Title

Two-Level Lattice Neural Network Architectures for Control of Nonlinear Systems

Authors

Ferlez, James
Sun, Xiaowu
Shoukry, Yasser

Publication Date

2020-12-18

DOI

10.1109/cdc42340.2020.9304079

Two-Level Lattice Neural Network Architectures for Control of Nonlinear Systems

James Ferlez*†, Xiaowu Sun*†, and Yasser Shoukry*

***Abstract*— In this paper, we consider the problem of automatically designing a Rectified Linear Unit (ReLU) Neural Network (NN) architecture (number of layers and number of neurons per layer) with the guarantee that it is sufficiently parametrized to control a nonlinear system. Whereas current state-of-the-art techniques are based on hand-picked architectures or heuristic based search to find such NN architectures, our approach exploits the given model of the system to design an architecture; as a result, we provide a guarantee that the resulting NN architecture is sufficient to implement a controller that satisfies an achievable specification. Our approach exploits two basic ideas. First, assuming that the system can be controlled by an unknown Lipschitz-continuous state-feedback controller with some Lipschitz constant upper-bounded by K_{cont}, we bound the number of affine functions needed to construct a Continuous Piecewise Affine (CPWA) function that can approximate the unknown Lipschitz-continuous controller. Second, we utilize the authors' recent results on a novel NN architecture named as the Two-Level Lattice (TLL) NN architecture, which was shown to be capable of implementing any CPWA function just from the knowledge of the number of affine functions that compromises this CPWA function. We evaluate our method on designing a NN architecture to control an inverted pendulum shows the efficiency of the proposed approach.**

I. Introduction

Multilayer Neural Networks (NN) have shown tremendous success in realizing feedback controllers that can achieve several complex control tasks [1]. Nevertheless, the current state-of-the-art practices for designing these deep NN-based controllers are based on heuristics and hand-picked hyper-parameters (e.g., number of layers, number of neurons per layer, training parameters, training algorithm) without an underlying theory that guides their design. For example, several researchers have studied the problem of Automatic Machine Learning (AutoML) and in particular the problem of hyperparameter (number of layers, number of neurons per layer, and learning algorithm parameters) optimization and tuning in deep NN (see for example [2], [3], [4], [5], [6] and the references within). In this line of work, an iterative and exhaustive search through a manually specified subset of the hyperparameter space is performed. The best hyper-parameters are then selected according to some performance metric without any guarantee on the correctness of the chosen architecture.

In this paper, we focus on the fundamental question of how to systematically choose the NN architecture (number of layers and number of neurons per layer) such that we guarantee the correctness of the chosen NN architecture in terms of its ability to control a nonlinear dynamical system. In particular, we seek to use knowledge of the underlying control problem to guide the design of NN architectures.

Our approach exploits several insights. First, state-of-the-art NN utilizes Rectified Linear Units (ReLU), which in turn restricts the NN controller to implement only Continuous Piecewise Affine (CPWA) functions. As is widely known, CPWA function is compromised of several affine functions (named local linear functions), which are defined over a set of polytypic regions (called local linear regions). In other words, a ReLU NN—by virtue of its CPWA character—partitions its input space into a set of polytypic regions (named activation regions), and applies a linear controller at each of these regions. Therefore, a NN architecture dictates the number of such activation regions in the corresponding CPWA function that is represented by the trainable parameters in the NN. That is, to design a NN architecture, one needs to perform two steps: (i) compute (or upper bound) the number of activation regions required to implement a controller that satisfy the specifications and (ii) transform this number of activation regions into a NN architecture that is guaranteed to give rise to this number of activation regions.

To approach the first step, namely counting the number of the required activation regions, we assume the existence of an unknown robust Lipschitz-continuous, state-feedback controller with some Lipschitz constant upper-bounded by K_{cont} that is capable of controlling the system while meeting the specifications. Without the knowledge of such controller, other than the upper bound on its Lipschitz constant K_{cont}, we can upper-bound the number of activation regions needed to approximate this controller by a CPWA function while still meeting the same specifications.

Next, we build on recent results obtained by the authors on a novel NN architecture named Two-Level Lattice (TLL) NN architecture [7]. Unlike other NN architecture for which the number of activation regions is unknown *a priori*, the TLL-NN architecture enjoys the property that it is parametrized directly by the number of its activation regions. That is, once the number of activation regions is computed using the existence of such an unknown robust Lipschitz-continuous controller, a TLL-NN architecture can be directly generated from this knowledge. Such NN is then guaranteed to be sufficiently parametrized to implement a CPWA function that approximates the unknown Lipschitz-continuous controller, providing a systematic approach to design such architecture for NN controllers.

II. Preliminaries

A. Notation

We will denote by $\mathbb{N}$, $\mathbb{R}$ and $\mathbb{R}^+$ the set of natural numbers, the set of real numbers and the set of non-negative real numbers, respectively. For a function $f : A \to B$, let $\text{dom}(f)$ return the domain of f, and let $\text{range}(f)$ return the range of

† Equally contributing first authors.
*Department of Electrical Engineering and Computer Science, University of California, Irvine {jferlez,xiaowus,yshoukry}@uci.edu
This work was partially sponsored by the NSF awards #CNS-2002405 and #CNS-2013824.

f. For a set $V \in \mathbb{R}^n$, let $\text{int}(V)$ return the interior of V. For $x \in \mathbb{R}^n$, we will denote by $\|x\|$ the infinity norm of x; for $x \in \mathbb{R}^n$ and $\epsilon \geq 0$ we will denote by $B(x;\epsilon)$ the ball of radius ϵ centered at x as specified by $\|\cdot\|$. For $f : \mathbb{R}^n \to \mathbb{R}^m$, $\|f\|_\infty$ will denote the essential supremum norm of f. Finally, given two sets A and B denote by B^A the set of all functions with domain A and range B.

B. Dynamical Model

In this paper, we will consider a continuous-time nonlinear dynamical system specified by the ordinary differential equation (ODE):

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

where the state vector $x(t) \in \mathbb{R}^n$, and the control vector $u(t) \in \mathbb{R}^m$. Formally, we have the following definition:

Definition 1 (Control System)**.** *A **control system** is a tuple $\Sigma = (X, U, \mathcal{U}, f)$ where*

- *$X \subseteq \mathbb{R}^n$ is the compact state space;*
- *$U \subseteq \mathbb{R}^m$ is the compact set of admissible (instantaneous) controls;*
- *$\mathcal{U} \subseteq U^{\mathbb{R}^+}$ is the space of admissible open-loop control functions – i.e. $v \in \mathcal{U}$ is a function $v : \mathbb{R}^+ \to U$; and*
- *$f : \mathbb{R}^n \times U \to \mathbb{R}^n$ is a vector field specifying the time evolution of states according to (1).*

*A control system is said to be (globally) **Lipschitz** if there exists constants K_x and K_u such that for all $x, x' \in \mathbb{R}^n$ and $u, u' \in \mathbb{R}^m$:*

$$\|f(x,u) - f(x',u')\| \leq K_x\|x - x'\| + K_u\|u - u'\|. \tag{2}$$

In the sequel, we will primarily be concerned with solutions to (1) that result from instantaneous state-feedback controllers, $\Psi : X \to U$. Thus, we use $\zeta_{x_0\Psi}$ to denote the *closed-loop* solution of (1) starting from initial condition x_0 (at time $t = 0$) and using *state-feedback controller* Ψ. We refer to such a $\zeta_{x_0\Psi}$ as a (closed-loop) *trajectory* of its associated control system.

