Lyapunov-stable Neural Control for State and Output Feedback: A Novel Formulation

Lujie Yang[*1] **Hongkai Dai**[*2] **Zhouxing Shi**[3] **Cho-Jui Hsieh**[3] **Russ Tedrake**[1 2] **Huan Zhang**[4]

Abstract

Learning-based neural network (NN) control policies have shown impressive empirical performance in a wide range of tasks in robotics and control. However, formal (Lyapunov) stability guarantees over the region-of-attraction (ROA) for NN controllers with nonlinear dynamical systems are challenging to obtain, and most existing approaches rely on expensive solvers such as sums-of-squares (SOS), mixed-integer programming (MIP), or satisfiability modulo theories (SMT). In this paper, we demonstrate a new framework for learning NN controllers together with Lyapunov certificates using fast empirical falsification and strategic regularizations. We propose a novel formulation that defines a larger verifiable region-of-attraction (ROA) than shown in the literature, and refines the conventional restrictive constraints on Lyapunov derivatives to focus only on certifiable ROAs. The Lyapunov condition is rigorously verified post-hoc using branch-and-bound with scalable linear bound propagation-based NN verification techniques. The approach is efficient and flexible, and the full training and verification procedure is accelerated on GPUs without relying on expensive solvers for SOS, MIP, nor SMT. The flexibility and efficiency of our framework allow us to demonstrate Lyapunov-stable output feedback control with synthesized NN-based controllers and NN-based observers with formal stability guarantees, for the first time in literature. Source code at github.com/Verified-Intelligence/Lyapunov_Stable_NN_Controllers

[*]Equal contribution [1]MIT [2]Toyota Research Institute [3]UCLA [4]UIUC. Correspondence to: Lujie Yang <lujie@mit.edu>, Huan Zhang <huan@huan-zhang.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

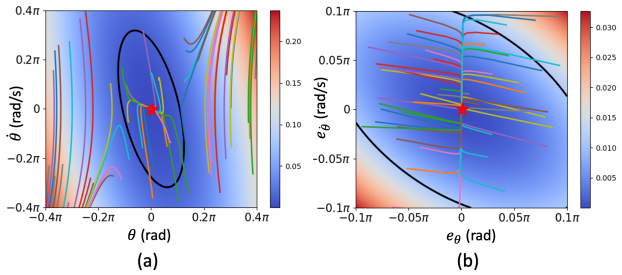

Figure 1: Phase portrait (colorful trajectories) of simulating the angle-observed pendulum with the synthesized neural-network controller and observer in different 2-dimensional slice. The torque limit $|u| \leq \frac{mgl}{3}$ is challenging for both synthesis and verification. The certified ROA is outlined by the black contour. To the best of our knowledge, our method provides the first formal neural certificate for the pendulum with output feedback control.

1. Introduction

Deep learning has significantly advanced the development of neural-network-based controllers for robotic systems, especially those leveraging output feedback from images and sensor data (Kalashnikov et al., 2018; Zhang et al., 2016). Despite their impressive performance, many of these controllers lack the critical stability guarantees that are essential for safety-critical applications. Addressing this issue, Lyapunov stability (Lyapunov, 1892) in control theory offers a robust framework to ensure the closed-loop stability of nonlinear dynamical systems. Central to this theory, a Lyapunov function is a scalar function whose value decreases along the system's closed-loop trajectories, guiding the system towards a stable equilibrium from any states within the region-of-attraction (ROA). While existing research has successfully synthesized *simple* (linear or polynomial) controllers (Tedrake et al., 2010; Majumdar et al., 2013; Yang et al., 2023; Dai & Permenter, 2023) with rigorous stability guarantees, certified by Lyapunov functions and their sublevel sets as region-of-attraction (Slotine et al., 1991) using linear quadratic regulator (LQR) or sum-of-squares (SOS) (Parrilo, 2000) based method, a gap persists in extending these guarantees to more complex neural-network controllers. To bridge this gap, we aim to synthesize neural-network controllers with Lyapunov functions, in order to

certify the stability of the closed-loop system with either state or output feedback.

Many recent works have considered searching for Lyapunov or barrier functions using sampled data to guide the synthesis of neural-network controllers (Dawson et al., 2022; Jin et al., 2020; Liu et al., 2023; Sun et al., 2021). Although empirically successful in a diverse range of tasks, they do not yet provide formal guarantees. In contrast, other research (Dai et al., 2021; Wu et al., 2023; Everett et al., 2021; Vincent & Schwager, 2022) focuses on the rigorous certification, which is grounded in formal methods (Liu et al., 2021; Edwards et al., 2023), with tools such as Satisfiable Modulo Theories (SMT) (Chang et al., 2019; Abate et al., 2020), Mixed-integer Programming (MIP) (Dai et al., 2021; Chen et al., 2021) or Semidefinite Programming (SDP) (Wang et al., 2023; Yin et al., 2021; Fazlyab et al., 2020). These formal methods formulate the Lyapunov certification problem as proving that certain functions (the Lyapunov function itself, together with the negation of its time derivative) are always non-negative over a domain. The state-of-the-art SMT solvers (Gao et al., 2013; De Moura & Bjørner, 2008) become limited by the complexity of the functions they can certify, especially when the controller, dynamics, sensor output function, observer, and the Lyapunov function intertwine. Consequently, the SMT-based approaches only synthesized simple controllers (Chang et al., 2019). On the other hand, MIP solvers (Bertsimas & Tsitsiklis, 1997) employ a branch-and-bound process and divide the verification problem into linear subproblems. This approach has better scalability to higher dimensional systems with neural-network controllers (Dai et al., 2021), with the limitation of requiring the original system dynamics to be approximated as piecewise linear functions; hence, it cannot handle generic nonlinear dynamical systems. Due to these limitations in scalability, previous neural-network Lyapunov control works predominantly provided guarantees only for state-feedback control. Our work addresses the more challenging but practically relevant domain of output-feedback control, identifying and overcoming the limitations of existing methods to synthesize and certify controllers for real-world applications.

In addition to relying on resource-intensive solvers for SMT, MIP or SDP, prior works on neural certificates (Chang et al., 2019; Dai et al., 2021; Wu et al., 2023) imposed the Lyapunov derivative constraint across an entire explicitly defined region, rather than the implicitly defined region-of-attraction. This results in unnecessarily restrictive conditions over uncertified regions. Moreover, all of them failed to find the largest certifiable ROA by applying incorrect restrictions on the ROA. We remedy these issues with a new formulation in Sec. 3.2 that eliminates the overly stringent constraints on the Lyapunov time-derivative over uncertifiable regions.

To achieve the ambitious goal of synthesizing Lyapunov-stable neural control for general nonlinear dynamical systems with both state and output feedback, our work utilizes the latest progress from the neural network verification community. Recently, α,β-CROWN (Zhang et al., 2018; Xu et al., 2020a;b; Wang et al., 2021; Zhang et al., 2022; Shi et al., 2023) demonstrated great scalability in robustness verification of large-scale computer vision neural networks and safety verification of neural-network controllers (Everett et al., 2023; Mathiesen et al., 2022; Rober et al., 2023; Kotha et al., 2024). This complete verifier has a few distinct features that are specialized for verifying NN-controlled systems. First, it *exploits the network structure* of the underlying verification problem by efficiently propagating linear bounds through neural networks; in contrast, general-purpose MIP or SMT solvers do not effectively exploit the rich NN structure. Second, the bound propagation process is *GPU-friendly*, allowing the efficient verification of large networks and the fast evaluation of many subproblems using branch-and-bound.

Our key contributions include:

- We synthesize and verify neural-network *controllers*, *observers* together with Lyapunov functions for *general nonlinear* dynamical systems. To the best of our knowledge, this is the first work to achieve this goal with formal guarantees.
- We propose a novel formulation that defines a large certifiable region-of-attraction (see Fig. 1) and removes the unnecessarily restrictive Lyapunov time-derivative constraints in uncertified regions. Compared with previous works, our new formulation is easier to train and certify, while affording control over the growth of the ROA during training.
- Unlike previous work with formal guarantees (Dai et al., 2021; Chang et al., 2019), which guided training with expensive verifiers like SMT or MIP, we show that cheap adversarial attacks with strategic regularization are sufficient to guide the learning process and achieve a *certified* ROA via *post-training* verification using a strong verifier.

