Synthesizing Stable Reduced-Order Visuomotor Policies for Nonlinear Systems via Sums-of-Squares Optimization

Glen Chou and Russ Tedrake

***Abstract*— We present a method for synthesizing dynamic, reduced-order output-feedback polynomial control policies for control-affine nonlinear systems which guarantees runtime stability to a goal state, when using visual observations and a learned perception module in the feedback control loop. We leverage Lyapunov analysis to formulate the problem of synthesizing such policies. This problem is nonconvex in the policy parameters and the Lyapunov function that is used to prove the stability of the policy. To solve this problem approximately, we propose two approaches: the first solves a sequence of sum-of-squares optimization problems to iteratively improve a policy which is provably-stable by construction, while the second directly performs gradient-based optimization on the parameters of the polynomial policy, and its closed-loop stability is verified *a posteriori*. We extend our approach to provide stability guarantees in the presence of observation noise, which realistically arises due to errors in the learned perception module. We evaluate our approach on several underactuated nonlinear systems, including pendula and quadrotors, showing that our guarantees translate to empirical stability when controlling these systems from images, while baseline approaches can fail to reliably stabilize the system.**

I. Introduction

For autonomous robots to be effective in the real world, we need provable assurances on their safety and reliability. In these unstructured settings, robots typically lack full state information, and must complete tasks while controlling using only sensor measurements (i.e., outputs); this perception-based control problem is known as *output-feedback*. For robots of interest with nonlinear dynamics, synthesizing output-feedback controllers is known to be difficult. Partial observability and sensor noise require that output-feedback controllers extract information from a *history* of observations to stabilize the system. Moreover, in many domains of interest, these observations are non-smooth and high-dimensional (e.g., images), which can increase the amount of data needed to learn a good visuomotor control policy, i.e., a policy which takes (a history of) images as input and returns a control action. Finally, underactuation due to input limits and nonlinearities in the dynamics can lead to numerous local minima when attempting to optimize output-feedback control policies. To address these difficulties, recent work (e.g., [1], [2]) leverages the power of neural networks (NNs) and deep reinforcement learning (RL) to tackle the output-feedback problem. However, the resulting control policies are complex, rendering the learning process data-hungry and the closed-loop stability of the system difficult to certify.

Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139. {gchou, russt}@mit.edu. This work is supported in part by the MIT Quest for Intelligence, and Amazon.com Services LLC Award #2D-06310236.

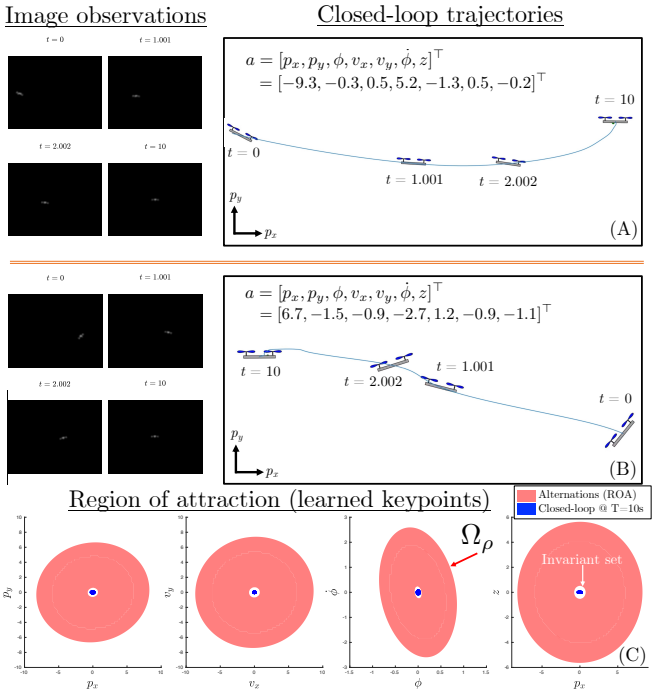

Fig. 1. **(A-B)**: We synthesize a dynamic-output-feedback controller that stabilizes the planar quadrotor to the origin from pixels, using a learned perception map $\hat{h}_e$ in the control loop, from initial conditions (ICs) $a = [p_x, p_y, \phi, v_x, v_y, \dot{\phi}, z]^\top$. Here, $z \in \mathbb{R}$ is the controller latent variable. Left: Grayscale 128x96 images input to the controller. Right: Time-lapse of the stabilized trajectories. **(C)**: Slices (all other states set to zero) of a certified inner approximation of the closed-loop region of attraction (ROA) in red. While the system may not converge exactly to the origin due to errors in the learned perception map, all states in the ROA are guaranteed to converge to an invariant set (shown in white) around the origin. To empirically show the invariance of this set, we plot (in blue) the states reached after 10s have elapsed, for 500 ICs sampled from the ROA.