Definition 2 (Closed-loop Trajectory)**.** *Let Σ be a Lipschitz control system, and let $\Psi : \mathbb{R}^n \to U$ be a globally Lipschitz continuous function. A **closed-loop trajectory** of Σ under controller Ψ and starting from $x_0 \in X$ is the function $\zeta_{x_0\Psi} : \mathbb{R}^+ \to X$ that uniquely solves the integral equation:*

$$\zeta_{x_0\Psi}(t) = x_0 + \int_0^t f(\zeta_{x_0\Psi}(\sigma), \Psi(\zeta_{x_0\Psi}(\sigma)))d\sigma. \tag{3}$$

It is well known that such solutions exist and are unique under these assumptions [8]. We will only consider feedback controllers for which X is positively invariant under feedback, i.e. range$(\zeta_{x_0\Psi}) \subseteq X$.

For any given feedback controller, Ψ, the open-loop control functions created by its trajectories may not be elements of $\mathcal{U}$. Thus, we make the following additional definition:

Definition 3 (Feedback Controllable)**.** *A Lipschitz control system Σ is **feedback controllable** by a Lipschitz controller $\Psi : \mathbb{R}^n \to U$ if the following is satisfied:*

$$\Psi \circ \zeta_{x\Psi} \in \mathcal{U} \qquad \forall x \in X. \tag{4}$$

A Lipschitz control system is called feedback controllable if it is feedback controllable for each globally Lipschitz feedback controller.

Remark 1. *In this paper, we will henceforth consider only feedback controllable Lipschitz control systems.*

We conclude this subsection by defining the (sampled) transition system embedding of a feedback-controlled system that is inspired by the work in [9].

Definition 4 (τ-sampled Transition System Embedding)**.** *Let $\Sigma = (X, U, \mathcal{U}, f)$ be a feedback controllable Lipschitz control system, and let $\Psi : \mathbb{R}^n \to U$ be a Lipschitz continuous feedback controller. For any $\tau > 0$, the **τ-sampled transition system embedding** of Σ under Ψ is the tuple $S_\tau(\Sigma_\Psi) = (X_\tau, \mathcal{U}_\tau, \xrightarrow[\Sigma_\Psi]{})$ where:*

- *$X_\tau = X$ is the state space;*
- *$\mathcal{U}_\tau = \{(\Psi \circ \zeta_{x_0\Psi})|_{t\in[0,\tau]} : x_0 \in X\}$ is the set of open loop control inputs generated by Ψ-feedback, each restricted to the domain $[0,\tau]$; and*
- *$\xrightarrow[\Sigma_\Psi]{} \subseteq X_\tau \times \mathcal{U}_\tau \times X_\tau$ such that $x \xrightarrow[\Sigma_\Psi]{u} x'$ iff both $u = (\Psi \circ \zeta_{x\Psi})|_{t\in[0,\tau]}$ and $x' = \zeta_{x\Psi}(\tau)$.*

*$S_\tau(\Sigma_\Psi)$ is thus a **metric** transition system [9][1].*

C. Abstract Disturbance Simulation

In this subsection, we propose a new simulation relation, which we call *abstract disturbance simulation*, as a formal notion of specification satisfaction for metric transition systems. Abstract disturbance simulation enforces a notion of specification that is robust to perturbation of the state, and this will facilitate solving the main problem in this paper.

Abstract disturbance simulation is inspired by robust bisimulation [10] and especially disturbance bisimulation [11], but it abstracts those notions away from their definitions in terms of control system embeddings and explicit modeling of disturbance inputs. In this way, it is conceptually similar to the technique used in [9] and [12] to define a quantized abstraction, where deliberate non-determinism is introduced in order to account for input errors. As a prerequisite, we introduce the following definition.

Definition 5 (Perturbed Metric Transition System)**.** *Let $S = (X, U, \xrightarrow{S})$ be a metric transition system where $X \subseteq X_M$ for some metric space (X_M, d). Then the δ-**perturbed metric transition system** of S, $\mathfrak{S}^\delta$, is a tuple $\mathfrak{S}^\delta = (X, U, \xrightarrow{\mathfrak{S}^\delta})$ where the (altered) transition relation, $\xrightarrow{\mathfrak{S}^\delta}$, is defined as follows:*

$$x \xrightarrow{\mathfrak{S}^\delta}{}^{u} x' \text{ iff } \exists x'' \in X \text{ s.t. } d(x'', x') \leq \delta \text{ and } x \xrightarrow{S}{}^{u} x''. \tag{5}$$

Note that $\mathfrak{S}^\delta$ has identical states and input labels to S, and it also subsumes all of the transitions therein, i.e. $\xrightarrow{S} \subset \xrightarrow{\mathfrak{S}^\delta}$. However, the transition relation for $\mathfrak{S}^\delta$ explicitly contains new nondeterminism relative to the transition relation of S. This nondeterminism can be thought of as perturbing the targets state of each transition in S; each such perturbation becomes the target of a (nondeterministic) transition with the same input label as the original transition.

[1]For our purposes, a metric transition system is one whose state space is contained in a metric space.

With this definition in hand, we can finally define an abstract disturbance simulation between two metric transition systems.

Definition 6 (Abstract Disturbance Simulation)**.** *Let $S = (X_S, U, \xrightarrow[S]{})$ and $T = (X_T, U_T, \xrightarrow[T]{})$ be metric transition systems whose state spaces X_S and X_T are subsets of the same metric space (X_M, d). Then T **abstract-disturbance simulates** S under disturbance δ, written $S \preceq_{\mathcal{AD}_\delta} T$ if there is a relation $R \subseteq X_S \times X_T$ such that*

1) *for every $(x, y) \in R$, $d(x, y) \leq \epsilon$;*
2) *for every $x \in X_S$ there exists a pair $(x, y) \in R$; and*
3) *for every $(x, y) \in R$ and $x \xrightarrow[\mathfrak{S}^\delta]{u} x'$ there exists a $y \xrightarrow[T]{v} y'$ such that $(x', y') \in R$.*

Remark 2. *$\preceq_{\mathcal{AD}_0}$ corresponds with the usual notion of simulation for metric transition systems. Thus,*

$$S \preceq_{\mathcal{AD}_\delta} T \Leftrightarrow \mathfrak{S}^\delta \preceq_{\mathcal{AD}_0} T. \tag{6}$$

D. ReLU Neural Network Architectures

In this paper, our primary focus will be on controlling the nonlinear system defined in (1) with a state-feedback neural network controller $\mathscr{NN}$:

$$\mathscr{NN} : X \to U \tag{7}$$

where $\mathscr{NN}$ denotes a Rectified Linear Unit Neural Network (ReLU NN). Such a (K-layer) ReLU NN is specified by composing K *layer* functions (or just *layers*). A layer with $\mathfrak{i}$ inputs and $\mathfrak{o}$ outputs is specified by a $(\mathfrak{o} \times \mathfrak{i})$ real-valued matrix of *weights*, W, and a $(\mathfrak{o} \times 1)$ real-valued matrix of *biases*, b, as follows:

$$\begin{aligned} L_\theta : \mathbb{R}^{\mathfrak{i}} &\to \mathbb{R}^{\mathfrak{o}} \\ z &\mapsto \max\{Wz + b, 0\} \end{aligned} \tag{8}$$

where the max function is taken element-wise, and $\theta \triangleq (W, b)$ for brevity. Thus, a K-layer ReLU NN function as above is specified by K layer functions $\{L_{\theta^{(i)}} : i = 1, \ldots, K\}$ whose input and output dimensions are *composable*: that is they satisfy $\mathfrak{i}_i = \mathfrak{o}_{i-1} : i = 2, \ldots, K$. Specifically:

$$\mathscr{NN}(x) = (L_{\theta^{(K)}} \circ L_{\theta^{(K-1)}} \circ \cdots \circ L_{\theta^{(1)}})(x). \tag{9}$$

When we wish to make the dependence on parameters explicit, we will index a ReLU function $\mathscr{NN}$ by a *list of matrices* $\Theta \triangleq (\theta^{(1)}, \ldots, \theta^{(K)})$ [2].