The paper is organized as follows. In Sec.2, we discuss the problem formulation and our parameterization of the controllers/observers using NNs. In Sec.3, we present our new formulation to verify Lyapunov stability and our new training algorithm to synthesize controllers. In Sec.4, we demonstrate that our novel formulation leads to larger ROAs compared to the state-of-the-art approaches in multiple dynamical systems. For the first time in literature, we present verified neural network controllers and observers for pendulum and 2D quadrotor with *output feedback* control.

2. Problem Statement

We consider a nonlinear discrete-time plant

$$x_{t+1} = f(x_t, u_t) \tag{1a}$$

$$y_t = h(x_t) \tag{1b}$$

where $x_t \in \mathbb{R}^{n_x}$ is the state, $u_t \in \{u | u_{\text{lo}} \leq u \leq u_{\text{up}}\} \subset \mathbb{R}^{n_u}$ is the control input, and $y_t \in \mathbb{R}^{n_y}$ is the system output. We denote the goal state/control at equilibrium as x^*/u^* and assume that f is continuous.

Our objective is to jointly search for a Lyapunov function and a neural-network control policy (together with a neural-network state observer for output feedback scenarios) to formally certify the Lyapunov stability of the closed-loop system. Moreover, we aim to train the policy that maximizes the region-of-attraction (ROA) for the closed-loop system and certify its inner-approximation. We will first introduce our parameterization of the policy and the state observer, and then specify our training and verification goal.

State feedback control. In this scenario, the controller has full access to the accurate state x_t. We parameterize the control policy with a neural network $\phi_\pi : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ as

$$u_t = \pi(x_t) = \text{clamp}\left(\phi_\pi(x_t) - \phi_\pi(x^*) + u^*, u_{\text{lo}}, u_{\text{up}}\right). \tag{2}$$

By construction, this control policy $\pi(\bullet)$ produces the goal control u^* at the goal state x^*.

Output feedback control. In the output feedback setting, the controller does not have access to the true state x_t but rather only observes the output y_t. The output can be either a subset of the state or, more generally, a nonlinear function of x_t. In this paper, we consider the situation where there are only uncertainties in the initial conditions. We aim to estimate the state as $\hat{x}_t$ with a dynamic state observer using a neural network $\phi_{\text{obs}} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_x}$ as

$$\hat{x}_{t+1} = f(\hat{x}_t, u_t) + \phi_{\text{obs}}(\hat{x}_t, y_t - h(\hat{x}_t)) - \phi_{\text{obs}}(\hat{x}_t, \mathbf{0}_{n_y}), \tag{3}$$

where $\mathbf{0}_{n_y} \in \mathbb{R}^{n_y}$ is a vector of all 0s. Notice that this state observer resembles the Luenberger observer (Luenberger, 1971), where the observer gain is replaced by the neural network ϕ_{obs}. By construction, if $\hat{x}_t = x_t$, then our observer ensures that $\hat{x}_{t+1} = x_{t+1}$. The network $\phi_\pi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_u}$ now takes in both the state estimate $\hat{x}_t$ and output y_t rather than the true state x_t

$$\begin{aligned} u_t &= \pi(\hat{x}_t, y_t) \\ &= \text{clamp}\left(\phi_\pi(\hat{x}_t, y_t) - \phi_\pi(x^*, h(x^*)) + u^*, u_{\text{lo}}, u_{\text{up}}\right). \end{aligned} \tag{4}$$

Unlike linear dynamical systems whose optimal output feedback controller only depends on the estimated state $\hat{x}_t$ (i.e., the separation principle (Åström, 2012; Athans, 1971)), we expand the design of our neural-network controller to depend on both $\hat{x}_t$ and y for the nonlinear dynamical systems. By also incorporating the output y_t, we enable the controller to distinguish and appropriately react to different actual states x_t that may correspond to the same state estimate. We find this controller design to be sufficient for all our experiments.

Unifying state and output feedback notation. To unify the design for both state and output feedback control and simplify the notation, we introduce an internal state $\xi_t \in \mathbb{R}^{n_\xi}$ with the closed-loop dynamics

$$\xi_{t+1} = f_{\text{cl}}(\xi_t). \tag{5}$$

For state feedback, the internal state is simply the true state $\xi_t = x_t$ and the closed-loop dynamics is

$$f_{\text{cl}}(\xi_t) = f(\xi_t, \pi(\xi_t)). \tag{6}$$

For output feedback, we define the state prediction error $e_t = \hat{x}_t - x_t$, whose value at the equilibrium is required to be $e^* \equiv \mathbf{0}_{n_x}$. The internal state is defined as $\xi_t = [x_t,\ e_t]^\top$ with closed-loop dynamics

$$f_{\text{cl}}(\xi_t) = \begin{bmatrix} f(x_t, \pi(\hat{x}_t, h(x_t))) \\ f(\hat{x}_t, \pi(\hat{x}_t, h(x_t))) + g(x_t, \hat{x}_t) - x_t \end{bmatrix} \tag{7a}$$

$$g(x_t, \hat{x}_t) = \phi_{\text{obs}}(\hat{x}_t, h(x_t) - h(\hat{x}_t)) - \phi_{\text{obs}}(\hat{x}_t, \mathbf{0}_{n_y}). \tag{7b}$$

Definition 2.1 (region-of-attraction)**.** The region-of-attraction for an equilibrium state ξ^* is the largest invariant set $\mathcal{R}$ such that under the closed-loop dynamics (5), $\lim_{t\to\infty} \xi_t = \xi^*$ for all $\xi_0 \in \mathcal{R}$.

Training and verification goal. Formally, we aim at finding a Lyapunov function $V(\xi_t) : \mathbb{R}^{n_\xi} \to \mathbb{R}$, and an invariant and bounded set $\mathcal{S}$ that contains the goal state at the equilibrium ξ^* as a certified inner-approximation of the ROA $\mathcal{R}$. Our objective is formalized in the optimization problem

$$\max_{V,\pi,\phi_{\text{obs}}} \ \text{Vol}(\mathcal{S}) \tag{8a}$$

$$\text{s.t } V(\xi_t) > 0 \ \forall \xi_t \neq \xi^* \in \mathcal{S} \tag{8b}$$

$$V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \ \forall \xi_t \in \mathcal{S} \tag{8c}$$

$$V(\xi^*) = 0, \tag{8d}$$

where $\kappa > 0$ is a fixed constant for exponential stability convergence rate. Constraints (8b)-(8d) guarantee that trajectories originating from any state within $\mathcal{S}$ will eventually converge to the goal state ξ^*. Hence, $\mathcal{S}$ is an inner-approximation of the ROA. Our subsequent efforts will focus on expanding this set $\mathcal{S}$ for broader stability guarantees.

3. Methodology

Previous works on verified neural certificates (Chang et al., 2019; Dai et al., 2021; Wu et al., 2023) enforced overly restrictive Lyapunov derivative constraints in an entire explicitly defined region, and failed to find the largest verifiable ROA. In this section, we present our new formulation that defines a larger certifiable ROA and removes these constraints outside the ROA. We then discuss our verification and training algorithms to generate stabilizing controllers and observers together with Lyapunov certificates.

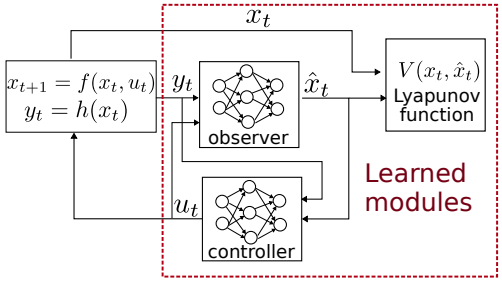

Figure 2: Given a dynamical system, we find an observer, a controller, and a Lyapunov function (parametrized by functions such as NNs), to prove the stability of the closed-loop system with a large certified region-of-attraction.

3.1. Design of learnable Lyapunov functions

To enforce the positive definite condition (8b), we adopt two types of parameterizations for the Lyapunov function. We construct the Lyapunov function using either 1) a neural network $\phi_V : \mathbb{R}^{n_\xi} \to \mathbb{R}$ as

$$V(\xi_t) = |\phi_V(\xi_t) - \phi_V(\xi^*)| + \|(\epsilon I + R^T R)(\xi_t - \xi^*)\|_1, \quad (9)$$

or 2) a quadratic function

$$V(\xi_t) = (\xi_t - \xi^*)^T (\epsilon I + R^T R)(\xi_t - \xi^*), \quad (10)$$

where ϵ is a small positive scalar and $R \in \mathbb{R}^{n_\xi \times n_\xi}$ is a learnable matrix parameter to be optimized. Notice that since $\epsilon I + R^T R$ is a strictly positive definite matrix, the term $|(\epsilon I + R^T R)(\xi_t - \xi^*)|_1$ or $(\xi_t - \xi^*)^T (\epsilon I + R^T R)(\xi_t - \xi^*)$ guarantees the Lyapunov candidate to be strictly positive when $\xi_t \neq \xi^*$. Also, by construction, the Lyapunov candidates (9) and (10) satisfy $V(\xi^*) = 0$ (condition (8d)). We illustrate our entire system diagram in Fig. 2.