In this paper, we challenge the notion that complex controllers are required to stabilize nonlinear robotic systems from high-dimensional sensor inputs like images. In particular, we show that given a useful reduction of the high-dimensional observations which can be learned from data (such as keypoints rigidly attached to the robot), simple *dynamic* (i.e., stateful) policies, which are linear or a low-degree polynomial in the reduced observations, can effectively stabilize a variety of nonlinear, underactuated systems. The latent states of dynamic policies enable non-trivial information-gathering, which cannot be achieved by a naïve interconnection of independently-designed state estimators and state-feedback controllers. Moreover, due to the simplicity of these controllers, our method can leverage powerful tools like sums-of-squares (SOS) programming to synthesize provably-stable output-feedback policies. Inspired by the scalability of gradient-based policy learning in RL [3],

we also provide a method that directly learns the parameters of the polynomial controller through gradient descent (GD) (Sec. V-B). Due to this parameterization, an ROA for the learned controller can be readily verified in a post-hoc fashion by solving a set of smaller SOS optimizations. Finally, a key strength of our controllers is that they *do not explicitly reconstruct the full state*. This is critical as it 1) reduces the problem dimension, keeping the SOS problems tractable for high-dimensional systems, and 2) sidesteps the need for an accurate initial state estimate. Our specific contributions are:

- Two approaches for synthesizing dynamic reduced-order output-feedback controllers (via SOS programming and via gradient-based optimization)
- An extension for verifying robust closed-loop stability under observation error, enabling end-to-end stability guarantees when using images as input and an imperfect learned perception module in the control loop
- Validation on a variety of nonlinear systems, showing that our simple polynomial controllers match or outperform baselines in optimizing visuomotor policies

II. Related Work

Many learning-based approaches for designing control policies for nonlinear robotic systems from rich sensor inputs like images have been recently proposed [1], [2], [4]. These approaches typically rely on model-based or model-free reinforcement learning, which leverage data to learn complex NN controllers which take images as input. However, these controllers are complex, difficult to analyze, and may not reliably stabilize the system. On the other hand, many model-based methods exist for synthesizing stable state-feedback controllers for nonlinear systems, and we know that simple, low-degree polynomial controllers can stabilize many robots. Using tools like SOS optimization [5], Lyapunov analysis [6], barriers [7], and occupation measures [8] can be used to algorithmically synthesize stable polynomial controllers.

For the specific case of linear systems, techniques like linear-quadratic-Gaussian (LQG) control [9] and its reduced-order counterpart [10] can efficiently synthesize dynamic-output-feedback controllers. While there is extensive theoretical study on sufficient conditions for nonlinear output-feedback stabilization [11], [12], there are far fewer methods for algorithmic synthesis of such controllers. Most existing work leverages SOS optimization, and focuses on finding static output-feedback controllers [13], [14] (i.e., output-feedback controllers which only take in the current observation as input) or decouples the state-feedback and full-order observer design problems [15], [16], [17]. While in principle, dynamic controllers can be written as static controllers by augmenting the observations with the latent variables, directly applying these methods when the latent dynamics are also being optimized, as in our method, leads to additional nonconvexities. Moreover, these methods are evaluated on low-dimensional systems, and are driven by low-dimensional observations. In contrast, we solve a reduced-order dynamic-output-feedback policy synthesis problem, and apply it on high-dimensional robotic systems for image-based control.

Finally, there is recent work in Lyapunov-based control from high-dimensional observations. Learned approximate Lyapunov functions are used in [18] to stabilize from LiDAR; however, the Lyapunov conditions are not actually certified. Other work uses images with barriers [7]; however, the full state must be directly invertible from a single observation, which is not possible for most robotic systems. More recent work [19] leverages contraction theory to control safely from images, but does so by decoupling the state-feedback and estimation problems and requires estimation of the full state.

