Gx + g \geq 0\}$ *be nonempty, where* $G \in \mathbb{R}^{l\times n}$, $g \in \mathbb{R}^l$ *for some* $l \in \mathbb{N}$. *Let* $p \in \mathbb{R}^n$ *and* $q \in R$. *Then, the following are equivalent*

*(1)* $Gx + g \geq 0 \implies p^T x + q \leq 0$.
*(2)* $\exists v \in \mathbb{R}^n$, *where* $v \geq 0$ *such that* $G^T v + p = 0$ *and* $g^T v + q \leq 0$.

**PROOF.** We use an inhomogenous Theorem of Alternatives due to Duffin [30]. The theorem states that either $Gx + g \geq 0$ and $p^T x + q > 0$ are feasible in $x$, or one of the following are feasible in $v$:

(1) $G^T v + p = 0$, $g^T v + q \leq 0$, $v \geq 0$.
(2) $G^T v = 0$, $g^T v < 0$, $v \geq 0$.

By Gale's Theorem of Alternatives [31], the system of equations $G^T v = 0$, $g^T v < 0$, $v \geq 0$ is feasible if and only if $Gx + g \geq 0$ is infeasible. Infeasibility of the system $Gx + g \geq 0$ and $p^T x + q > 0$, when $Gx + g \geq 0$ is feasible, is equivalent to the implication $Gx + g \geq 0 \implies p^T x + q \leq 0$. Therefore, when $Gx + g \geq 0$ is feasible, Duffin's Theorem of Alternatives reduces to a form that proves this Lemma. □

**Lemma 9** *Let $V(x) = p^T x$, where $p, x \in \mathbb{R}^n$. Let*

$$S = \{x \in X : Gx + g \geq 0\}, \text{ and} \tag{24}$$
$$\mathcal{A}(x) = \text{co}\left(\{A_k x + a_k\}_{k \in I(\mathcal{A})}\right), \tag{25}$$

*where $G \in \mathbb{R}^{l \times n}$, $g \in \mathbb{R}^l$ for some $l \in \mathbb{N}$, and $A_k \in \mathbb{R}^{n \times n}$, $a_k \in \mathbb{R}^n$ for each $k \in I(\mathcal{A})$. Consider a piecewise affine dynamical system given by*

$$\dot{x} \in \mathcal{A}(x) \quad \forall x \in S. \tag{26}$$

*If there exist $\nu_k \geq 0$ for each $k \in I(\mathcal{A})$ satisfying*

$$G^T \nu_k + A_k^T p = 0, \tag{27}$$
$$g^T \nu_k + a_k^T p \leq 0, \text{ and} \tag{28}$$
$$\nu_k \geq 0, \tag{29}$$

*then*

$$\max \mathcal{L}_{\mathcal{A}} V(x) \leq 0 \ \forall x \in S.$$

**PROOF.** If $V(x) = p^T x$, then its generalized gradient is simply its gradient $DV(x) = p$. At each $x$, $\mathcal{A}(x)$ is the convex hull of vectors $A_k x + a_k$ for $k \in I(\mathcal{A})$. The set-valued Lie derivative $\mathcal{L}_{\mathcal{A}} V(x)$ at $x$ (Definition 6) is

$$\mathcal{L}_{\mathcal{A}} V(x) = \text{co}\left(\{p^T (A_k x + a_k)\}_{k \in I(\mathcal{A})}\right). \tag{30}$$

If conditions (27)-(29) are satisfied for some $\nu_k$, for each $k \in I(\mathcal{A})$, then by Lemma 8, $p^T (A_k x + a_k) \leq 0$ for all $k \in I(\mathcal{A})$ and all $x \in S$. This conclusion implies that $\mathcal{L}_{\mathcal{A}} V(x) \leq 0$ for each $x \in S$, proving the Lemma. □

To apply Lemma 9 to all regions of $\mathcal{R}(X)$, where each region has an inclusion $\mathcal{A}_j^\Delta$, we define the index set

$$I_{dec}(\mathcal{R}) = \{(i, j, k) \in I(\mathcal{R}) \times \mathbb{N} : k \in I(\mathcal{A}_j^\Delta)\}. \tag{31}$$

We now state the main result.