3.2. A Novel Verification Formulation

Our verifier α,β-CROWN, along with others like dReal, can verify statements such as $V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t)$, over an *explicitly* defined region. Therefore, we choose a compact "region-of-interest" $\mathcal{B}$ (e.g., a box) containing the equilibrium state ξ^*, and constrain $\mathcal{S}$ as the intersection of $\mathcal{B}$ and a sublevel set of V as

$$\mathcal{S} := \{\xi_t \in \mathcal{B} | V(\xi_t) < \rho\}, \quad (11)$$

where ρ ensures

$$\xi_{t+1} \in \mathcal{B} \quad \forall \xi_t \in \mathcal{S}. \quad (12)$$

Proposition 3.1. *If $\xi_t \in \mathcal{S}$, then $\xi_{t+1} \in \mathcal{S}$. Moreover, $\mathcal{S} \subseteq \mathcal{R}$.*

Proof. For any ξ_t in $\mathcal{S}$, we have that $V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \leq 0$ by (8c) and therefore $V(\xi_{t+1}) \leq V(\xi_t) < \rho$. By (12), we also know that $\xi_{t+1} \in \mathcal{B}$. Therefore, we have that $\xi_{t+1} \in \mathcal{S}$ and $\mathcal{S}$ is invariant. □

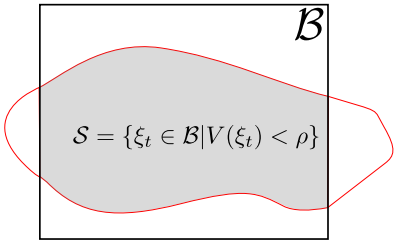

Figure 3: We choose a "region-of-interest" $\mathcal{B}$. The invariant set $\mathcal{S}$ (the shaded region) is the intersection of the sub-level set $\{\xi_t | V(\xi_t) < \rho\}$ (the red curve) and the region $\mathcal{B}$. ρ is chosen such that $\xi_{t+1} \in \mathcal{B}\ \forall \xi_t \in \mathcal{S}$.

Fig. 3 illustrates the region of interest $\mathcal{B}$, and the invariant set $\mathcal{S}$. Taking $\mathcal{S}$ to be the intersection of the sublevel set and $\mathcal{B}$ is particularly important here, since smaller $\mathcal{B}$ with relatively large $\mathcal{S}$ reduces the burden on our α,β-CROWN verifier compared to larger $\mathcal{B}$ with relatively small $\mathcal{S}$.

Drawbacks of existing approaches. Many complete verifiers, including α,β-CROWN and MIP solvers, do not directly handle verification over an *implicitly* defined region, such as our invariant set $\mathcal{S}$ that depends on V. To get around this limitation and enforce the derivative condition (8c), previous works (Chang et al., 2019; Dai et al., 2021; Wu et al., 2023) typically adopt a two-step approach to obtain the region-of-attraction. In step 1, they synthesize and verify the Lyapunov derivative condition over the entire explicitly defined domain

$$V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \ \ \forall \xi_t \in \mathcal{B}. \quad (13)$$

In step 2, they compute the largest sublevel set *within* $\mathcal{B}$, denoted by $\tilde{\mathcal{S}} := \{\xi_t \in \mathcal{B} | V(\xi_t) < \min_{\bar{\xi}_t \in \partial\mathcal{B}} V(\bar{\xi}_t)\}$, as the certified ROA. The drawback of this two-step approach is twofold: 1) the Lyapunov derivative condition is unnecessarily certified over $\mathcal{S}^c \cap \mathcal{B}$. This region, represented as the unshaded region *outside* of $\mathcal{S}$ and within $\mathcal{B}$ in Fig. 3, is not invariant. This means states initiating from this region have no guarantees on stability or convergence. 2) Their certified ROA $\tilde{\mathcal{S}}$ is not guaranteed to be invariant and is much smaller than the largest possible $\mathcal{S}$ by construction. Consequently, this two-step approach makes the synthesis and verification unnecessarily hard, significantly reducing the size of the certified ROA.

Example 3.2. *Consider a 2-dimensional single integrator plant $x_{t+1} = x_t + 0.1 \cdot u_t$ with the controller $\pi(x_t) = -x_t$ and Lyapunov function $V(x_t) = x_t^T x_t$. Let $\mathcal{B} = [-1, 1]^2$ and $\kappa = 0.1$, the Lyapunov derivative condition is satisfied on the entire set $\mathcal{B}$. Moreover, since the closed-loop dynamics lead to $x_{t+1} = (1 - 0.1) \cdot x_t \in \mathcal{B}, \forall x_t \in \mathcal{B}$, we have that $\mathcal{S} = \mathcal{B}$. However, previous works can only find the largest sublevel set as the circle $\tilde{\mathcal{S}} = \{x_t | x_t^T x_t \leq 1\}$, which is strictly contained in $\mathcal{B}$.*

A new formulation for verifying ROA. To overcome the limitations of existing approaches, we describe how to reformulate the derivative condition (8c), originally defined over $\mathcal{S}$, to be verified over the explicitly defined region $\mathcal{B}$.

Theorem 3.3. *Let $F(\xi_t) := V(f_{cl}(\xi_t)) - (1-\kappa)V(\xi_t)$. If the condition*

$$(-F(\xi_t) \geq 0 \wedge \xi_{t+1} \in \mathcal{B}) \vee (V(\xi_t) \geq \rho),\ \forall \xi_t \in \mathcal{B} \quad (14)$$

holds, then the closed-loop system (5) is Lyapunov stable with $\mathcal{S}$ as the certified ROA.

Namely a point $\xi_t \in \mathcal{B}$ either satisfies the Lyapunov function value decreasing and will stay in $\mathcal{B}$ at the next time step, or is outside of the certified ROA $\mathcal{S}$. Adding the condition $V(\xi_t) \geq \rho$ makes verification *easier*, as the verifier only needs to check the Lyapunov derivative condition when ξ_t is within the sublevel set $V(\xi_t) < \rho$.

Verification with α,β-CROWN. The verification problem (14) is presented as a general computation graph to α,β-CROWN, and we extended the verifier to support all nonlinear operations in our system, such as trigonometric functions. Initially, α,β-CROWN uses an efficient bound propagation method (Zhang et al., 2018) to lower bound $-F(\xi_t)$ and $V(\xi_t) - \rho$ on $\mathcal{B}$; if one of the lower bound is nonnegative, (14) is verified. Otherwise, α,β-CROWN conducts branch-and-bound: it splits $\mathcal{B}$ into smaller regions by cutting each dimension of $\mathcal{B}$ and solving verification subproblems in each subspace. The lower bounds tend to be tighter after branching, and (14) is verified when all subproblems are verified. We modified the branching heuristic on $\mathcal{B}$ to encourage branching at ξ^*, since $F(\xi_t)$ tends to be 0 around ξ^*, and tighter lower bounds are required to prove its positiveness. Compared to existing verifiers for neural Lyapunov certificates (Chang et al., 2019; Dai et al., 2021), the efficient and GPU-friendly bound propagation procedure in α,β-CROWN that exploits the structure of the verification problem is the key enabler to solving the difficult problem presented in (14). We can use bisection to find the largest sublevel set value $\rho_{\max}$ that satisfies (14). Our verification algorithm is outlined in 1.