Specifying the number of layers and the *dimensions* of the associated matrices $\theta^{(i)} = (\ W^{(i)}, b^{(i)}\ )$ specifies the *architecture* of the ReLU NN. Therefore, we will use:

$$\text{Arch}(\Theta) \triangleq ((n, \mathfrak{o}_1), (\mathfrak{i}_2, \mathfrak{o}_2), \ldots, (\mathfrak{i}_{K-1}, \mathfrak{o}_{K-1}), (\mathfrak{i}_K, m)) \tag{10}$$

to denote the architecture of the ReLU NN $\mathscr{NN}_\Theta$.

Since we are interested in designing ReLU architectures, we will also need the following result from [7, Theorem 7], which states that a Continuous, Piecewise Affine (CPWA) function, f, can be implemented exactly using a Two-Level-Lattice (TLL) NN architecture that is parameterized exclusively by the number of local linear functions in f.

[2]That is Θ is not the concatenation of the $\theta^{(i)}$ into a single large matrix, so it preserves information about the sizes of the constituent $\theta^{(i)}$.

Definition 7 (Local Linear Function)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be CPWA. Then a **local linear function of** f is a linear function $\ell : \mathbb{R}^n \to \mathbb{R}^m$ if there exists an open set $\mathfrak{O}$ such that $\ell(x) = f(x)$ for all $x \in \mathfrak{O}$.*

Definition 8 (Linear Region)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be CPWA. Then a **linear region of** f is a the largest set $\mathfrak{R} \subseteq \mathbb{R}^n$ such that f has only one local linear function on the interior of $\mathfrak{R}$.*

Theorem 1 (Two-Level-Lattice (TLL) NN Architecture [9, Theorem 7])**.** *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a CPWA function, and let $\bar{N}$ be an upper bound on the number of local linear functions in f. Then there is a Two-Level-Lattice (TLL) NN architecture Arch$(\Theta^{TLL}_{\bar{N}})$ parameterized by $\bar{N}$ and values of $\Theta^{TLL}_{\bar{N}}$ such that:*

$$f(x) = \mathscr{NN}_{\Theta^{TLL}_{\bar{N}}}(x). \tag{11}$$

In particular, the number of linear regions of f is such an upper bound on the number of local linear functions.

We refer the reader to [7] for more details.

Finally, note that a ReLU NN function, $\mathscr{NN}$, is known to be a continuous, piecewise affine (CPWA) function consisting of finitely many linear segments. Thus, $\mathscr{NN}$ is itself necessarily globally Lipschitz continuous.

III. Problem Formulation

We can now state the main problem we will consider in this paper. In brief, we wish to identify the architecture for a ReLU network to be used as an instantaneous feedback controller for the control system Σ: this architecture must have parameter weights that allow it to control Σ up to a specification that can be met by some other, non-NN controller.

Despite our choice to consider fundamentally continuous-time models, we formulate our main problem in terms of their (τ-sampled) transition system embeddings. This choice reflects recent success in verifying specifications for such transition system embeddings by means of techniques adapted from computer science; see e.g. [13], where a variety of specifications are considered in this context, among them LTL formula satisfaction. Thus, our main problem is stated in terms of the simulation relations in the previous section.

Problem 1. *Let $\delta > 0$ and $K_{cont} > 0$ be given. Let Σ be a feedback controllable Lipschitz control system, and let $S_{spec} = (X_{spec}, U_{spec}, \xrightarrow[S_{spec}]{})$ be a transition system encoding for a specification on Σ. Finally, let $\tau = \tau(f, K_x, K_u, K_{cont}, \delta)$ be determined by the parameters specified.*

Now, suppose that there exists a Lipschitz continuous controller $\Psi : \mathbb{R}^n \to U$ with Lipschitz constant $K_\Psi \leq K_{cont}$ such that:

$$S_\tau(\Sigma_\Psi) \preceq_{\mathcal{AD}_\delta} S_{spec}. \tag{12}$$

Then the problem is to identify a ReLU architecture, Arch(Θ), with the property that there exists values for Θ such that:

$$S_\tau(\Sigma_{\mathscr{NN}_\Theta}) \preceq_{\mathcal{AD}_0} S_{spec}. \tag{13}$$

One of the primary assumptions in Problem 1 is that there exists a controller Ψ which satisfies the specification, S_{spec}. We use this assumption largely to help ensure that the

problem is well posed. For example, this assumption ensures that we aren't trying to assert the existence of NN controller for a system and specification that can't be achieved by *any* continuous controller – such examples are known to exist for nonlinear systems. In this way the existence of a controller Ψ subsumes any possible conditions of this kind that one might wish to impose: stabilizability or controllability for example.

Moreover, there is a strong conceptual reason to consider abstract disturbance simulation in specification satisfaction for such a Ψ. Our approach to solve this problem will be to design a NN architecture that can approximate ***any*** such Ψ sufficiently closely. However, $\mathscr{NN}_\Theta$ clearly belongs to a smaller class of functions than Ψ, so an arbitrary controller Ψ cannot, in general, be represented *exactly* by means of $\mathscr{NN}_\Theta$. This presents an obvious difficulty because instantaneous errors between Ψ and $\mathscr{NN}_\Theta$ may accumulate by means of the system dynamics, i.e. via (3).

IV. ReLU Architectures for Nonlinear Systems

Before we state the main theorem of the paper, we introduce the following notation in the form of two definitions.

Definition 9 (Vector Field Bound, $\mathcal{K}$)**.** *Let:*

$$\mathcal{K} \triangleq \max_{x\in X, u\in U} \|f(x,u)\|, \tag{14}$$

which is well defined because $X \times U$ is compact and f is continuous.

Definition 10 (Extent of X)**.** *The **extent** of the compact set X is defined as:*

$$ext(X) \triangleq \max_{k=1,\dots,n} \left| \max_{x\in X} \pi_k(x) - \min_{x\in X} \pi_k(x) \right|, \tag{15}$$

where $\pi_k(x)$ is the projection of x onto its k^{th} component.

The main result of the paper is the following theorem, which directly solves Problem 1.

Theorem 2 (ReLU Architecture)**.** *Let $\delta > 0$ and $K_{cont} > 0$ be given, and let Σ and S_{spec} be as in the statement of Problem 1. Finally, choose a $\mu > 0$ such that:*

$$K_u \cdot \mu \cdot \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}} \cdot e^{K_x \frac{\mu}{6\cdot K_{cont}\cdot \mathcal{K}}} < \delta, \tag{16}$$

and set:

$$\tau \le \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}} \quad \text{and} \quad \eta \le \frac{\mu}{6 \cdot K_{cont}}, \tag{17}$$

(which depend only on f, K_x, K_u, K_{cont} and δ).

If there exists a Lipschitz continuous controller $\Psi : \mathbb{R}^n \to U$ with Lipschitz constant $K_\Psi \le K_{cont}$ such that:

$$S_\tau(\Sigma_\Psi) \preceq_{\mathcal{AD}_\delta} S_{spec}. \tag{18}$$

Then a TLL NN architecture $Arch(\Theta_N^{TLL})$ of size:

$$N \ge m \cdot \left(n! \cdot \sum_{k=1}^{n} \frac{2^{2k-1}}{(n-k)!} \right) \cdot \left(\frac{ext(X)}{\eta} \right)^n \tag{19}$$

has the property that there exist values for Θ_N^{TLL} such that:

$$S_\tau(\Sigma_{\mathscr{NN}_{\Theta_N^{TLL}}}) \preceq_{\mathcal{AD}_0} S_{spec}. \tag{20}$$

Proof Sketch:
The proof of Theorem 2 consists of establishing the following two implications:

Step 1) *"Approximate controllers satisfy the specification":* There is an approximation accuracy, μ, and sampling period, τ, with the following property: if the unknown controller Ψ satisfies the specification (under δ disturbance and sampling period τ), then any controller – NN or otherwise – which approximates Ψ to accuracy μ will also satisfy the specification (but under no disturbance). This implication is shown in Lemma 2 of Section V.