**Theorem 10** *Consider a piecewise affine differential inclusion $\Omega_\Delta(C_X)$ of the form (12), and a set $S_{inv}$ defined by*

$$S_{inv} = \{x \in \mathbb{R}^n : V_{\mathcal{Q}}(x) \leq 1\}, \tag{32}$$

*where $V_{\mathcal{Q}}(x)$ is given by (15) and satisfies the conditions of Lemma 4. Assume that $X$ contains the boundary $\partial S_{inv}$ of $S_{inv}$. If there exist variables $\nu_{ijk} \in \mathbb{R}^n$ that satisfy*

$$G_{ij}^T \nu_{ijk} + A_k^T p_i = 0, \tag{33}$$
$$g_{ij}^T \nu_{ijk} + a_k^T p_i \leq 0, \text{ and} \tag{34}$$
$$\nu_{ijk} \geq 0, \tag{35}$$

*for each $(i, j, k) \in I_{dec}(\mathcal{R})$, then $S_{inv}$ is invariant under dynamics $\Omega(C_X)$.*

## 8 Control-Oriented Training Of Classifiers

In this section, we present an algorithm to train a classifier $C_X$ corresponding to arrangement $W$ using the results of Section 7. This algorithm involves solving a bilinear optimization problem that combines control-oriented conditions (33)-(35) in Theorem 10 with conditions related to classifying data.

We will constrain the hyperplanes $W$ that define classifier $C_X$ to correctly classify the labeled dataset $D_{xb}$. Recall that $W$ defines a partition $\mathcal{P}(X)$ whose regions we label through map $\pi$. Let index set $I(D)$ identify the data in $D_{xb}$ associated with each region $X_i$ in $\mathcal{P}(X)$:

$$I(D) = \{(i, k) \in I(\mathcal{P}) \times I(D_{xb}) : \pi(i) = b^k\}. \tag{36}$$

Let $S_{inv}$ be a convex polygon containing the origin. We may represent this convex polygon as the set $\{x \in \mathbb{R}^n : V_{\mathcal{Q}}(x) \leq 1\}$ for an appropriate partition $\mathcal{Q}(\mathbb{R}^n)$ and parameters $\{p_i\}_{i \in I(\mathcal{Q})}$. By construction, the corresponding function $V_{\mathcal{Q}}(x)$ will be positive definite, locally Lipschitz, and regular. Let the regions $X_j^\Delta$ of $\Omega_\Delta(C_X)$ be given by

$$X_j^\Delta = \{x \in X : E_j(W_\Delta) x + e_j(W_\Delta) \geq 0\}. \tag{37}$$

We derive the matrices and vectors in (23) that define the regions in partition $I(\mathcal{R})$ as

$$G_{ij} = \begin{bmatrix} E_j(W_\Delta) \\ F_i \\ p_i^T \\ -p_i^T \end{bmatrix}, \ g_{ij} = \begin{bmatrix} e_j(W_\Delta) \\ 0 \\ -1 \\ -1 \end{bmatrix}. \tag{38}$$

With this notation, we define the following optimization problem.

$$\min_{W,\nu_{ijk}} \sum_{1}^{N} \|w_1^i\| \tag{39}$$

s.t.

$$E_i(W)x^k + e_i(W) \geq \mathbf{1} \quad \forall (i,k) \in I(D) \tag{40}$$

$$G_{ij}^T \nu_{ijk} + A_k^T p_i = 0 \quad \forall (i,j,k) \in I_{dec}(\mathcal{R}), \tag{41}$$

$$g_{ij}^T \nu_{ijk} + a_k^T p_i \leq 0, \quad \forall (i,j,k) \in I_{dec}(\mathcal{R}), \tag{42}$$

$$\nu_{ijk} \geq 0, \quad \forall (i,j,k) \in I_{dec}(\mathcal{R}), \tag{43}$$

The optimization variables include $W$, $\mu_i$, and $\nu_{ijk}$ for all values of indices as mentioned in (41)-(43). The objective function (39) and (40) implements training of multiple support vector machines [32] that classify datasets defined by $I(D)$. The constraints (41)-(43) are bilinear in the variables of the optimization problem. A feasible solution of (39)-(43) defines a classifier that separates the training data $D_{xb}$ and renders $S_{inv}$ positively invariant.

Optimization problems with bilinear constraints are typically NP-hard. We solve this optimization problem using projected gradient descent [33]. The projection step is solved using Alternate Convex Search (ACS) [34]. To implement the ACS, we group the variables of the optimization into two groups. One group contains $W$ and $\{p_i\}_{i \in I(\mathcal{Q})}$, which define regions, and the second group consists of $\nu_{ijk}$ for $(i,j,k) \in I_{dec}(\mathcal{R})$. ACS is a heuristic approach, and convergence is not guaranteed.