3.3. Training Formulation

We adopt a new *single-step* approach that can directly synthesize and verify the ROA. We define $H(\xi_{t+1})$ as the violation of ξ_{t+1} staying within $\mathcal{B}$, which is positive for $\xi_{t+1} \notin \mathcal{B}$ and 0 otherwise. Mathematically, for an axis-aligned bounding box $\mathcal{B} = \{\xi | \xi_{\text{lo}} \leq \xi \leq \xi_{\text{up}}\}$,

$$H(\xi_{t+1}) = \|\text{ReLU}(\xi_{t+1} - \xi_{\text{up}})\|_1 + \|\text{ReLU}(\xi_{\text{lo}} - \xi_{t+1})\|_1. \quad (15)$$

Theorem 3.4. *The following conditions are necessary and sufficient for each other:*

$$(F(\xi_t) \leq 0) \wedge (H(\xi_{t+1}) \leq 0)\ \forall \xi_t \in \mathcal{S} \Leftrightarrow \quad (16a)$$

Algorithm 1 Lyapunov-stable Neural Control Verification

Input: neural-network controller π, observer network ϕ_{obs}, Lyapunov function V, sublevel set value estimate $\hat{\rho}_{\max}$ (possibly from training), scaling factor λ, convergence tolerance tol
Output: certified sublevel set value $\rho_{\max}$
// Find initial bounds ρ_{lo} and ρ_{up} for bisection
Verify (14) with $(\pi, \phi_{\text{obs}}, V, \hat{\rho}_{\max})$ via α,β-CROWN
if verified **then**
  $\rho_{\text{lo}} = \hat{\rho}_{\max}$
  $\rho_{\text{up}} =$ multiply $\hat{\rho}_{\max}$ by λ until verification fails
else
  $\rho_{\text{up}} = \hat{\rho}_{\max}$
  $\rho_{\text{lo}} =$ divide $\hat{\rho}_{\max}$ by λ until verification succeeds
end if
// Bisection to find ρ_{max}
while $\rho_{\text{up}} - \rho_{\text{lo}} > tol$ **do**
  $\rho_{\max} \leftarrow \frac{\rho_{\text{lo}} + \rho_{\text{up}}}{2}$
  Verify (14) with $(\pi, \phi_{\text{obs}}, V, \rho_{\max})$ via α,β-CROWN
  if verified **then**
    $\rho_{\text{lo}} \leftarrow \rho_{\max}$
  else
    $\rho_{\text{up}} \leftarrow \rho_{\max}$
  end if
end while

$$\min(\textit{ReLU}\,(F(\xi_t)) + c_0 H(\xi_{t+1}),\ \rho - V(\xi_t)) \leq 0\ \forall \xi_t \in \mathcal{B}. \quad (16b)$$

Here, $c_0 > 0$ balances the violations of Lyapunov derivative condition and set invariance during training. The condition $H(\xi_{t+1}) \leq 0$ ensures $\mathcal{S}$ is invariant. Now we can synthesize the Lyapunov function and the controller satisfying condition (16b) over the explicitly defined domain $\mathcal{B}$.[1] We define the violation on (16b) as

$$L_{\dot{V}}(\xi_t; \rho) = \text{ReLU}(\min(\text{ReLU}(F(\xi_t)) + c_0 H(\xi_{t+1}), \rho - V(\xi_t))). \quad (17)$$

The objective function (8a) aims at maximizing the volume of $\mathcal{S}$. Unfortunately, the volume of this set cannot be computed in closed form. Hence, we seek a surrogate function that, when optimized, indirectly expands the volume of $\mathcal{S}$. Specifically, we select some candidate states $\xi^{(i)}_{\text{candidate}}, i = 1, \ldots, n_{\text{candidate}}$ that we wish to stabilize with our controller. The controller and Lyapunov function are optimized to cover $\xi^{(i)}_{\text{candidate}}$ with $\mathcal{S}$, i.e., $V(\xi^{(i)}_{\text{candidate}}) < \rho$.

[1] To enforce (8c) in polynomial optimization, the S-procedure (Pólik & Terlaky, 2007) from control theory employs finite-degree Lagrange multipliers to decide whether a polynomial inequality $V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t)$ is satisfied over an invariant semi-algebraic set $\{\xi_t | V(\xi_t) < \rho\}$. In contrast, we can directly enforce (16b) thanks to the flexibility of α,β-CROWN.

Formally we choose to minimize this surrogate function

$$L_{\text{roa}}(\rho) = \sum_{i=1}^{n_{\text{candidate}}} \text{ReLU}\left(\frac{V(\xi_{\text{candidate}}^{(i)})}{\rho} - 1\right) \tag{18}$$

in place of maximizing the volume of $\mathcal{S}$ as in (8a). By carefully selecting the candidate states $\xi_{\text{candidate}}^{(i)}$, we can control the growth of the ROA. We discuss our strategy to select the candidates in more detail in Appendix B.1.

3.4. Training Controller, Observer and Lyapunov Function

We denote the parameters being searched during training as θ, including:

- The weights/biases in the controller network ϕ_π;
- (NN Lyapunov function only) The weights/biases in the Lyapunov network ϕ_V;
- The matrix R in (9) or (10).
- (Output feedback only) The weights/biases in the observer network ϕ_{obs}.

Mathematically we solve the problem (8) through optimizing θ in the following program

$$\min_\theta \text{ objective (18)} \tag{19a}$$

$$\text{s.t constraint (16b)}, \tag{19b}$$

where (8b) and (8d) are satisfied by construction of the Lyapunov function. Note that the constraint (16b) should hold for infinitely many $\xi_t \in \mathcal{B}$. To make this infinite-dimensional problem tractable, we adopt the Counter Example Guided Inductive Synthesis (CEGIS) framework (Abate et al., 2018; Dai et al.; Ravanbakhsh & Sankaranarayanan), which treats the problem (19) as a bi-level optimization problem. In essence, the CEGIS framework follows an iterative process. During each iteration,

a. *Inner problem*: it finds counterexamples ξ_{adv}^i by maximizing (17) .
b. *Outer problem*: it refines the parameters θ by minimizing a surrogate loss function across all accumulated (and hence, finitely many) counterexamples ξ_{adv}^i.

This framework has been widely used in previous works to synthesize Lyapunov or safety certificates (Chang et al., 2019; Dai et al., 2021; Abate et al., 2018; Ravanbakhsh & Sankaranarayanan). However, a distinct characteristic of our approach for complex systems is the avoidance of resource-intensive verifiers to find the worst case counterexamples. Instead, we use cheap projected gradient descent (PGD) (Madry et al., 2017) to find counterexamples that violate (16b). We outline our training algorithm in 2.

CEGIS within $\mathcal{S}$. A major distinction compared to many CEGIS-based approaches is in line 12 and 16, where $L_{\dot{V}}$ only enforces the Lyapunov derivative constraint inside the certifiable ROA which depends on ρ. To encourage the sublevel set in (11) to grow beyond $\mathcal{B}$, we parameterize $\rho = \gamma \cdot \min_{\bar{\xi}_j \in \partial\mathcal{B}} V(\bar{\xi}_j)$ with the scaling factor $\gamma > 1$. The largest γ that leads to $\hat{\rho}_{\max}$ can be found using bisection. In line 5−9, we sample many points $\bar{\xi}_j$ on the periphery of $\mathcal{B}$, and apply PGD to minimize $V(\bar{\xi}_j)$. In line 10−14, we apply PGD again to maximize the violation (17) over randomly sampled $\xi_{\text{adv}}^i \in \mathcal{B}$ to generate a set of counterexamples in the training set $\mathcal{D}$. To make the training more tractable, we start with a small $\mathcal{B}$ and gradually grow its size to cover the entire region we are interested in.

Algorithm 2 Training Lyapunov-stable Neural Controllers

1: **Input:** plant dynamics f and h, region-of-interest $\mathcal{B}$, scaling factor γ, PGD stepsizes α and β, learning rate η
2: **Output:** Lyapunov candidate V, controller π, observer ϕ_{obs} all in θ
3: Training dataset $\mathcal{D} = \varnothing$
4: **for** $iter = 1, 2, \cdots$ **do**
5: Sample points $\bar{\xi}_j \in \partial\mathcal{B}$
6: **for** $rho_descent = 1, 2, \cdots$ **do**
7: $\bar{\xi}_j \leftarrow \text{Project}_{\partial\mathcal{B}}\left(\bar{\xi}_j - \alpha \cdot \frac{\partial V(\bar{\xi}_j)}{\partial \bar{\xi}}\right)$
8: **end for**
9: $\rho = \gamma \cdot \min_j V(\bar{\xi}_j)$
10: Sample counterexamples $\xi_{\text{adv}}^i \in \mathcal{B}$
11: **for** $adv_descent = 1, 2, \cdots$ **do**
12: $\xi_{\text{adv}}^i \leftarrow \text{Project}_{\mathcal{B}}\left(\xi_{\text{adv}}^i + \beta \cdot \frac{\partial L_{\dot{V}}(\xi_{\text{adv}}^i;\rho)}{\partial \xi_{\text{adv}}}\right)$
13: **end for**
14: $\mathcal{D} \leftarrow \{\xi_{\text{adv}}^i\} \cup \mathcal{D}$
15: **for** $epoch = 1, 2, \cdots$ **do**
16: $\theta \leftarrow \theta - \eta \nabla_\theta L(\theta; \mathcal{D}, \rho)$
17: **end for**
18: **end for**