III. Preliminaries

We consider control-affine, partially-observed nonlinear systems, with state space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$, control space $\mathcal{U} \subseteq \mathbb{R}^{n_u}$, and observation space $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$,

$$\dot{x}(t) = f(x(t), u(t)) = f_1(x(t)) + f_2(x(t))u(t) \tag{1a}$$

$$y(t) = h(x(t)), \tag{1b}$$

where $f : \mathcal{X} \times \mathcal{U} \to \bigcup_{x\in\mathcal{X}} T_x\mathcal{X}$ and $h : \mathcal{X} \to \mathcal{Y}$, and where $T_x\mathcal{X}$ is the tangent space of $\mathcal{X}$ at x. In this paper, we assume $f_1 : \mathcal{X} \to \bigcup_{x\in\mathcal{X}} T_x\mathcal{X}$ and $f_2 : \mathcal{X} \to \mathbb{R}^{n_x\times n_u}$ are polynomial functions of x, and that $\mathcal{Y}$ contains high-dimensional image observations. We do not assume h is polynomial. Since designing controllers directly as a function of the pixels, which may be a non-smooth function of the state (due to aliasing from finite resolution images), can be difficult, we assume knowledge of an approximate smooth reduced-dimensional representation of the information in the images. Popular visual representations can be used here (e.g., dense descriptors [20]) if they are learned such that they are a simple (roughly polynomial) function of x. In this paper, we use keypoints, denoted as $y_k^* \in \mathcal{Y}_k \subseteq \mathbb{R}^{n_k}$, which are points in the workspace that are rigidly attached to the robot (see Fig. 2 for examples). Keypoints are a common feature representation used in computer vision [21] and robotics [22] for pose estimation. For robots modeled as rigid bodies, y_k^* can be written as a polynomial function of the state, with a change of variables (see Sec. VI for examples). An approximate map from images to keypoints $\hat{h}_e : \mathcal{Y} \to \mathcal{Y}_k$, i.e., a keypoint *extractor*, can be learned in a supervised fashion from a labeled dataset of images and keypoints. Here, $\hat{h}_e$ need not be polynomial; in this paper, we represent $\hat{h}_e$ as a convolutional neural network (CNN), and train it to directly output keypoints y_k. The resulting keypoints are

$$\begin{aligned} y_k(t) = \hat{h}_e(h(x(t))) = &\quad h_k(x(t)) + w(t) \\ \doteq &\quad y_k^*(t) + w(t) \end{aligned} \tag{2}$$

where $h_k : \mathcal{X} \to \mathcal{Y}_k$ is polynomial in x and $w \in \mathcal{W} \subseteq \mathbb{R}^{n_k}$ is a bounded disturbance which models the error in the keypoints predicted by the learned keypoint extractor y_k, relative to the perfect keypoints y_k^*.

A. Lyapunov analysis and SOS optimization

For a system $\dot{x} = f(x)$ with equilibrium point x_0, if we can find a C^1 Lyapunov function $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ satisfying

$$V(x_0) = 0, \quad \dot{V}(x_0) = 0, \tag{3a}$$

$$x \in \Omega_\rho^x \wedge x \neq x_0 \Rightarrow V(x) > 0 \wedge \dot{V}(x) < 0, \tag{3b}$$

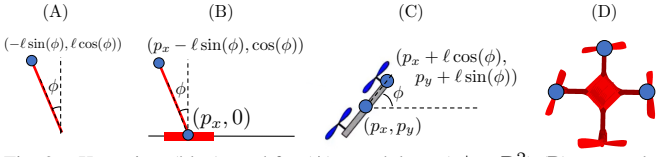

Fig. 2. Keypoints (blue) used for (**A**): pendulum, ($y_k^* \in \mathbb{R}^2$) (**B**): cart-pole ($y_k^* \in \mathbb{R}^4$), (**C**): 2D quadrotor ($y_k^* \in \mathbb{R}^4$), (**D**): 3D quadrotor ($y_k^* \in \mathbb{R}^9$).