Step 2) *"Any controller can be approximated by a CPWA with the same fixed number of linear regions":* If unknown controller Ψ has a Lipschitz constant $K_\Psi \le K_{\text{cont}}$, then Ψ can be approximated by a CPWA with a number of regions that depends only on K_{cont} and the desired approximation accuracy. This implication is shown in Lemma 4 of Section VI.

We will show these results for *any* controller Ψ that satisfies the assumptions of Theorem 2. Thus, these results together show the following implication: if there *exists* a controller Ψ that satisfies the assumptions of Theorem 2, then there is a CPWA controller that satisfies the specification. And moreover, this CPWA controller has a at most a number of linear regions that depends only on the parameters of the problem *and not the particular controller* Ψ.

The conclusion of the theorem will then follow directly from Theorem 1 [7, Theorem 7]: together, they specify that any CPWA with the same number of linear regions (or fewer) can be implemented exactly by a common TLL NN architecture. Since this proof is so short given the lemmas described above, it appears in Appendix E of [14].

V. Proof of Theorem 2, Step 1: Approximate Controllers Satisfy the Specification

The goal of this section is to choose constants $\mu > 0$ and $\tau > 0$ such that *any* controller Υ with $\|\Upsilon - \Psi\|_\infty \le \mu/3$ satisfies the specification:

$$S_\tau(\Sigma_\Upsilon) \preceq_{\mathcal{AD}_0} S_{\text{spec}}. \tag{21}$$

The approach will be as follows. First, we confine ourselves to a region in the state space on which the controller Ψ doesn't vary much: the size of this region is determined entirely by the approximation accuracy, μ, and the bound on the Lipschitz constant, K_{cont}. Then we confine the *trajectories* of Σ_Ψ to this region by bounding the duration of those trajectories, i.e. τ. Finally, we feed these results into a Grönwall-type bound to choose μ. In particular, we choose μ small enough such that the error incurred by using Υ instead of Ψ is within the disturbance robustness, δ. From this we will conclude that Υ satisfies the specification as claimed whenever $\|\Upsilon - \Psi\| \le \mu/3$. A more detailed road map of these steps is as follows.

- Let μ be an approximation error. Then:
  - *i)* Choose $\eta = \eta(\mu)$ such that a Lipschitz function with constant K_{cont} doesn't vary by more than $\mu/3$ between any two points that are 2η apart.

ii) Choose $\tau = \tau(\mu)$ such that $\|x - \xi_{xv}(\tau)\| \leq \eta$ for any continuous open-loop control v (use the fact that $\|f\|$ is bounded).

iii) Use a) and b) to conclude that $\|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Psi}(t))\| \leq \|\Upsilon - \Psi\|_\infty + 2\mu/3$ for $t \in [0, \tau]$

iv) Assume $\|\Upsilon - \Psi\|_\infty \leq \mu/3$. Choose $\mu = \mu(\delta)$ such that a Grönwall-type bound satisfies:

$$\|\zeta_{x\Upsilon}(\tau(\mu)) - \zeta_{x\Psi}(\tau(\mu))\| \leq K_u \cdot \mu \cdot \tau(\mu) \cdot e^{K_x \tau(\mu)} < \delta. \quad (22)$$

Conclude that if $\|\Upsilon - \Psi\|_\infty \leq \mu/3$, then:

$$S_{\tau'}(\Sigma_\Upsilon) \preceq_{\mathcal{AD}_0} \mathfrak{S}_{\tau'}(\Sigma_{\Psi'}) \preceq_{\mathcal{AD}_0} S_{\text{spec}}. \quad (23)$$

Now we proceed with the proof. First, we formalize Steps *i)*, *ii)* and *iii)* in the next three propositions, the proofs of which are given in Appendix A - Appendix C of [14].

Proposition 1. *Let $\mu > 0$ be given, and let Ψ be as above. Then there exists an $\eta = \eta(\mu)$ such that:*

$$\|x - x'\| \leq 2\eta \implies \|\Psi(x) - \Psi(x')\| \leq \mu/3. \quad (24)$$

Proposition 2. *Let $\mu > 0$ be given, and let $\eta = \eta(\mu)$ be as in the previous proposition. Finally, let Σ be as specified in the statement of Theorem 2. Then there exists a $\tau = \tau(\mu)$ such that for any Lipschitz feedback controller Υ:*

$$\|x - \zeta_{x\Upsilon}(t)\| \leq \eta = \eta(\mu) \quad \forall t \in [0, \tau]. \quad (25)$$

Proposition 3. *Let $\mu > 0$ be given. Let Σ and Ψ be as in the statement of Theorem 2; let $\eta = \eta(\mu)$ be as in Proposition 1; let $\tau = \tau(\mu)$ be as in Proposition 2; and let $\Upsilon : \mathbb{R}^n \to U$ be a Lipschitz continuous function. Then:*

$$\forall t \in [0, \tau] \quad \|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Psi}(t))\| \leq \|\Upsilon - \Psi\|_\infty + 2\mu/3 \quad (26)$$

To prove Step *iv)* we first need the following two results.

Proposition 4 (Grönwall Bound)**.** *Let Σ and Ψ be as in the statement of Theorem 2, and let Υ be as in the statement of Proposition 3. If:*

$$\|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Psi}(t))\| \leq \kappa \quad \forall t \in [0, \tau] \quad (27)$$

then:

$$\|\zeta_{x\Upsilon}(t) - \zeta_{x\Psi}(t)\| \leq K_u \cdot \kappa \cdot t \cdot e^{K_x t} \quad \forall t \in [0, \tau]. \quad (28)$$

The proof of Proposition 4 appears in Appendix D of [14].

Lemma 1. *Let Σ, Ψ and Υ be as before. Also, suppose that $\mu' > 0$ is such that:*

$$K_u \cdot \mu \cdot \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}} \cdot e^{K_x \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}}} < \delta. \quad (29)$$

If $\|\Upsilon - \Psi\|_\infty \leq \mu/3$, then:

$$\|\zeta_{x\Upsilon}(\tau(\mu)) - \zeta_{x\Psi}(\tau(\mu))\| \leq \delta. \quad (30)$$

Proof. This is a direct consequence of applying Proposition 3 to Proposition 4. □

The final result in this section is the following Lemma.

Lemma 2. *Let Σ, Ψ and Υ be as before, and suppose that $\mu > 0$ is such that:*

$$K_u \cdot \mu \cdot \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}} \cdot e^{K_x \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}}} < \delta. \quad (31)$$

If $\|\Upsilon - \Psi\|_\infty \leq \mu/3$, then for $\tau \leq \frac{\mu}{6 \cdot K_{cont} \cdot \mathcal{K}}$ we have:

$$S_\tau(\Sigma_\Upsilon) \preceq_{\mathcal{AD}_0} \mathfrak{S}_\tau(\Sigma_\Psi). \quad (32)$$

And hence:

$$S_\tau(\Sigma_\Upsilon) \preceq_{\mathcal{AD}_0} S_{spec}. \quad (33)$$

Proof. By definition, $S_\tau(\Sigma_\Upsilon)$ and $\mathfrak{S}_\tau(\Sigma_\Psi)$ have the same state spaces, X. Thus we propose the following as an abstract disturbance simulation under 0 disturbance (i.e. a conventional simulation for metric transition systems):

$$R = \{(x, x) | x \in X\}. \quad (34)$$

Clearly, R satisfies the property that for all $(x, y) \in R$, $d(x, y) \leq 0$, and for every $x \in X$, there exists an $y \in X$ such that $(x, y) \in R$. Thus, it only remains to show the third property of Definition 6 under 0 disturbance.