Loss functions for training. In line 16 of Algorithm 2, we define the overall surrogate loss function as

$$L(\theta; \mathcal{D}, \rho) = \sum_{\xi_{\text{adv}}^i \in \mathcal{D}} L_{\dot{V}}(\xi_{\text{adv}}^i; \rho) + c_1 L_{\text{roa}}(\rho) + c_2 \|\theta\|_1 + c_3 L_{\text{obs}}, \tag{20}$$

where $c_1, c_2, c_3 > 0$ are all given positive constants. The term $L_{\dot{V}}(\bullet)$ is the violation on the Lyapunov derivative condition, defined in (17); the term L_{roa} is the surrogate loss for enlarging the region-of-attraction, defined in (18). To ease the difficulty of verification in the next step, we indirectly reduce the Lipschitz constant of the neural networks through the l_1 norm regularization $\|\theta\|_1$. Finally, we add L_{obs} for output feedback case. We observe that it is important to explicitly regulate the observer performance during the training process. Otherwise, the training can easily diverge, and the observer will become unstable. In particular,

we define the following observer performance loss

$$L_{\text{obs}} = \sum_{\xi_t \in \mathcal{C}} \|\hat{x}_{t+1} - x_{t+1}\|_2, \tag{21}$$

so that by minimizing this loss, the NN-based observer will try to predict the state at the next time step accurately. $\mathcal{C}$ is the set of randomly generated internal states within $\mathcal{B}$.

Discussions. To obtain a certified ROA, prior CEGIS approaches on synthesizing neural-network Lyapunov functions (Chang et al., 2019; Dai et al., 2021) invoked expensive verifiers (SMT or MIP) during training to generate counterexamples. In contrast, we generate a large batch of counterexamples efficiently through the much cheaper PGD attack (requiring gradients only) on GPUs. Our results demonstrate that these cheap counterexamples are sufficient for guiding the training procedure, and the expensive verification process only needs to run once post-training. This finding is aligned with similar observations in the machine learning community (Balunovic & Vechev, 2019; De Palma et al., 2022). Our integration of heuristic PGD and sound verification combines the benefits of both.

4. Experiments

We demonstrate the effectiveness of our formulation in both verification and training. The proposed formulation leads to larger certifiable ROAs than previous works for multiple dynamical systems. The baseline approaches for comparison include: 1) discrete-time neural Lyapunov control (DITL) (Wu et al., 2023) which uses MIP for verification; 2) neural Lyapunov control (NLC) (Chang et al., 2019) employing SMT solvers for verification and counterexample generation; 3) neural Lyapunov control for unknown systems (UNL)(Zhou et al., 2022). Table 2 reports the verification runtime for our trained models in Sec. 4.2 and 4.3.

System	Our runtime	DITL runtime
Pendulum state	11.3s†	7.8s
Path tracking	11.7s	13.3s
Cartpole	129s	448s
PVTOL‡	217s	1935s

† Runtime dominated by α,β-CROWN startup cost.
‡ We discovered the verification implementation for PVTOL in (Wu et al., 2023) missed certain regions in $\mathcal{B}$. But for a fair comparison of verification time, we used the same regions as theirs (see Sec. B.6).

Table 1: Verification time comparison for models obtained using DITL. Our verification scales better than DITL to challenging environments because we do not use MIP solvers.

4.1. Verification of Existing Neural Lyapunov Models

To show better scalability and larger ROAs of our verification algorithm, we first apply our verification procedure to models obtained using state-of-the-art DITL and compare against their verifier. Table 1 records the verification runtime. All the system parameters and dynamics models are the same as in (Wu et al., 2023).

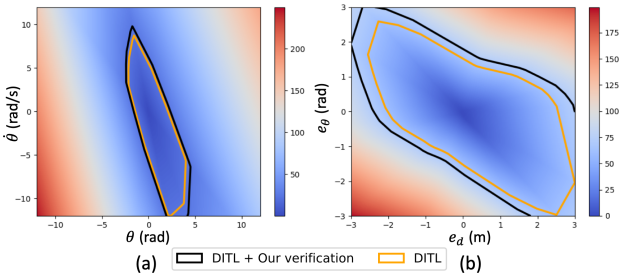

Figure 4: Comparison of certified ROA for the *same model* trained using DITL's code. Our novel verification formulation (14) finds a larger ROA $\mathcal{S}$ (black) than $\tilde{\mathcal{S}}$ (orange) in DITL for (a) inverted pendulum and (b) path tracking.

System	Runtime	System	Runtime
Pendulum state	33s	Pendulum output	94s
Quadrotor state	1.1hrs	Quadrotor output	8.9hrs
Path tracking	39s		

Table 2: Verification runtime for our trained models.

Similar to (Wu et al., 2023), we visualize the ROA for two low-dimensional systems: 1) *Inverted pendulum:* swing up the pendulum to the upright equilibrium $[\theta, \dot{\theta}] = [0, 0]$ with a state-feedback controller. 2) *Path tracking:* a planar vehicle tracks a given path with a state feedback controller. Our novel verification formulation discussed in Sec. 3.2 results in a larger ROA given the same models, as shown in Figure 4. Our ROA can nontrivially intersect the boundary of $\mathcal{B}$, represented as the borders of these figures, rather than only touching the boundary at a single point as in the previous approaches. Compared to the MIP-based verification in DITL, Table 1 shows that our verification procedure offers significant advantages in verification runtime over DITL, especially on more challenging higher-dimensional tasks, such as Cart-pole and PVTOL.

4.2. Training and Verification with New Formulation

Our training and verification formulation, when combined, leads to even larger ROAs. We evaluate the effectiveness of our approach in the following state-feedback systems:

Inverted pendulum and path tracking. We compare our trained models for the inverted pendulum and path tracking against multiple baselines reported in (Wu et al., 2023). Fig. 5 shows that the ROA found by our improved formulation (11) and (12) is a strict superset of all the baseline ROAs. Again, our ROAs nontrivially intersect with the boundary of $\mathcal{B}$ (red borders), which is impossible with the formulation in prior works (Chang et al., 2019; Dai et al., 2021; Wu et al.,

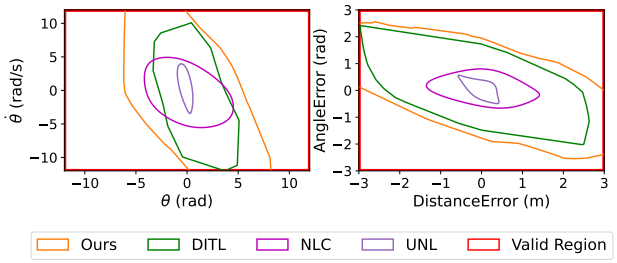

Figure 5: Certified ROA for models trained with different methods. Our approach finds the largest ROA among all approaches for the inverted pendulum (left) and vehicle path tracking (right). The comparison to DITL uses the best Lyapunov network released by its original authors.

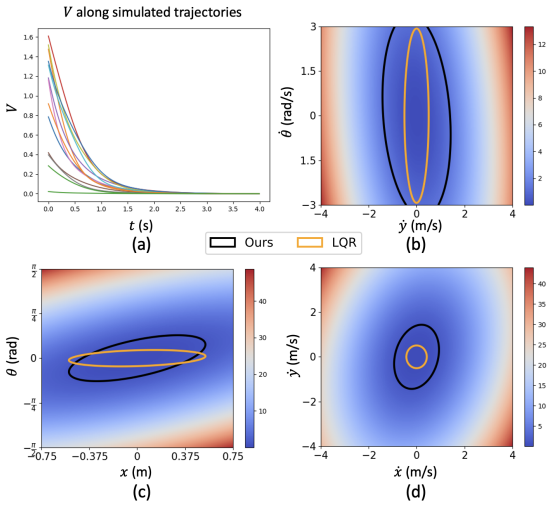

Figure 6: 2D quadrotor state feedback: (a) the Lyapunov function keeps decreasing along simulated trajectories using our NN controller; (b-d) the verified region-of-attraction using our approach (black) and LQR (orange) in different 2-dimensional slices.