where $\Omega_\rho^x \doteq \{x \mid V(x) \leq \rho\}$ is the ρ-sublevel set of V, then Ω_ρ^x is contained in the system's region of attraction (ROA). Finding a V satisfying (3) requires enforcing non-negativity of polynomials (V and $-\dot{V}$) over a basic semialgebraic set (i.e., a set defined by a finite number of polynomial (in)equalities). While this is NP-hard, we can efficiently enforce that a polynomial is a sum of squares (SOS), which implies non-negativity. A polynomial p of degree d in indeterminate variables $x_1, \ldots, x_n$, $p(x_1, \ldots, x_n)$, is SOS if it can be written as $\sum_{i=1}^m q_i^2(x)$, where $q_i(\cdot)$ are polynomials. This is equivalent to the existence of a $Q \succeq 0$ such that $p(x) = \tilde{m}(x)^\top Q \tilde{m}(x)$, where $\tilde{m}(x)$ is a polynomial basis; Q can be found with semidefinite programming.

IV. Problem Formulation

We wish to design a dynamic-output-feedback controller $u : \mathcal{Y}_k \times \mathcal{Z} \to \mathcal{U}$, with latent dynamics $z : \mathcal{Y}_k \times \mathcal{Z} \to \bigcup_{z \in \mathcal{Z}} T_z \mathcal{Z}$, where $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$:

$$u = k(z, y_k), \qquad \text{(4a)} \qquad\qquad \dot{z} = l(z, y_k), \qquad \text{(4b)}$$

which when applied to system (1), maximizes the volume of the closed-loop ROA around a desired equilibrium point x_0. To make the synthesis and verification of this controller compatible with SOS programming, we search over a subset of *polynomial* controllers by parameterizing k and l as a polynomial of z and y_k, i.e., $u = \theta_k^\top m_k(z, y_k)$ and $l = \theta_l^\top m_l(z, y_k)$, for monomial bases $m_{k/l}(z, y_k)$ of degree d_k and d_l, respectively. Note that while (4b) is not a explicitly a function of u (to avoid bilinearities between θ_c and θ_m), (4b) can still recover u since (4a) is also a polynomial of y_k and z. To find an inner approximation of the ROA, we first define the augmented state $a = [x^\top, z^\top]^\top \in \mathcal{X} \times \mathcal{Z} \doteq \mathcal{A}$ and dynamics $\dot{a} = [f(x, u)^\top, l(z, y_k)^\top]^\top$. Then, ideally, we wish to jointly search for a Lyapunov function $V : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}_{\geq 0}$, controller k, and latent dynamics l that solves

$$\begin{aligned} \underset{V,k,l}{\text{maximize}} \quad & \text{Vol}(\Omega_\rho) \\ \text{subject to} \quad & V(x,z) > 0, \quad \forall (x,z) \neq (x_0, 0) \doteq a_0 \\ & V(x_0, 0) = 0 \\ & \dot{V}(x,z) < 0, \quad \forall x \in \Omega_\rho, \end{aligned} \tag{5}$$

where the ρ-sublevel set of V is denoted $\Omega_\rho \doteq \{x, z \mid V(x,z) \leq \rho\}$, and ρ is fixed (to set the scaling of V). The controller returned by (5) is guaranteed to stabilize any initial conditions (ICs) $(x,z) \in \Omega_\rho$ to the augmented goal a_0. While the pointwise (non)-negativity constraints can be handled via SOS, (5) is still nonconvex, due to bilinearities between V and k, l in the final constraint of (5); moreover, additional nonconvexities arise with the constraints needed to model input constraints. Given the challenge of solving (5) exactly, we propose methods to approximately solve (5) in Sec. V.

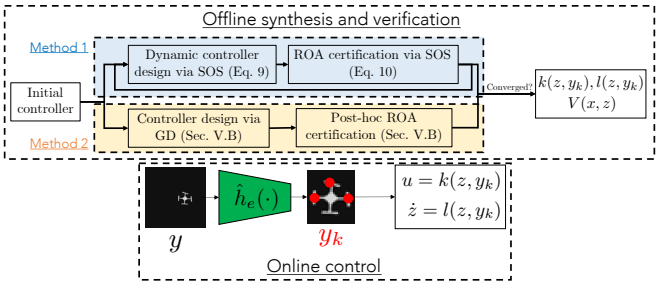

Fig. 3. Method overview. **Offline**: we synthesize a dynamic-output-feedback control policy, either through SOS or through direct gradient-based optimization. **Online**: we compose the learned perception system with the output-feedback policy to obtain a control action from an input image.