## 9 Case Study: Path Following

This case study is motivated by the work in [10]. In that work, the authors collect three types of measurements corresponding to different headings relative to the path, and train a deep-neural-network to classify camera images into one of three simple control actions: moving forward, turning left, or turning right. We mimic this setting using the Gazebo robot simulation environment. We task a quadrotor equipped with an infra-red-based scanning device to navigate a canyon-like terrain [21]. The quadrotor must follow the path defined by this canyon, while avoiding its sides. The varying curvature of the path and irregularity of the canyon present uncertainty in the dynamics and the relationship between states and measurements. We use the optimization package `cvx` and MatLAB R2018b to train classifiers, using a computer with a 2.6 GHz processor and 16 GB RAM.

### *9.1 Modeling*

We model the quadrotor kinematics as a differential-drive mobile robot. That is, we command the quadrotor to achieve has a forward velocity $v$ and an angular velocity $\omega$. This path defined by the canyon has varying

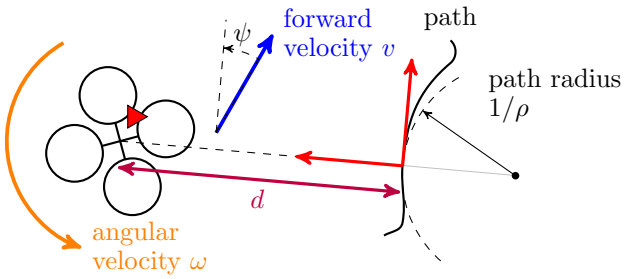


Fig. 3. A quadrotor with forward speed $v$, and angular velocity $\omega$. The curved black line represents a local segment of the path, with instantaneous path curvature $\rho$, that the quadrotor must follow. The local Frenet-Serret frame (red) attached to the path is also shown. The quadrotor's state consists of the offset $d$ and angle $\psi$ with respect to the path.

curvature, denoted by $\rho$ (see Figure 3). We can attach a moving Frenet-Serret frame to this path and express the dynamics of the quadrotor within this frame. The configuration of the quadrotor in the Frenet-Serret frame is $x = (\psi, d)$, where angle $\psi$ is the heading of the quadrotor with respect to the path-aligned axis of the frame, and offset $d$ is the distance between the quadrotor's location and the origin of the frame.

The quadrotor uses three control inputs like in [10]: $u_{b_1} = \begin{bmatrix} v^* & 0 \end{bmatrix}^T$ (move forward), $u_{b_2} = \begin{bmatrix} 0 & \omega^* \end{bmatrix}^T$ (turn left), and $u_{b_3} = \begin{bmatrix} 0 & -\omega^* \end{bmatrix}^T$ (turn right), where $v^* > 0$ and $\omega^* > 0$ are constants, so that $L = \{b_1, b_2, b_3\}$. The differential inclusions $\mathcal{A}_{b_2}(x)$ and $\mathcal{A}_{b_3}(x)$ are respectively

$$\mathcal{A}_{b_2}(x) = u_{b_2} \text{ and } \mathcal{A}_{b_3}(x) = u_{b_3}.$$

The dynamics under label $b_1$ are more complex, and depend on the curvature $\rho$ of the path. For a given constant curvature, it is given by a vector field $f_{b_1}(x)$, where

$$f_{b_1}(x) = \begin{bmatrix} \frac{v^* \rho \cos(\psi)}{1+\rho d} \\ v^* \sin(\psi) \end{bmatrix}. \tag{44}$$

We approximate this uncertain nonlinear dynamics by assuming that $|\psi|$, $|d|$, and $|\rho|$ are bounded by $\psi_{\max}$, $d_{\max}$ and $\rho_{max}$ respectively, where $\psi_{\max} < \pi/2$. Noting that $\sin\psi \approx \psi$ near $\psi = 0$, we model (44) through the affine differential inclusion

$$\mathcal{A}_{b_1}(x) = \text{co}\left(\{Ax + a, Ax - a\}\right), \text{ where}$$

$$A = \begin{bmatrix} 0 & 0 \\ v^* & 0 \end{bmatrix}, \quad a = \begin{bmatrix} v^* \alpha \\ 0 \end{bmatrix}, \text{ and}$$

$$\alpha = \max\left( \left| \frac{\rho_{\max} \cos\psi_{\max}}{1 + \rho_{\max} d_{\max}} \right|, \left| \frac{\rho_{\max} \cos\psi_{\max}}{1 - \rho_{\max} d_{\max}} \right| \right).$$