2023). In Appendix B.2, we present certified ROAs for both examples with more challenging torque limits.

2D quadrotor. This is a more challenging setting not included in (Wu et al., 2023), where we aim to stabilize a quadrotor to hover at the equilibrium state $[x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}] = \mathbf{0}$. Our new formulation (16b) plays a crucial role in verifying this system. The previous formulation (13) enforces the Lyapunov derivative condition over the entire region $\mathcal{B}$, and we find that PGD attack can always find counterexamples during training. With (13), the learned NN controllers are impossible to verify using the corresponding Lyapunov functions because violations can be detected even during training. In fact, (13) requires $\mathcal{B}$ to lie within the true ROA that can be verified by V, which is not necessarily true for such a large $\mathcal{B}$ in the high-dimensional space. We simulate the system using our NN controller from various initial

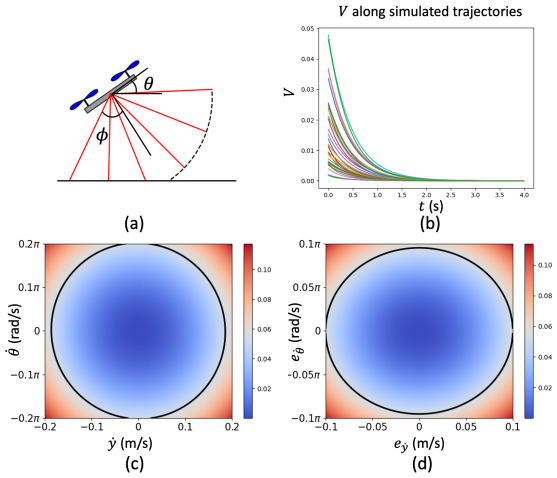

Figure 7: 2D quadrotor output feedback: (a) the quadrotor obtains lidar measurements with 6 truncated rays; (b) the Lyapunov function keeps decreasing along simulated trajectories with our NN controller and observer; (c-d) the black contours represent 2-d slices of the certified ROA. Our approach provides the first formal neural certificate for the quadrotor output feedback system.

conditions within the verified ROA and observe that $V(x)$ always decreases along the simulated trajectories in Fig. 6a. In Fig. 6b−d, we visualize the certified ROA in different 2D slices and compare with that of the clamped LQR controller verified by the quadratic Lyapunov function obtained from the Riccati solution.

4.3. Neural Lyapunov Output Feedback Control

We now apply our method to the more challenging output feedback control setting, which requires training a controller, an observer, and a Lyapunov function. For the first time in the literature, we demonstrate *certified* neural Lyapunov control with *output feedback* in two settings:

Inverted pendulum with angle observation. For the output-feedback pendulum, the controller can only observe the angle θ. Unlike (Chang et al., 2019; Zhou et al., 2022; Wu et al., 2023) which enforced an input constraint much larger than the gravity torque ($|u| \leq 8.15 \cdot mgl$) for state-feedback pendulum, we impose the challenging torque limit $|u| \leq \frac{mgl}{3}$. The black contours in Fig. 1a and 1b show a large verified ROA, whose corresponding sublevel set expands beyond $\mathcal{B}$.

2D quadrotor with a lidar sensor. We validate our approach on a more complicated task of steering a 2D quadrotor to cruise at a constant height $y = 0$ (the ground is at $y = -1$m) as visualized in Fig. 7a. The quadrotor obtains observations from a lidar sensor, which provides truncated distance along 6 rays in the range $\phi \in [-0.15\pi, 0.15\pi]$ up to a sensing horizon of 5m. We remark that SOS-based

methods cannot handle such non-polynomial observation function with clamp $\left(\frac{y}{\cos(\theta-\phi)}, 0, 5\right)$. Similar to the state feedback 2D quadrotor, we compare against the previous formulation (13) and observe that verification is impossible since PGD attack can always find adversarial samples during training. In contrast, training using our formulation converges quickly to the stage where PGD attack cannot find adversarial samples. Fig. 7b demonstrates that the synthesized Lyapunov function using our approach keeps decreasing along the simulated trajectories using our Lyapunov-stable NN controller and observer. The black contours in Fig. 7c and 7d represent a decently large ROA verified by α,β-CROWN.

5. Conclusion

In this paper, we propose a novel formulation to efficiently synthesize and verify neural-network controllers and observers with Lyapunov functions, providing one of the earliest formal stability guarantees for output feedback systems in the literature. Our new formulation actively promotes a large certifiable region-of-attraction. Distinct from prior works which rely on resource-intensive verifiers (e.g., SOS, MIP or SMT) to generate counterexamples during training, we incorporate cost-effective adversarial attacks that notably enhance training efficiency. Post-training, the Lyapunov conditions undergo a rigorous verification procedure tailored for NN verification using α,β-CROWN.

Limitations

While our method improves scalability for neural certificates by avoiding resource-intensive solvers for SOS, MIP, or SMT, the system dimensionality still poses a challenge for rigorous certification. Previous methods relying on expensive complete solvers were only able to handle *state* feedback systems with lower dimensions: (Zhou et al., 2022) only dealt with 2-dimensional systems, (Chang et al., 2019) also suffered beyond 2 dimensions (errors and reproducibility issues are reported here), and (Wu et al., 2023) scaled up to a 4-dimensional cartpole system (as noted in Appendix B.6, their corrected implementation failed for the 6-dimensional PVTOL). Although our approach extends neural certificates from state feedback to *output* feedback control with 8 dimensions, the dimensions of the addressed systems remain moderate. We are interested in exploring the framework's potential in higher dimensional systems with more complicated observation functions beyond the truncated lidar readings, such as images or point clouds.

Acknowledgement

This work was supported by Amazon PO 2D-12585006, NSF 2048280, 2325121, 2244760, 2331966, 2331967 and ONR N00014-23-1-2300:P00001. Huan Zhang is supported in part by the AI2050 program at Schmidt Sciences (Grant #G-23-65921) and NSF 2331967. The authors would like to thank Zico Kolter for valuable discussions and insightful feedback on the paper.

Impact Statement

This paper presents work whose goal is to advance the field of verification for neural network control with Lyapunov stability. Our work steps towards providing guarantees for real-world safety-critical control applications.

References

Abate, A., David, C., Kesseli, P., Kroening, D., and Polgreen, E. Counterexample guided inductive synthesis modulo theories. In *International Conference on Computer Aided Verification*, 2018.

Abate, A., Ahmed, D., Giacobbe, M., and Peruffo, A. Formal synthesis of lyapunov neural networks. *IEEE Control Systems Letters*, 2020.

Åström, K. J. *Introduction to stochastic control theory*. Courier Corporation, 2012.

Athans, M. The role and use of the stochastic linear-quadratic-gaussian problem in control system design. *IEEE transactions on automatic control*, 1971.

Balunovic, M. and Vechev, M. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2019.

Bertsimas, D. and Tsitsiklis, J. N. *Introduction to linear optimization*. Athena scientific Belmont, MA, 1997.

Chang, Y.-C., Roohi, N., and Gao, S. Neural lyapunov control. *Advances in neural information processing systems*, 2019.

Chen, S., Fazlyab, M., Morari, M., Pappas, G. J., and Preciado, V. M. Learning lyapunov functions for hybrid systems. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, 2021.

Dai, H. and Permenter, F. Convex synthesis and verification of control-lyapunov and barrier functions with input constraints. In *IEEE American Control Conference (ACC)*, 2023.

Dai, H., Landry, B., Pavone, M., and Tedrake, R. Counterexample guided synthesis of neural network lyapunov functions for piecewise linear systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*.

Dai, H., Landry, B., Yang, L., Pavone, M., and Tedrake, R. Lyapunov-stable neural-network control. *Robotics: Science and Systems*, 2021.

Dawson, C., Qin, Z., Gao, S., and Fan, C. Safe nonlinear control using robust neural lyapunov-barrier functions. In *Conference on Robot Learning*, 2022.

De Moura, L. and Bjørner, N. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008.

De Palma, A., Bunel, R., Dvijotham, K., Kumar, M. P., and Stanforth, R. Ibp regularization for verified adversarial robustness via branch-and-bound. *arXiv preprint arXiv:2206.14772*, 2022.

Edwards, A., Peruffo, A., and Abate, A. A general verification framework for dynamical and control models via certificate synthesis, 2023.

Everett, M., Habibi, G., Sun, C., and How, J. P. Reachability analysis of neural feedback loops. *IEEE Access*, 2021.