In particular, let $(x, x) \in R$, and suppose that $x \xrightarrow[\Sigma_\Upsilon]{\Upsilon \circ \zeta_{x\Upsilon}} x' = \zeta_{x\Upsilon}(\tau)$. If $x \xrightarrow[\Sigma_\Psi]{\Psi \circ \zeta_{x\Psi}} x'$, then we have shown that R is an abstract disturbance simulation under 0 disturbance. But by definition, $x \xrightarrow[\Sigma_\Psi]{\Psi \circ \zeta_{x\Psi}} x'' = \zeta_{x\Psi}(\tau)$, and by Lemma 1, $\|x' - x''\| \leq \delta$. Thus, by definition of $\mathfrak{S}_\tau(\Sigma_\Psi)$, $x \xrightarrow[\Sigma_\Psi]{\Psi \circ \zeta_{x\Psi}} x'$ as required. □

VI. Proof of Theorem 2, Step 2: CPWA Approximation of a Controller

The results in Section V showed that any controller, Υ, whether it is CPWA or not, will satisfy the specification if it is close to Ψ in the sense that $\|\Upsilon - \Psi\|_\infty \leq \mu/3$ (where μ is as specified therein). Thus, the main objective of this section will be to show that an arbitrary Ψ can be approximated to this accuracy by a CPWA controller, Υ_{CPWA}, subject to the following caveat. It is well known that CPWA functions are good function approximators in general, but we have to keep in mind our eventual use of Theorem 1: thus, we need to approximate *any such* Ψ by a CPWA with the **same, bounded number of linear regions**. Thus, our objective in this section is to find not just a controller Υ_{CPWA} that approximates Ψ to the specified accuracy, but one that achieves this objective using not more than some common, fixed number of linear regions that depends only on the problem parameters (and not the function Ψ itself which is assumed to be unknown except for a bound on its Lipschitz constant).

With this objective in mind, our strategy will be to partition the state space X into a grid of sup-norm balls such that no Ψ can vary by much between them: indeed we will use balls of size η, as specified in Section V. Thus, we propose the following starting point: inscribe a slightly smaller ball within each η ball of the partition, and choose the value of Υ_{CPWA} on each such ball to be a constant value equal to $\Psi(x)$ for some x therein. Because we have chosen the size of the partition to be small, such an Υ_{CPWA} will still be a good approximation of Ψ for these points in its domain. Using this approach, then, we only have to concern ourselves with how "extend" a function so defined to the entire space X

as a CPWA. Moreover, note that this procedure is actually independent of the particular Ψ chosen, despite appearances: we are basing our construction on a grid size η that depends only on the problem parameters, and the construction will work no matter what particular value $\Psi(x)$ has within each grid square.

The first step in this procedure will be to show how to extend such a function over the largest-dimensional "gaps" between the smaller inscribed balls; the blue region depicted in Fig. 1 is an example of this large-dimensional gap for $X \subset \mathbb{R}^2$ (the notation in the figure will be explained later). This result must control the error of the extension so as to preserve our desired approximation bound, as well provide a count of the number of linear regions necessary to do so; this is Lemma 3. This result can then be extended to all of the other gaps between inscribed balls to yield a CPWA function with domain X, approximation error $\mu/3$, and a known number of regions; this is Lemma 4.

In order to prove our first lemma of this section, we need a couple of definitions to help with the terminology.

Definition 11 (Face)**.** *Let $C = [0,1]^n$ be a unit hypercube of dimension n. A set $F \subseteq C$ is a k-dimensional **face** of C if there exists a set $J \subseteq \{1,\dots,n\}$ such that $|J| = n-k$ and*

$$\forall x \in F \;.\; \bigwedge_{j \in J} \pi_j(x) \in \{0,1\}. \tag{35}$$

Let $\mathscr{F}_k(C)$ denote the set of k-dimensional faces of C, and let $\mathscr{F}(C)$ denote the set of all faces of C (of any dimension).

Remark 3. *A k-dimensional face of the hypercube $C = [0,1]^n$ is isomorphic to the hypercube $[0,1]^k$.*

Definition 12 (Corner)**.** *Let $C = [0,1]^n$. A **corner** of C is a 0-dimensional face of C.*

Lemma 3. *Let $C = [0,1]^n$, and suppose that:*

$$\Gamma_c : \mathscr{F}_0(C) \to \mathbb{R} \tag{36}$$

is a function defined on the corners of C. Then there is a CPWA function $\Gamma : C \to \mathbb{R}$ such that:

- *$\forall x \in \mathscr{F}_0(C).\Gamma(x) = \Gamma_c(x)$, i.e. Γ extends Γ_c to C;*
- *Γ has at most $2^{n-1} \cdot n!$ linear regions; and*
- *for all $x \in C$,*

$$\min_{x \in \mathscr{F}_0(C)} \Gamma_c(x) \le \Gamma(x) \le \max_{x \in \mathscr{F}_0(C)} \Gamma_c(x). \tag{37}$$

Proof. First, we assume without loss of generality that the given function Γ_c takes distinct values on each element of its domain.

This is a proof by induction on dimension. In particular, we will use the following induction hypothesis:

- There is a function $\Gamma_k : \cup_{i=1}^k \mathscr{F}_i(C) \to \mathbb{R}$ such that for all $\tilde{F} \in \mathscr{F}_k(C)$, $\Gamma_k|_{\tilde{F}}$ has the following properties:
  - it is CPWA
  - it has at most $2^{k-1} \cdot k!$ linear regions; and
  - for all $x \in \tilde{F}$:

$$\min_{x \in \mathscr{F}_0(\tilde{F})} \Gamma_c(x) \le \Gamma_k(x) \le \max_{x \in \mathscr{F}_0(\tilde{F})} \Gamma_c(x). \tag{38}$$

We start by showing that if the induction hypothesis above holds for k, then it also holds for $k+1$.

To show the induction step, first note that for any face $F \in \mathscr{F}_{k+1}(C)$, all of *its* faces are already in the domain of Γ_k. That is $\cup_{i=1}^k \mathscr{F}_i(F) \subseteq \mathrm{dom}(\Gamma_k)$. Thus, we can define Γ_{k+1} by extending Γ_k to $\mathrm{int}(F)$ for each $F \in \mathscr{F}_{k+1}(C)$. Since these interiors are mutually disjoint, we can do this by explicit construction on each individually, in such a way that the desired properties hold.

In particular, let $F \in \mathscr{F}_{k+1}(C)$, and let ν be the midpoint of F, i.e. the k-cube isomorphism of ν is $[\frac{1}{2},\dots,\frac{1}{2}]$. ν is clearly in the interior of F, so define:

$$\Gamma_{k+1}(\nu) = \frac{1}{|\mathscr{F}_0(F)|} \sum_{x \in \mathscr{F}_0(F)} \Gamma_k(x) \tag{39}$$

and note that the corners of F are also corners of C Thus, $\Gamma_{k+1}(\nu)$ is the average of all of the corners of the $k+1$-face that contains it. Now, extend Γ_{k+1} to the rest of $\mathrm{int}(F)$ as follows: let $b \in \cup_{i=1}^k \mathscr{F}_i(F)$ and define:

$$\Gamma_{k+1}(\lambda \cdot \nu + (1-\lambda) \cdot b) = \\ \lambda \cdot \Gamma_{k+1}(\nu) + (1-\lambda) \cdot \Gamma_k(b) \quad \forall \lambda \in [0,1]. \tag{40}$$

This definition clearly covers $\mathrm{int}(F)$, and it also satisfies the requirement that:

$$\forall x \in F \quad \min_{x \in \mathscr{F}_0(F)} \Gamma_c(x) \le \Gamma_{k+1}(x) \le \max_{x \in \mathscr{F}_0(F)} \Gamma_c(x) \tag{41}$$

because the induction hypothesis ensures that each b is on a face of F, and the corners of a face of F are a subset of the corners of F. Thus, it remains to show the bound on the number of linear regions. But from the construction, $\Gamma_{k+1}|_F$ has one linear region for linear region of Γ_k on a k-face of F. Since the $k+1$-face F has $2 \cdot (k+1)$ k-faces, we conclude by the induction hypothesis that $\Gamma_{k+1}|_F$ has at most:

$$2 \cdot (k+1) \cdot 2^{k-1} \cdot k! = 2^k \cdot (k+1)! \tag{42}$$

linear regions. This completes the proof of the induction step.