V. Control Synthesis and Verification

We propose two methods for approximately solving (5) (see Fig. 3 for an overview). We first apply bilinear alternations (Sec. V-A), i.e., we perform coordinate ascent on the ROA volume by fixing alternating subsets of the variables in (5) and solving the resulting convex subproblems. Our second method directly learns θ_c and θ_m through GD (Sec. V-B) and certifies an ROA *a posteriori*. We compare the two methods in Sec. VI, and discuss their tradeoffs in Sec. VII.

A. Method 1: SOS alternations

Our alternation scheme is summarized in Alg. 1. We break down each individual optimization below.

Algorithm 1: Bilinear alternations for solving (5)

Input: θ_k^{init}, θ_l^{init} `// initial controller parameters`
1 $V, L \leftarrow$ solve (6) using $(\theta_k^{\text{init}}, \theta_l^{\text{init}})$
2 $\rho, L_b \leftarrow$ solve (7) using (V)
3 $\hat{E} \leftarrow I$
4 **for** $j = 1, \ldots,$ *MAX ITER* **do**
5 $\quad \rho, \theta_k, \theta_l, E, L, L_{\text{ell}} \leftarrow$ solve (9) using $(V, \rho, \hat{E})$
6 $\quad E, V \leftarrow$ solve (10) using $(\theta_k, \theta_l, L, L_{\text{ell}}, \rho, \hat{E})$
7 $\quad \hat{E} \leftarrow E$
8 **return** $V, \rho, \theta_k, \theta_l$

First, given a set of initial controller parameters θ_k^{init} and θ_l^{init} (see Sec. V-A.1 on how we initialize), we search for a Lyapunov function valid in a ball around a_0 (Alg. 1, line 1):

$$\begin{aligned} \text{find} \quad & V, L \\ \text{subject to} \quad & V \text{ is SOS}, \quad L \text{ is SOS} \\ & -\tfrac{\partial V}{\partial a}^\top \dot{a} + L(\|a - a_0\|_2^2 - r) \text{ is SOS}. \end{aligned} \tag{6}$$

This enforces $\dot{V} < 0$ over a ball of radius r centered at a_0. To find an initial ROA, we find the largest sublevel set of V, Ω_ρ, contained in this ball, by solving for a fixed r (line 2):

$$\begin{aligned} \underset{\rho, L_b}{\text{maximize}} \quad & \rho \\ \text{subject to} \quad & V - \rho + L_b(r - \|a - a_0\|_2^2) \text{ is SOS} \\ & L_b \text{ is SOS}. \end{aligned} \tag{7}$$

Then, in line 5, given the fixed Lyapunov function V and sublevel set Ω_ρ, we search for θ_k, θ_l and SOS Lagrange multipliers L for enforcing the Lyapunov conditions over Ω_ρ. As a surrogate for maximizing the volume of Ω_ρ, we maximize the volume of an ellipsoid $\mathcal{E}$ inscribed within Ω_ρ: $\mathcal{E} \doteq \{a \mid (a - a_0)^\top E (a - a_0) \leq 1\}$. This containment condition can be enforced by the following SOS constraints:

$$(a-a_0)^\top E(a-a_0) - 1 + L_{\text{ell}}(\rho - V) \text{ is SOS}, \quad (8a)$$
$$L_{\text{ell}} \text{ is SOS}, \quad (8b)$$

while maximizing the volume of $\mathcal{E}$ can be done by minimizing the log determinant of E. As $\log\det(E)$ is concave in E, its minimization is non-convex; thus, we minimize a linearization of $\log\det(\cdot)$ around the ellipsoid from the previous iteration $\hat{E}$, which can be written as $\log\det(E) \approx \log\det(\hat{E}) + \text{tr}(\hat{E}^{-1}(E-\hat{E}))$ [23]. Removing constants in the linearization and putting everything together gives

$$\begin{array}{ll} \underset{\theta_k,\theta_l,E,L,L_{\text{ell}}}{\text{minimize}} & \text{tr}(\hat{E}^{-1}E) \\ \text{subject to} & -\frac{\partial V}{\partial a}^\top \dot{a}(\theta_k,\theta_l) + L(V-\rho) \text{ is SOS} \\ & L \text{ is SOS}, \quad \text{Eq. (8a)}, \quad \text{Eq. (8b)} \end{array} \quad (9)$$

However, note that V and ρ are fixed in (9), and the objective is just meant to provide a Lagrange multiplier L_{ell} for the tightest inscribed ellipsoid in Ω_ρ, which is to be used in the next alternation step (10). To explicitly increase the ROA in (9), we can maximize ρ in an outer maximization via bisection search, which aims to find a controller that increases the ROA with respect to the current candidate V.