Everett, M., Bunel, R., and Omidshafiei, S. Drip: domain refinement iteration with polytopes for backward reachability analysis of neural feedback loops. *IEEE Control Systems Letters*, 2023.

Fazlyab, M., Morari, M., and Pappas, G. J. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 2020.

Gao, S., Kong, S., and Clarke, E. M. dreal: An smt solver for nonlinear theories over the reals. In *Automated Deduction–CADE-24: 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings 24*. Springer, 2013.

Jin, W., Wang, Z., Yang, Z., and Mou, S. Neural certificates for safe control policies. *arXiv preprint arXiv:2006.08465*, 2020.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Kotha, S., Brix, C., Kolter, J. Z., Dvijotham, K., and Zhang, H. Provably bounding neural network preimages. *Advances in Neural Information Processing Systems*, 36, 2024.

Liu, C., Arnon, T., Lazarus, C., Strong, C., Barrett, C., Kochenderfer, M. J., et al. Algorithms for verifying deep neural networks. *Foundations and Trends® in Optimization*, 2021.

Liu, S., Liu, C., and Dolan, J. Safe control under input limits with neural control barrier functions. In *Conference on Robot Learning*. PMLR, 2023.

Luenberger, D. An introduction to observers. *IEEE Transactions on automatic control*, 1971.

Lyapunov, A. M. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1892.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Majumdar, A., Ahmadi, A. A., and Tedrake, R. Control design along trajectories with sums of squares programming. In *2013 IEEE International Conference on Robotics and Automation*, pp. 4054–4061. IEEE, 2013.

Mathiesen, F. B., Calvert, S. C., and Laurenti, L. Safety certification for stochastic systems via neural barrier functions. *IEEE Control Systems Letters*, 7:973–978, 2022.

Parrilo, P. A. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology, 2000.

Pólik, I. and Terlaky, T. A survey of the s-lemma. *SIAM review*, 2007.

Ravanbakhsh, H. and Sankaranarayanan, S. Counter-example guided synthesis of control lyapunov functions for switched systems. In *2015 54th IEEE conference on decision and control (CDC)*.

Rober, N., Katz, S. M., Sidrane, C., Yel, E., Everett, M., Kochenderfer, M. J., and How, J. P. Backward reachability analysis of neural feedback loops: Techniques for linear and nonlinear systems. *IEEE Open Journal of Control Systems*, 2023.

Shi, Z., Jin, Q., Kolter, J. Z., Jana, S., Hsieh, C.-J., and Zhang, H. Formal verification for neural networks with general nonlinearities via branch-and-bound. *2nd Workshop on Formal Verification and Machine Learning*, 2023.

Slotine, J.-J. E., Li, W., et al. *Applied nonlinear control*. Prentice hall Englewood Cliffs, NJ, 1991.

Sun, D., Jha, S., and Fan, C. Learning certified control using contraction metric. In *Conference on Robot Learning*. PMLR, 2021.

Tedrake, R., Manchester, I. R., Tobenkin, M., and Roberts, J. W. Lqr-trees: Feedback motion planning via sums-of-squares verification. *The International Journal of Robotics Research*, 2010.

Vincent, J. A. and Schwager, M. Reachable polyhedral marching (rpm): An exact analysis tool for deep-learned control systems. *arXiv preprint arXiv:2210.08339*, 2022.

Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C.-J., and Kolter, J. Z. Beta-CROWN: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. *Advances in Neural Information Processing Systems*, 2021.

Wang, Y., Zhan, S., Wang, Z., Huang, C., Wang, Z., Yang, Z., and Zhu, Q. Joint differentiable optimization and verification for certified reinforcement learning. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, pp. 132–141, 2023.

Wu, J., Clark, A., Kantaros, Y., and Vorobeychik, Y. Neural lyapunov control for discrete-time systems. *arXiv preprint arXiv:2305.06547*, 2023.

Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K.-W., Huang, M., Kailkhura, B., Lin, X., and Hsieh, C.-J. Automatic perturbation analysis for scalable certified robustness and beyond. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a.

Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., and Hsieh, C.-J. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2020b.

Yang, L., Dai, H., Amice, A., and Tedrake, R. Approximate optimal controller synthesis for cart-poles and quadrotors via sums-of-squares. *IEEE Robotics and Automation Letters*, 2023.

Yin, H., Seiler, P., and Arcak, M. Stability analysis using quadratic constraints for systems with neural network controllers. *IEEE Transactions on Automatic Control*, 2021.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Zhang, H., Wang, S., Xu, K., Li, L., Li, B., Jana, S., Hsieh, C.-J., and Kolter, J. Z. General cutting planes for bound-propagation-based neural network verification. *Advances in Neural Information Processing Systems*, 2022.

Zhang, T., Kahn, G., Levine, S., and Abbeel, P. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. In *2016 IEEE international conference on robotics and automation (ICRA)*, 2016.

Zhou, R., Quartz, T., De Sterck, H., and Liu, J. Neural lyapunov control of unknown nonlinear systems with stability guarantees. *Advances in Neural Information Processing Systems*, 35:29113–29125, 2022.

System	Feedback	Lyapunov function	controller	observer	Region-of-interest (upper limit)
Pendulum	State	(16, 16, 8)	(8, 8, 8, 8)	—	$[12, 12]$
Path tracking	State	(16, 16, 8)	(8, 8, 8, 8)	—	$[3, 3]$
Quadrotor	State	Quadratic	(8, 8)	—	$[0.75, 0.75, \frac{\pi}{2}, 4, 4, 3]$
Pendulum	Output	Quadratic	(8, 8, 8)	(8, 8)	$[0.4\pi, 0.4\pi, 0.1\pi, 0.1\pi]$
Quadrotor	Output	Quadratic	(8, 8)	(8, 8)	$[0.1, 0.2\pi, 0.2, 0.2\pi, 0.05, 0.1\pi, 0.1, 0.1\pi]$

Table 3: Neural network size and region-of-interest for each task. The tuples denote the number of neurons in each layer of the neural network. All the networks use the leaky ReLU activation function.

A. Proofs

A.1. Proof of Theorem 3.3

Proof.

$$
\begin{aligned}
&(-F(\xi_t) \geq 0 \wedge \xi_{t+1} \in \mathcal{B}) \ \vee \ (V(\xi_t) \geq \rho), \ \forall \xi_t \in \mathcal{B} && \text{(22a)}\\
\Leftrightarrow &(-F(\xi_t) \geq 0 \wedge \xi_{t+1} \in \mathcal{B}), \forall (\xi_t \in \mathcal{B} \wedge V(\xi_t) < \rho) && \text{(22b)}\\
\Leftrightarrow &(V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \wedge \xi_{t+1} \in \mathcal{B}), \forall \xi_t \in \mathcal{S} && \text{(22c)}\\
\Leftrightarrow &(V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \wedge \xi_{t+1} \in \mathcal{B} \wedge V(\xi_{t+1}) < \rho), \forall \xi_t \in \mathcal{S} && \text{(22d)}\\
\Leftrightarrow &(V(\xi_{t+1}) - V(\xi_t) \leq -\kappa V(\xi_t) \wedge \xi_{t+1} \in \mathcal{S}), \forall \xi_t \in \mathcal{S} && \text{(22e)}
\end{aligned}
$$

Hence $\mathcal{S}$ is an invariant set and the function V decreases exponentially within this invariant set, which proves stability, and $\mathcal{S}$ as an inner approximation of the ROA. The appearance of $V(\xi_{t+1}) < \rho$ in (22d) arises from the fact that $V(\xi_t) < \rho, \forall \xi_t \in \mathcal{S}$ and $V(\xi_{t+1}) \leq V(\xi_t) < \rho$ by (8c). □

A.2. Proof of Theorem 3.4

Proof.