It remains only to show a base case. For this, we select $k = 1$, i.e. the line-segment faces of C. Each 1-face of C has only two corners and no other faces other than itself. Thus, for each $F \in \mathscr{F}_0(C)$ we can simply define $\Gamma_1|_F$ to linearly interpolate between those two corners. $\Gamma_1|_F$ is thus CPWA, and it satisfies the required bounds on its values. Moreover, $\Gamma_1|_F$ has exactly $2^{1-1} \cdot 1! = 1$ linear region. Thus, the function Γ_1 so defined satisfies the induction hypothesis stated above. □

Definition 13 (η-partition)**.** *Let $\eta > 0$ be given. Then an η-partition of X is a regular, non-overlapping grid of $\eta/2$ balls in the* sup *norm that partitions X. Let X_{cent} denote the set of centers of these balls, and let $X_{part} = \{B(x_c;\eta/2) | x_c \in X_{cent}\}$ denote the partition.*

Definition 14 (Neighboring Grid Center/Square)**.** *Let X_{part} be an η-partition of X, and let $B(x_c;\eta/2) \in X_{part}$. Then a **neighboring grid center (resp. square)** to x_c is an $x_c' \in X_c$ (respectively $B(x_c';\eta/2) \in X_{part}$) such that $B(x_c';\eta/2)$ shares a face (of any dimension) with $B(x_c;\eta/2)$. The set of neighbors of a center, x_c, will be denoted by $\mathcal{N}(x_c)$.*

Lemma 4. *Let $\eta = \eta(\mu)$ be chosen as in Proposition 1, and let Ψ be as before. Then there is a CPWA function $\Upsilon_{CPWA} : \mathbb{R}^n \to U$ such that:*

- $\|\Upsilon_{CPWA} - \Psi\|_\infty \leq \frac{\mu}{3}$; *and*
- Υ_{CPWA} *has at most*

$$m \cdot \left(n! \cdot \sum_{k=1}^{n} \frac{2^{2k-1}}{(n-k)!} \right) \cdot \left(\frac{ext(X)}{\eta} \right)^n \tag{43}$$

linear regions.

Proof. Our proof will assume that $U \subseteq \mathbb{R}$, since the extension to $m > 1$ is straightforward from the $m = 1$ case. The basic proof will be to create an η-partition of X, and define Υ_{CPWA} to be constant on $\rho \cdot \eta/2 < \eta/2$ radius balls centered at each of the grid centers in the partition; we will then use Lemma 3 to "extend" this function to the rest of X as a CPWA function. In particular, for each $x_c \in X_C$, we start by defining:

$$\Upsilon_{\text{CPWA}}(x) = \Psi(x_c) \quad \forall x \in B(x_c; \rho \cdot \eta/2). \tag{44}$$

Then we will extend this function to the rest of X, and prove the claims for that extension.

To simplify the proof, we will henceforth focus on a particular x_c, and show how to extend Υ_{CPWA} from $B(x_c; \rho \cdot \eta/2)$ to the "gaps" between it and each of the neighboring balls, $B(x_c'; \rho \cdot \eta/2)$ for $x_c' \in \mathcal{N}(x_c)$. To further simplify the proof, we define here two additional pieces of notation. First, for each $x_c \in X_c$ and each $k \in \{1, \ldots, n\}$ define a function $\omega_k^{(x_c)}$ as follows:

$$\begin{aligned}
&\omega_k^{(x_c)} : \{-1, 0, +1\} \to 2^{\mathbb{R}} \\
&\omega_k^{(x_c)} : \quad 0 \mapsto [\pi_k(x_c) - \rho\tfrac{\eta}{2},\ \pi_k(x_c) + \rho\tfrac{\eta}{2}] \\
&\omega_k^{(x_c)} : +1 \mapsto [\pi_k(x_c) + \rho\tfrac{\eta}{2}, \pi_k(x_c) + \tfrac{\eta}{2} + (1-\rho)\tfrac{\eta}{2}] \\
&\omega_k^{(x_c)} : -1 \mapsto [\pi_k(x_c) - \tfrac{\eta}{2} - (1-\rho)\tfrac{\eta}{2}, \pi_k(x_c) - \rho\tfrac{\eta}{2}].
\end{aligned}$$

Then, define the function:

$$\begin{aligned}
&\mathcal{R}^{(x_c)} : \{-1, 0, 1\}^n \to 2^{\mathbb{R}^n} \\
&\mathcal{R}^{(x_c)} : \iota \mapsto \omega_1^{(x_c)}(\pi_1(\iota)) \times \omega_2^{(x_c)}(\pi_2(\iota)) \times \cdots \times \omega_n^{(x_c)}(\pi_n(\iota)),
\end{aligned}$$

and let $\mathbf{0} \triangleq (0, \ldots, 0) \in \{-1, 0, 1\}^n$. Also, define $\dim(\iota)$ as the number of non-zero elements in ι.

Now let $x_c \in X_c$ be fixed. Using the above notation, the ball $B(x_c; \rho \cdot \eta/2)$ is given by:

$$B(x_c; \rho \cdot \eta/2) = \mathcal{R}^{(x_c)}(\mathbf{0}), \tag{45}$$

Similarly each of the "gaps" between $\mathcal{R}^{(x_c)}(\mathbf{0})$ and its neighbors, $\mathcal{R}^{(x_c')}(\mathbf{0})$ for $x_c' \in \mathcal{N}(x_c)$, are the hypercubes:

$$\mathcal{R}^{(x_c)}(\iota) \text{ for } \iota \in \{-1, 0, 1\}^n \backslash \{\mathbf{0}\}, \tag{46}$$

and hence:

$$\bigcup_{\substack{x_c \in X_c, \\ \iota \in \{-1,0,1\}^n \backslash \mathbf{0}}} \mathcal{R}^{(x_c)}(\iota) = X \backslash \bigcup_{x_c' \in X_c} B(x_c'; \rho \cdot \eta/2). \tag{47}$$

This notation is illustrated in two dimensions in Fig. 1.

The first step is to show that Υ_{CPWA} can be extended from the constant-valued region, $\mathcal{R}^{(x_c)}(\mathbf{0})$, to each of its neighbors, $\mathcal{R}^{(x_c)}(\iota)$, in a consistent way as a CPWA. To do this, first note that $\mathcal{R}^{(x_c)}(\mathbf{0})$ has 2^n neighboring regions with indices $\iota' \in \{-1, +1\}^n$, and each of these regions intersects a different $\mathcal{R}^{(x_c')}(\mathbf{0})$ for $x_c' \in \mathcal{N}(x_c)$ at each corner, but is

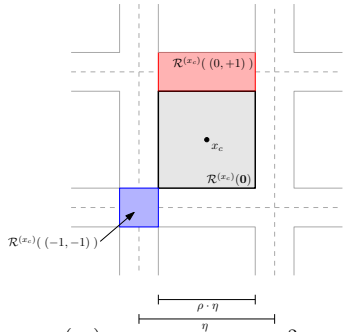

Fig. 1. Illustration of $\mathcal{R}^{(x_c)}$ notation for $X \subset \mathbb{R}^2$. For x_c as labeled, the regions $\mathcal{R}^{(x_c)}((-1,-1))$, $\mathcal{R}^{(x_c)}((0,+1))$ and $\mathcal{R}^{(x_c)}(\mathbf{0})$ are shown in blue, red and light gray, respectively.