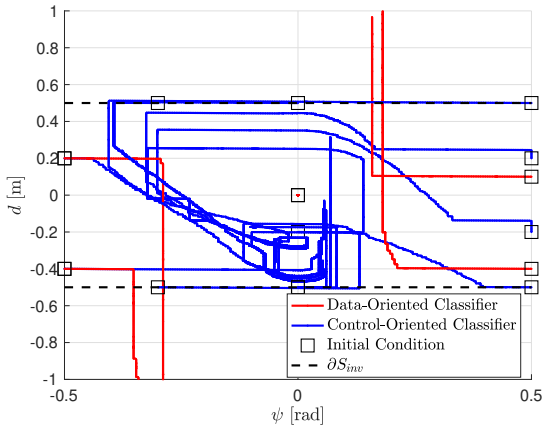


Fig. 4. Gazebo simulation results for set invariance. The trajectory in blue corresponds to a classifier trained using control-oriented constraints (40)-(43) to make the set $S_{inv}$ invariant. The trajectory in red corresponds to a classifier trained without constraints, and this trajectory leaves $S_{inv}$.

### *9.2 Training Data And Classifier*

We collect training data in a canyon created using a height map. This canyon is 5m deep, 1m wide on average at the bottom, and widens to an average width of 5m at the top. We consider a canyon with zero curvature. The laser scanner provides a measurement $y \in \mathbb{R}^{200}$, and has a field of view of 2 rad.

The dataset $D_{xy}$ consists of 180 pairs $(x^k, y^k, )$ where $x^k$ are uniformly sampled from the unit square. This dataset has no class labels. The dataset $D_{xb}$ consists of pairs $(x^k, b^k, )$ where $x^k = (\psi^k, d^k)$. For all $k \in \{1, \ldots, |D_{xb}|\}$, $d^k = 0$. The data points with state $x^k$ for which $\psi^k$ is $\pi/6$ rad, 0 rad, or $-\pi/6$ rad are labeled as $b_3$, $b_1$, and $b_2$ respectively. The set $D_{xb}$ mimics the data collected in [10]. For each state in $D_{xb}$, we also have the measurement observed in that set, thereby creating the dataset $D_{yb}$ consisting of labeled measurements.

Based on the three labeled states, we use a classifier $C_X$ consisting of two hyperplanes

$$W = \{(w_1^1, w_0^1), (w_1^2, w_0^2)\}. \tag{45}$$

The classification rule is then as follows:

$$C_X(x) = \begin{cases} b_2, & w_1^1 x + w_0^1 < 0 \text{ and } w_1^2 x + w_0^2 \geq 0, \\ b_3, & w_1^1 x + w_0^1 \geq 0 \text{ and } w_1^2 x + w_0^2 < 0, \\ b_1, & \text{otherwise.} \end{cases}$$

This choice partitions $X$ into three regions.

### *9.3 Training And Simulation Results*

We train two state-space classifiers, a control-oriented classifier designed to make a set $S_{inv}$ invariant, and a data-oriented classifier that only classifies the data in $D_{xb}$. We use $D_{xy}$ to convert both these classifiers into measurement-space classifiers (support vector machines) for use in simulation. The invariant set $S_{inv}$ is given by $S_{inv} = \{x \in \mathbb{R}^2 : |d| \leq 0.5 \text{ m}, |\psi| \leq \pi/2 \text{ rad}\}$. We take $v^*$ and $\omega^*$ to be 0.5 m/s and 0.15 rad/s respectively. We use the dynamics in Section 9.1 to define $\Omega_\Delta(C_X)$, where $\Delta = 0.05$, $\rho_{max} = 0.2\text{m}^{-1}$. The control-oriented classifier is trained by solving (39)-(43). The data-oriented classifier only solves (39)-(40).

Figure 4 shows trajectories corresponding to simulations, in Gazebo, of classifier-based control for navigation of a quadrotor. The blue and red trajectories corresponds to control-oriented and data-oriented classifiers respectively. The use of control-oriented constraints appears to render $S_{inv}$ positively invariant. Trajectories travel along the canyon, with non-zero net forward motion. The data-oriented classifier will lead to vertical switching surfaces, since the data lies on the $\psi$-axis. The diverging values of $d$ due to these vertical surfaces are predicted by switched systems methods. The control-oriented constraints therefore modify the switching surfaces to produce the desired set invariance, while classifying available data.

## 10 Discussion And Future Work

We have presented a training algorithm for classifiers that incorporates control-oriented constraints on the classifier parameters. These constraints are a result of modeling the closed-loop system as a piecewise affine differential inclusion, and using polytopic Lyapunov functions to verify the desired closed-loop properties. To apply these constraints on the measurement-space classifier, we use a two-step procedure. We constrain a classifier in the state space to create an invariant set. Then, we train a classifier in the measurement space that approximately creates the same boundaries in the state space as the designed state-space classifier.