Finally, for fixed controller parameters and Lagrange multipliers, we aim to find an improved Lyapunov function V that can certify a larger ROA for the current controller by maximizing the volume of an ellipsoid inscribed in Ω_ρ:

$$\begin{array}{ll} \underset{E,V}{\text{minimize}} & \text{tr}(\hat{E}^{-1}E) \\ \text{subject to} & -\frac{\partial V}{\partial a}^\top \dot{a} + L(V-\rho) \text{ is SOS} \\ & L \text{ is SOS}, \quad \text{Eq. (8a)} \end{array} \quad (10)$$

1) Initializing the alternations: Alg. 1 requires an initial guess for the controller. In the full-state reconstruction case, i.e., $n_z = n_x$, we can solve LQG for the linearization of (1a)-(2) around a_0 to obtain a locally-stable initialization. However, for the reduced-order case, i.e., $n_z < n_x$, the separation principle does not hold and solving the reduced-order LQG problem involves nonlinear solvers [10] which may not converge. As an alternative, we use GD to optimize sampled closed-loop trajectories to initialize the controller.

In particular, we initialize θ_k and θ_l to random values, sample ICs $\{x_i\}_{i=1}^{N_{\text{samp}}}$ from a region around the equilibrium a_0, roll out the policy for a fixed time horizon T on a ΔT-time discretized version of the dynamics (1a) to obtain N_{samp} trajectories $\xi_i \doteq \{x_i^t, u_i^t\}_{t=1}^T$. We define a cost on trajectories $c(\xi) = c_T(x_T) + \sum_{t=1}^{T-1} \alpha_a \|a_t - a_0\|_2^2 + \alpha_u \|u_t\|_2^2$ for weighting parameters $\alpha_a, \alpha_u \geq 0$, and evaluate the cost of each trajectory $c(\xi_i)$. Finally, we define our policy loss as an averaged cost over trajectories,

$$\mathcal{L} \doteq \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} c(\xi_i), \quad (11)$$

and minimize (11) to improve θ_k and θ_l. Minimizing (11) has the effect of drawing the trajectories toward a_0, thereby indirectly increasing the size of the ROA. We now discuss extensions to our basic alternations algorithm (Alg. 1).

2) Control input constraints: We can ensure that the synthesized controller can stabilize the system even in the presence of control constraints through adding additional constraints. For example, for the scalar-valued input case, to ensure stabilization given an upper control limit $u \leq \bar{u}$, we can enforce $\frac{\partial V}{\partial x} f(x,\bar{u}) + \frac{\partial V}{\partial z} l < 0$ for all states in $\Omega_\rho \setminus \{a_0\}$ where $u(x) \geq \bar{u}$. This involves additional SOS multipliers L_k (see Sec. IV of [6] for details) and bilinearities between L_k and k. To avoid these bilinearities, we search for these multipliers L_k in (10) (Alg. 1, line 6).

3) Trigonometric terms: To handle trigonometric terms which arise in rigid body dynamics, e.g., $\sin(\phi)$ for the angle ϕ of an inverted pendulum, we can perform a change of variables to render the dynamics polynomial. Specifically, we replace any instances of $\sin(\phi)$ and $\cos(\phi)$ in (1) with auxiliary state variables s and c, and add an additional constraint $s^2 + c^2 = 1$. As the dynamics are constrained, we only need to enforce the Lyapunov conditions over $\{x \mid s^2 + c^2 = 1\}$, which can be enforced with additional multipliers in each SOS program in Alg. 1, i.e., $L_t(s^2 + c^2 - 1)$, where L_t is a polynomial. As L_t does not multiply with any decision variables, it does not complicate the alternation scheme.