$$
\begin{aligned}
&\min(\text{ReLU}\,(F(\xi_t)) + c_0 H(\xi_{t+1}), \ \rho - V(\xi_t)) \leq 0 \ \forall \xi_t \in \mathcal{B} && \text{(23a)}\\
\Leftrightarrow &(\text{ReLU}\,(F(\xi_t)) + c_0 H(\xi_{t+1}) \leq 0) \vee (\rho - V(\xi_t) \leq 0) \ \forall \xi_t \in \mathcal{B} && \text{(23b)}\\
\Leftrightarrow &(\text{ReLU}\,(F(\xi_t)) \leq 0 \wedge c_0 H(\xi_{t+1}) \leq 0) \vee (\rho - V(\xi_t) \leq 0) \ \forall \xi_t \in \mathcal{B} && \text{(23c)}\\
\Leftrightarrow &(F(\xi_t) \leq 0 \wedge H(\xi_{t+1}) \leq 0) \vee (\rho - V(\xi_t) \leq 0) \ \forall \xi_t \in \mathcal{B} && \text{(23d)}\\
\Leftrightarrow &(F(\xi_t) \leq 0 \wedge H(\xi_{t+1}) \leq 0), \forall (\xi_t \in \mathcal{B} \wedge V(\xi_t) < \rho) && \text{(23e)}\\
\Leftrightarrow &(F(\xi_t) \leq 0 \wedge H(\xi_{t+1}) \leq 0), \forall \xi_t \in \mathcal{S} && \text{(23f)}
\end{aligned}
$$

(23c) follows from the fact that both $\text{ReLU}\,(F(\xi_t))$ and $H(\xi_{t+1})$ are nonnegative. □

B. Experiment Details

B.1. Candidate State Selection for Growing ROA

On the one hand, the candidate states that we hope to be covered in the invariant set $\mathcal{S}$ should be diverse enough to encourage the ROA to grow in all directions; on the other hand, they should not be irregularly spread across the entire state space because such candidates might shape the ROA in conflicting directions and deteriorate the satisfaction of the Lyapunov derivative condition (8c). We require the candidate states to have the same distance from the goal state in the metric of the Lyapunov function value and start by sampling states on the 1-level set of a reference Lyapunov function V_{ref}. For state feedback, we choose V_{ref} to be the LQR Lyapunov function $x^T S x$ (S is the solution to the Riccati equation); for output feedback, we select $V_{\text{ref}} = x^T S x + e^T P^{-1} e$ (P is the asymptotic state variance at the goal state obtained by solving the discrete Riccati equation). After the NN Lyapunov function is trained to achieve a reasonable ROA, we can sample states slightly outside the current ROA as candidates.

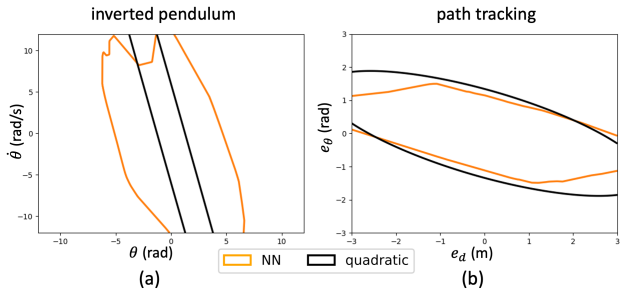

Figure 8: Comparison of certified ROAs obtained using NN and quadratic Lyapunov functions: (a) inverted pendulum with torque limit $|u| \leq 1.02 \cdot mgl$ (b) path tracking with $|u| \leq \frac{L}{v}$.

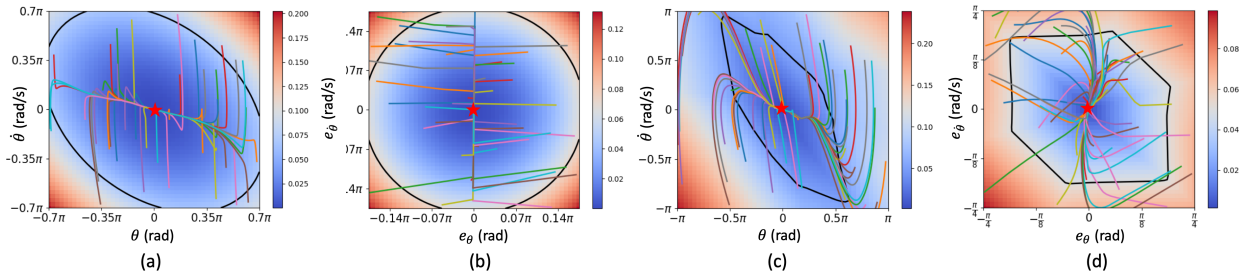

Figure 9: Pendulum output feedback: phase portrait and certified ROA with torque limit $|u| \leq 1.36 \cdot mgl$ using (a)-(b) quadratic Lyapunov function and (c)-(d) neural-network Lyapunov function.

B.2. Pendulum State Feedback & Path Tracking with Challenging Torque Limits

In Sec. 4.2, we provide certified ROAs for inverted pendulum and path tracking with large (easy) torque limits as a fair comparison to all the baselines ($|u| \leq 8.15 \cdot mgl$ for pendulum and $|u| \leq 1.68\frac{L}{v}$ for path tracking). In Fig. 8, we demonstrate ROAs for small (challenging) torque limits verified with both neural and quadratic Lyapunov functions. While neural Lyapunov functions can be more expressive, quadratic Lyapunov functions are often easier to train and have better interpretability. Our approach aims to leverage the strengths of both representations, allowing practitioners to select the most suitable form based on their specific requirements and trade-offs between expressivity, convergence, and interpretability.

B.3. Pendulum Output Feedback

In Fig. 9, we visualize the phase portrait and certified ROA with a larger torque limit $|u| \leq 1.36 \cdot mgl$. We synthesize a quadratic Lyapunov function in the region-of-interest $\pm[0.7\pi, 0.7\pi, 0.175\pi, 0.175\pi]$ and an NN Lyapunov function in $\pm[\pi, \pi, 0.25\pi, 0.25\pi]$. With such a large control constraint, the phase portrait demonstrates that starting from many initial states (even outside the verified ROA), the system can always converge to the upright equilibrium with the synthesized controller and observer. This result suggests that our novel loss function (20) both leads to a large certified ROA and enables good generalization.

B.4. 2D Quadrotor Output Feedback

In Fig. 10, we visualize the snapshots of the quadrotor stabilized by our NN controller and observer with decently large initial state estimation error. We observe that the NN controller and observer generalize well outside of the certified ROA, empirically steering the quadrotor to cruise at the constant height for most of the states within the box region.

B.5. Validation region $\mathcal{B}$ in Fig.5

We use the validation region $\mathcal{B}$ as reported in each paper, detailed in Table 4.

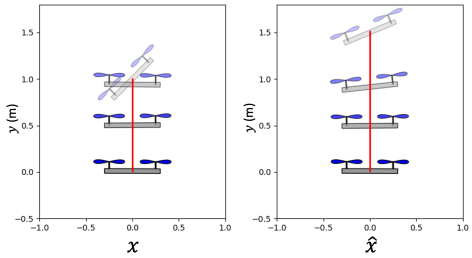

Figure 10: Snapshots of simulating 2D quadrotor with lidar sensor using our NN controller and observer. The red curve is the quadrotor's trajectory to balance at $y = 0$. The left plot illustrates the true states and the right plot shows the state estimates. The initial state estimate error is $[0.5, -\frac{\pi}{8}, 0.1, \frac{\pi}{8}]$.

	Inverted Pendulum	Path tracking
Ours	$\|x\|_\infty \leq 12$	$\|x\|_\infty \leq 3$
DITL	$\|x\|_\infty \leq 12$	$\|x\|_\infty \leq 3$
NLC	$\|x\|_2 \leq 6$	$\|x\|_2 \leq 1.5$
UNL	$\|x\|_2 \leq 4$	$\|x\|_2 \leq 0.8$

Table 4: Validation region $\mathcal{B}$ in each approach.

B.6. Region for PVTOL in (Wu et al., 2023)

We mentioned in Table 1 that there is an implementation issue in (Wu et al., 2023) regarding region $\mathcal{B}$ for PVTOL. (Wu et al., 2023) takes $0.1 \leq \|x\|_\infty \leq 1$ for $\mathcal{B}$, where $\|x\|_\infty$ should be the *maximum* absolute value among all the dimensions in x. We found that the code of (Wu et al., 2023)[2] mistakenly implemented $\|x\|_\infty$ as the *minimum* absolute value among all the dimensions in x when enforcing the $\|x\|_\infty \geq 0.1$ constraint, which makes the resulting $\mathcal{B}$ much smaller than desired. We found the issue in their code released by 12/10/2023, and we were able to reproduce the results on their paper using this version of code with an incorrect $\mathcal{B}$. While the implementation issue has been fixed in their current version of code released on 12/29/2023, we found that the new version is not able to successfully finish training the model on PVTOL with the correct $\mathcal{B}$.

[2] https://github.com/jlwu002/nlc_discrete/blob/main/pvtol.py