**Limitations and Future Work.** The approach is limited to cases where we know the state-space dynamics corresponding to output feedback. This approach is reasonable when we use constant control inputs for each label. An alternate approach is to learn low dimensional latent-space dynamical models directly from measurement data. We will explore such an approach in future. A second issue is that the bilinear constraints create a non-convex optimization problem with few guarantees. We will explore the use of ADMM techniques applied to bi-convex problems [35] to solve the projection step. Finally, the piecewise approach may lead to significant computational cost as the dimension of the state space increases. An interesting avenue of future work is to extend the framework to use the history of inputs and measurements in the classification, potentially improving the closed-loop behavior and performance.

## References


[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005.

[2] S. Hutchinson, "Vision-based control of robot motion," *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP 2010. Lecture Notes in Computer Science*, vol. 6419, 2010.

[3] Y. Li and E. B. Olson, "Extracting general-purpose features from lidar data," in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 1388–1393.

[4] A. Flynn, "Combining sonar and infrared sensors for mobile robot navigation," *The International Journal of Robotics Research*, vol. 7, no. 6, pp. 5–14, 1988.

[5] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning, Data Mining, Interference, and Prediction*. Springer, 2009.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[8] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," `arXiv:1604.07316`, 2016.

[9] S. Ghosh, A. Mercier, D. Pichapati, S. Jha, V. Yegneswaran, and P. Lincoln, "Trusted neural networks for safety-constrained autonomous control," `arXiv:1805.07075`, 2018.

[10] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro *et al.*, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, July 2016.

[11] S. Shalev-Shwartz, O. Shamir, and S. Shammah, "Failures of gradient-based deep learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 3067–3075.

[12] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[13] M. S. Ramanagopal, C. Anderson, R. Vasudevan, and M. Johnson-Roberson, "Failing to learn: Autonomously identifying perception failures for self-driving cars," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3860–3867, Oct 2018.

[14] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 908–919.

[15] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, pp. 1–1, 2018.

[16] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, July 2018.

[17] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794.

[18] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verisig: Verifying safety properties of hybrid systems with neural network controllers," in *Proceedings of the 22Nd ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '19. New York, NY, USA: ACM, 2019, pp. 169–178.

[19] S. M. Richards, F. Berkenkamp, and A. Krause, "The lyapunov neural network: Adaptive stability certification for safe learning of dynamic systems," in *CoRL*, 2018.

[20] H. A. Poonawala and U. Topcu, "Robustness of classifier-in-the-loop control systems: A hybrid-systems approach," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2738–2743.

[21] H. A. Poonawala, N. Lauffer, and U. Topcu, "Training classifiers for feedback control," in *2019 American Control Conference (ACC)*, 2019, pp. 4961–4967.

[22] A. F. Filippov and F. M. Arscott, *Differential equations with discontinuous righthand sides*, ser. Mathematics and its Applications, 1988.

[23] J. Cortes, "Discontinuous dynamical systems," *IEEE Control Systems Magazine*, vol. 28, no. 3, pp. 36–73, June 2008.

[24] T. Soga and N. Otsuka, "Quadratic stabilizability for polytopic uncertain continuous-time switched linear systems by output feedback," in *Proceedings of the 2010 American Control Conference*, June 2010, pp. 3920–3925.

[25] G. Zhai, H. Lin, and P. J. Antsaklis, "Quadratic stabilizability of switched linear systems with polytopic uncertainties," *International Journal of Control*, vol. 76, no. 7, pp. 747–753, 2003.

[26] L. O. Chua and A.-C. Deng, "Canonical piecewise-linear representation," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 1, pp. 101–111, 1988.

[27] A. Toriello and J. P. Vielma, "Fitting piecewise linear continuous functions," *European Journal of Operational Research*, vol. 219, no. 1, pp. 86 – 95, 2012.

[28] F. Blanchini, "Nonquadratic lyapunov functions for robust control," *Automatica*, vol. 31, no. 3, pp. 451 – 461, 1995.

[29] M. Johansson, "Piecewise linear control systems," Ph.D. dissertation, Lund University, 1999.

[30] O. L. Mangasarian, *Nonlinear Programming*, ser. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

[31] D. Gale, *The Theory of Linear Economic Models*. McGraw-Hill, New York, 1960.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep 1995.

[33] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[34] J. Gorski, F. Pfeuffer, and K. Klamroth, "Biconvex sets and optimization with biconvex functions: a survey and extensions," *Math Meth Oper Res*, pp. 373–407, 2007.

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.