4) Observation error: To model the impact of observation error in (2) on closed-loop stability, we can add additional indeterminates w in the SOS program. For instance, given a uniformly bounded disturbance $\mathcal{W} = \|w\|_2 \leq \bar{w}$, we can enforce the Lyapunov conditions to hold robustly for all $w \in \mathcal{W}$, e.g., the first constraint of (9) becomes

$$-\frac{\partial V}{\partial a}^\top \dot{a}(\theta_k,\theta_l) + L(V-\rho) + L_w(\|w\|_2^2 - \bar{w}) \text{ is SOS}. \quad (12)$$

for SOS multipliers L_w. In general, our formulation can handle an observation error description written as a set of polynomial (in)-equalities in x, i.e., $\mathcal{W}$ is a basic semialgebraic set. In this paper, when controlling from images, we bound our keypoint extractor error using a uniform error bound $\|w\|_2 \leq \bar{w}$ valid over a set $\mathcal{A}_w \subseteq \mathcal{A}$. To over-estimate $\bar{w}$ with high probability, we can leverage extreme value theory, which estimates $\bar{w}$ from i.i.d. samples of the error $\|\hat{h}_e(h(x)) - h_k(x)\|$ for x sampled from $\mathcal{A}_w$ by following the approach laid out in [24].

5) Implicit formulation: For systems that can be modeled with rational polynomial dynamics (e.g., the cart-pole), it can be easier to determine the system's ROA by writing its dynamics in implicit form, i.e., as a set of polynomial equalities $g(a,u,\dot{a}) = 0$, which can eliminate rational terms that appear when the dynamics are written explicitly as in (1a). To adapt the Lyapunov conditions (3) to systems in implicit form, we can ensure that $\dot{V} < 0$ via additional indeterminates b and enforcing $-\frac{\partial V}{\partial a} b + L_g g(a,u,b)$ is SOS, where the Lagrange multipliers L_g are polynomials.

Specifically, the alternation scheme in Alg. 1 is modified as follows. In line 1, we modify (6) to also search for L_g:

$$\begin{array}{ll} \text{find} & V, L, L_g \\ \text{subject to} & V \text{ is SOS}, \quad L \text{ is SOS} \\ & -\frac{\partial V}{\partial a}^\top \dot{a} + L_g g + L(\|a-a_0\|_2^2 - r) \text{ is SOS}. \end{array} \quad (13)$$

Lines 2-4 of Alg. 1 remain the same. We replace line 5 of Alg. 1 with the following optimization which finds Lagrange

multipliers and the inscribed ellipsoid $\mathcal{E}$

$$\begin{aligned} \underset{E,L,L_{\text{ell}},L_g}{\text{minimize}} \quad & \text{tr}(\hat{E}^{-1}E) \\ \text{subject to} \quad & -\frac{\partial V}{\partial a}^\top b + L_g g + L(V-\rho) \text{ is SOS} \\ & L \text{ is SOS}, \quad \text{Eq. (8)}. \end{aligned} \tag{14}$$

Finally, we replace line 6 of Alg. 1 with a simultaneous search for $V, \theta_k, \theta_l, E, L_{\text{ell}}$.

$$\begin{aligned} \underset{E,V,\theta_k,\theta_l,L_{\text{ell}}}{\text{minimize}} \quad & \text{tr}(\hat{E}^{-1}E) \\ \text{subject to} \quad & -\frac{\partial V}{\partial a}^\top b + L_g g(\theta_k,\theta_l) + L(V-\rho) \text{ is SOS} \\ & \text{Eq. (8a)} \end{aligned} \tag{15}$$

B. Method 2: Gradient-based synthesis

As an alternative to running control design alternations as in Sec. V-A, we can opt to directly learn the dynamic controller parameters θ_k and θ_l through minimizing (11). That is, instead of using the gradient-based approach of Sec. V-A.1 just to obtain an initialization for Alg. 1, we can select a larger set $\mathcal{A}_s$ over which we want the closed-loop system to be stable, sample ICs from $\mathcal{A}_s$, and exactly follow the procedure in Sec. V-A.1. Given a set of learned parameters θ_k and θ_l, we can find an inner approximation of the ROA of the closed-loop system by following the alternation scheme of Alg. 1, but with the simplification that θ_k and θ_l are fixed. This is an advantage over learning an NN policy, which are more difficult to verify than our simple policies. Overall, compared to alternations (Alg. 1), this method requires more parameter tuning, i.e., of the horizon T and time-step ΔT, in order for the policy learning to reliably converge. However, for some systems, the policies obtained using GD can have a larger ROA than what is obtained with alternations (see Sec. VI for comparisons with Alg. 1 and Sec. VII for discussion).