otherwise disjoint from them. Thus, Lemma 3 can be used to define a CPWA on each such $\mathcal{R}^{(x_c)}(\iota')$ in a way that is consistent with the definition of Υ_{CPWA} on the $\mathcal{R}^{(x_c')}(\mathbf{0})$. These definitions are also consistent with each other, since these regions are disjoint. Moreover, this procedure yields the same extension when started from $x_c' \in \mathcal{N}(x_c)$ instead of x_c (by the symmetric way that Lemma 3 is proved). Thus, it remains only to define Υ_{CPWA} on regions with indices of the form $\iota'' \in \{-1, 0, +1\}^n \backslash \{-1, +1\}^n \cup \{\mathbf{0}\}$. However, each such $\mathcal{R}^{(x_c)}(\iota'')$ intersects $2^{n-\dim(\iota'')}$ regions with indices of the form $\iota' \in \{-1, +1\}^n$, and each of those intersections is a $\dim(\iota'')$ face of the corresponding $\mathcal{R}^{(x_c)}(\iota')$. But on each such $\dim(\iota'')$ face, Υ_{CPWA} is defined and agrees with $\Gamma_{\dim(\iota'')}$ from the construction in Lemma 3. Finally, since $\Gamma_{\dim(\iota'')}$ (and hence Υ_{CPWA}) is identical up to isomorphism on each of these $\dim(\iota'')$ faces, Υ_{CPWA} can be extended on to $\mathcal{R}^{(x_c)}(\iota'')$ by isomorphism between the $\dim(\iota'')$ nonzero indices, and Υ_{CPWA} as defined on one of the $\dim(\iota'')$ faces of $\mathcal{R}^{(x_c)}(\iota')$. Finally, the symmetry of this procedure and Lemma 3 ensures that this assignment will be consistent when starting from some $x_c' \in \mathcal{N}(x_c)$ instead of x_c.

Next, we show that for this Υ_{CPWA}, $\|\Upsilon_{\text{CPWA}} - \Psi\| \leq \mu/3$. This largely follows from the interpolation property proven in Lemma 3. In particular, on some $\mathcal{R}^{(x_c)}(\iota)$, Υ_{CPWA} takes exactly the same values as some $\Gamma_{\dim(\iota)}$ constructed according to Lemma 3, where the interpolation happens between $\dim(\iota)$ points in $V \triangleq \{\Psi(x_c') | x_c' \in \mathcal{N}(x_c) \cup \{x_c\}\}$. Thus,

$$\forall x \in \mathcal{R}^{(x_c)}(\iota) \quad \min_{y \in V} \Psi(y) \leq \Upsilon_{\text{CPWA}}(x) \leq \max_{y \in V} \Psi(y). \tag{48}$$

Let $x \in \mathcal{R}^{(x_c)}(\iota)$ be fixed temporarily, and suppose that $\Upsilon_{\text{CPWA}}(x) - \Psi(x) \geq 0$ and $\max_{y \in V} \Psi(y) - \Psi(x) \geq 0$. Then:

$$\begin{aligned}
&|\Upsilon_{\text{CPWA}}(x) - \Psi(x)| = \Upsilon_{\text{CPWA}}(x) - \Psi(x) \\
&\leq \max_{y \in V} \Psi(y) - \Psi(x) = |\max_{y \in V} \Psi(y) - \Psi(x)| \leq \frac{\mu}{3}
\end{aligned} \tag{49}$$

where the last inequality follows from our choice of η from Proposition 1, since $|y - x| \leq 2\eta$ for all $y \in V$. The other cases can be considered as necessary, and they lead to the same conclusion. Hence, we conclude $\|\Upsilon_{\text{CPWA}} - \Psi\|_\infty \leq \mu/3$, since our choice of center x_c and ι was arbitrary.

Now we just need to (over)-count the number of linear regions needed in the extension Υ_{CPWA}. This too will follow from the construction in Lemma 3. Note that on each $\mathcal{R}^{(x_c)}(\iota)$, Υ_{CPWA} has the same number of linear regions as some $\Gamma_{\dim(\iota)}$ that was constructed by Lemma 3, which by the same lemma has $2^{\dim(\iota)-1} \cdot \dim(\iota)!$ regions. Thus, we

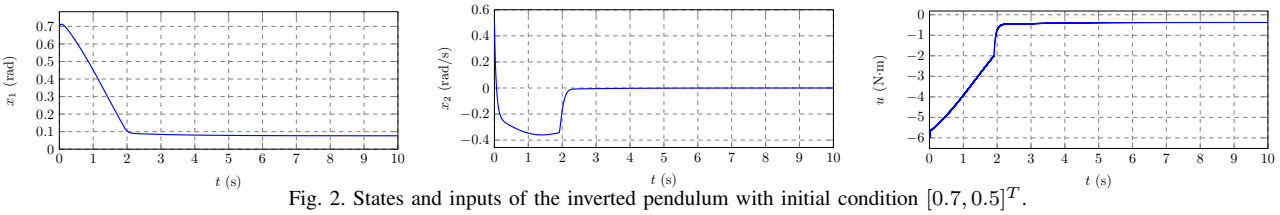

Fig. 2. States and inputs of the inverted pendulum with initial condition $[0.7, 0.5]^T$.

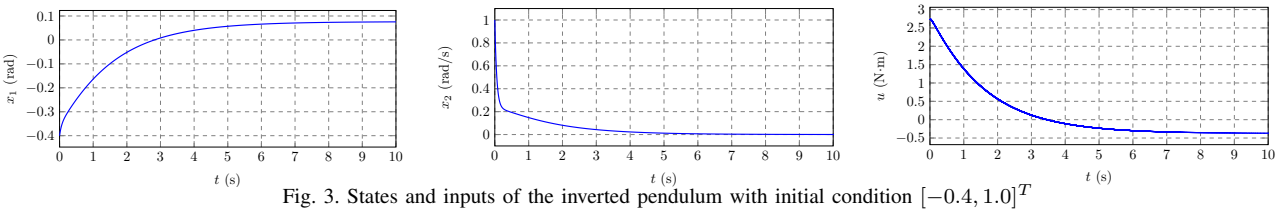

Fig. 3. States and inputs of the inverted pendulum with initial condition $[-0.4, 1.0]^T$

count at most:

$$\sum_{k=1}^{n} \binom{n}{k} \cdot 2^k \cdot 2^{k-1} \cdot k! = n! \cdot \sum_{k=1}^{n} \frac{2^{2k-1}}{(n-k)!} \tag{50}$$

linear regions. Finally, since we need this many regions for the neighboring regions of a single grid square, we obtain an upper bound for the total number of regions by multiplying (50) by the number of grid squares in the partition, $(\frac{\text{ext}(X)}{\eta})^n$ (then by the m, in the multi-dimensional output case). □

VII. Numerical Results

We illustrate the results in this paper on an inverted pendulum described by the following model:

$$f(x_1, x_2, u) = \begin{bmatrix} x_2 \\ \frac{g}{l}\sin(x_1) - \frac{h}{ml^2}x_2 + \frac{1}{ml}\cos(x_1)u \end{bmatrix},$$

where x_1 is the angular position, x_2 is the angular velocity, and control input u is the torque applied on the point mass. The parameters are the rod mass, m; the rod length, l; the (dimensionless) coefficient of rotational friction, h; and the acceleration due to gravity, g. For the purposes of our experiments, we considered a subset of the state/control space specified by: $x_1 \in [-1, 1]$, $x_2 \in [-1, 1]$ and $u \in [-6, 6]$. Furthermore, we considered model parameters: $m = 0.5$ kg; $l = 0.5$ m; $h = 2$; and $g = 9.8$ N/kg. Then for different choices of the design parameters μ, we obtain the following sizes N for the corresponding TLL-NN architecture along with the corresponding τ, η and the δ that is required for the specification satisfaction:

μ	δ	τ	η	N
0.35	0.8694	0.0098	0.583	235
0.3	0.5287	0.0083	0.5	320
0.25	0.3039	0.0069	0.417	460
0.2	0.1610	0.0056	0.334	720
0.15	0.0749	0.0042	0.25	1280
0.1	0.0275	0.0028	0.167	2880

In the sequel, we will show the control performance of a TLL-NN architecture with 400 local linear region. While there are a number of techniques that can be used to train the resulting NN, for the sake of simplicity, we utilize Imitation learning where the NN is trained in a supervised fashion from data collected from an expert controller. In particular, we designed an expert controller that stabilizes the inverted pendulum; we chose to use Pessoa [15] to design our expert using the parameter values specified above. In particular, we tasked Pessoa to design a zero-order-hold controller that stabilizes the inverted pendulum in a subset $X_{\text{spec}} = [-1, 1] \times [-0.5, 0.5]$: that is the controller should transfer the state of the system to this specified set and keep it there for all time thereafter. From this expert controller, we collected 8400 data points of state-action pairs; this data was obtained by uniformly sampling the state space. We then used Keras [16] to train the TLL NN using this data. Finally, we simulated the motion of the inverted pendulum using this TLL NN controller. Shown in Fig. 2 and Fig. 3 are the state and control trajectories for this controller starting from initial state $[0.7, 0.5]$ and $[-0.4, 1]$, respectively. In both, the TLL NN controller met the same specification that was used to design the expert.

References

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv:1604.07316*, 2016.

[2] F. Pedregosa, "Hyperparameter optimization with approximate gradient," *arXiv:1602.02355*, 2016.

[3] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[4] S. Paul, V. Kurin, and S. Whiteson, "Fast efficient hyperparameter tuning for policy gradients," *arXiv:1902.06583*, 2019.

[5] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," *arXiv:1611.02167*, 2016.

[6] Y. Quanming, W. Mengshuo, J. E. Hugo, G. Isabelle, H. Yi-Qi, L. Yu-Feng, T. Wei-Wei, Y. Qiang, and Y. Yang, "Taking human out of learning applications: A survey on automated machine learning," *arXiv:1810.13306*, 2018.

[7] J. Ferlez and Y. Shoukry, "AReN: Assured ReLU NN Architecture for Model Predictive Control of LTI Systems," in *Hybrid Systems: Computation and Control 2020 (HSCC'20)*, ACM, New York, NY USA, 2020.

[8] H. K. Khalil, *Nonlinear Systems*. Pearson, Third ed., 2001.

[9] M. Zamani, G. Pola, M. Mazo, and P. Tabuada, "Symbolic Models for Nonlinear Control Systems Without Stability Assumptions," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, 2012.

[10] V. Kurtz, P. M. Wensing, and H. Lin, "Robust Approximate Simulation for Hierarchical Control of Linear Systems under Disturbances," 2020.

[11] K. Mallik, A.-K. Schmuck, S. Soudjani, and R. Majumdar, "Compositional Synthesis of Finite-State Abstractions," *IEEE Transactions on Automatic Control*, vol. 64, no. 6, pp. 2629–2636, 2019.

[12] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.

[13] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer US, 2009.

[14] J. Ferlez, X. Sun, and Y. Shoukry, "Two-level lattice neural network architectures for control of nonlinear systems." `https://rcpsl.eng.uci.edu/files/2020/03/NNArchitectureTechnicalNote.pdf`, 2020.

[15] M. Mazo, A. Davitian, and P. Tabuada, "PESSOA: A tool for embedded controller synthesis," in *Proceedings of the 22nd International Conference on Computer Aided Verification*, CAV'10, pp. 566–569, Springer-Verlag, 2010.

[16] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

APPENDIX

A. Proof of Proposition 1

Proof. Choose $\eta = \eta(\mu) \leq \frac{\mu}{2 \cdot 3 \cdot K_{\text{cont}}}$ and use Lipschitz continuity of Ψ. □

B. Proof of Proposition 2

Proof. Since f is continuous on the compact set $X \times U$, it is bounded, and let $\mathcal{K}$ be this bound as stated in Definition 9. Then by (3) we have

$$\|x - \zeta_{x\Upsilon}(t)\| = \left\| \int_0^t f(\zeta_{x\Upsilon}(\sigma), \Upsilon(\zeta_{x\Upsilon}(\sigma)))d\sigma \right\| \tag{51}$$

$$\leq \int_0^t \|f(\zeta_{x\Upsilon}(\sigma), \Upsilon(\zeta_{x\Upsilon}(\sigma)))\| \, d\sigma \tag{52}$$

$$\leq \int_0^t \mathcal{K} d\sigma = \mathcal{K}t. \tag{53}$$

Hence, choose $\tau = \tau(\mu) \leq \frac{\eta(\mu)}{\mathcal{K}}$ and the conclusion follows. □

C. Proof of Proposition 3

Proof. By the triangle inequality, we have:

$$\begin{aligned}
&\|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Psi}(t))\| \\
&\quad = \|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Upsilon}(t)) + \Psi(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Psi}(t))\| \\
&\quad = \|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi'(\zeta_{x\Upsilon}(t)) + \\
&\qquad \Psi(\zeta_{x\Upsilon}(t)) - \Psi(x) + \Psi(x) - \Psi(\zeta_{x\Psi}(t))\| \\
&\quad \leq \|\Upsilon(\zeta_{x\Upsilon}(t)) - \Psi(\zeta_{x\Upsilon}(t))\| + \|\Psi(\zeta_{x\Upsilon}(t)) - \Psi(x)\| + \\
&\qquad \|\Psi(x) - \Psi(\zeta_{x\Psi}(t))\|.
\end{aligned} \tag{54}$$

The first term in (54) is bounded by $\|\Upsilon - \Psi\|_\infty$. Now consider the second term. By Proposition 2, $\|\zeta_{x\Upsilon}(t) - x\| \leq \eta$; thus, by Proposition 1 we conclude that $\|\Psi(\zeta_{x\Upsilon}(t)) - \Psi(x)\| \leq \mu/3$. The final term is likewise bounded by $\mu/3$ for the same reasons. Thus, the conclusion follows. □

D. Proof of Proposition 4

Proof. By definition and the properties of the integral, we have:

$$\begin{aligned}
&\|\zeta_{x\Upsilon}(t) - \zeta_{x\Psi}(t)\| \\
&= \| \int_0^t f(\zeta_{x\Upsilon}(\sigma), \Upsilon(\zeta_{x\Upsilon}(\sigma))) - f(\zeta_{x\Psi}(t), \Psi(\zeta_{x\Psi}(t)))d\sigma \| \\
&\leq \int_0^t \|f(\zeta_{x\Upsilon}(\sigma), \Upsilon(\zeta_{x\Upsilon}(\sigma))) - f(\zeta_{x\Psi}(t), \Psi'(\zeta_{x\Psi}(t)))\| d\sigma \\
&\leq \int_0^t K_x \|\zeta_{x\Upsilon}(\sigma) - \zeta_{x\Psi}(\sigma)\| \\
&\qquad + K_u \|\Upsilon(\zeta_{x\Upsilon}(\sigma)) - \Psi(\zeta_{x\Psi}(\sigma))\| d\sigma \\
&\leq \int_0^t K_x \|\zeta_{x\Upsilon}(\sigma) - \zeta_{x\Psi}(\sigma)\| + K_u \cdot \kappa \; d\sigma.
\end{aligned} \tag{55}$$

The claimed bound now follows directly from the Grönwall Inequality [8]. □

E. Proof of Theorem 2

Proof. By Lemma 4, there is a CPWA, Υ_{CPWA} that meets the assumptions of Lemma 2, and whose number of linear regions is upper-bounded by the quantity in (43). Thus, we are done if we can find a TLL NN architecture to implement the CPWA Υ_{CPWA}. But by [7, Theorem 2], just such an architecture can be specified directly by the number of linear regions needed (as in (43)). This completes the proof. □