VI. Results

We evaluate our method on an inverted pendulum, cart-pole, and planar/3D quadrotors. Our goal is to show that our method can provide controllers with large ROAs that outperform 1) popular RL-based methods for output-feedback policy synthesis, while remaining simpler and easily verifiable, and 2) the more common approach of separately synthesizing a controller and estimator and composing the two to obtain an output-feedback policy. Thus, we compare with 1) PPO [25] using a recurrent neural network (RNN) policy (to handle partial observability) and given perfect keypoint observations, and 2) full-order LQG for the linearization around the goal, which uses the separation principle [9] to independently synthesize a locally-stable full-state-feedback controller and state estimator, returning state estimate $\hat{x}$. Our SOS programs are implemented with SumOfSquares.jl [26], and the SOS constraints are interpreted with the Chebyshev basis to improve the numerical stability [5] of Alg. 1. Images for the quadrotor examples were rendered with PyBullet.

Inverted pendulum: We consider the swing-up problem for a torque-limited inverted pendulum [27, Ch. 2]. This system has state $x = [s, c, \dot{\phi}]^\top$, where s and c are the sine and cosine of the angular deviation ϕ from the upright equilibrium $x_0 = [0, 1, 0]^\top$, which we wish to stabilize to. We set the pendulum mass and length as $m = 1$kg and $\ell = 5$m, leading

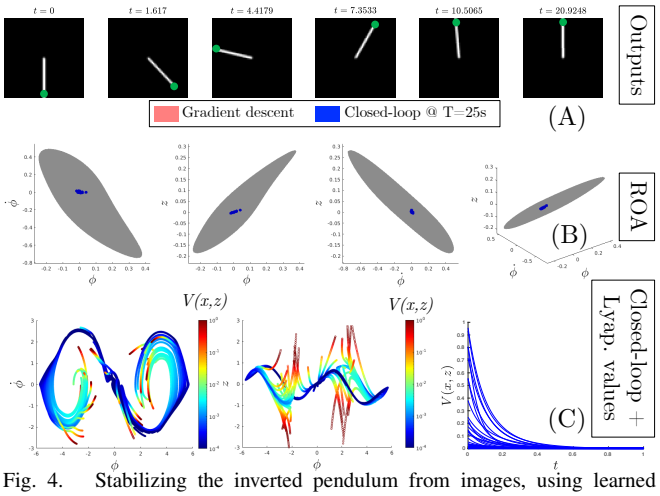

Fig. 4. Stabilizing the inverted pendulum from images, using learned keypoints y_k and a controller from *Method 2*. **(A)**: Images received when stabilizing from $a = [\phi, \dot{\phi}, z]^\top = [\pi, 0, 0]^\top$; y_k^* are marked in green. **(B)**: left to right, *Projections* of invariant set (Ω_{4e-5}) (gray) onto the $\phi\dot{\phi}$, ϕz, $\phi\dot{z}$ axes, and the full 3D invariant set. Here, Ω_{4e-5} is a sublevel set of a V which certifies global convergence to Ω_{4e-5} under observation error caused by $\hat{h}_e$; for 150 randomly sampled ICs, we plot the closed-loop states reached after 25s (blue). **(C)**: Closed-loop rollouts when controlling from images (left, center), color is V value; V along trajectories (right).

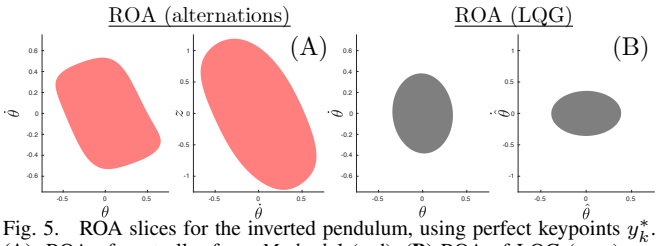

Fig. 5. ROA slices for the inverted pendulum, using perfect keypoints y_k^*. **(A)**: ROA of controller from *Method 1* (red). **(B)** ROA of LQG (gray).

to a gravity torque $mg\ell$ of 49.05 N·m, while our torque limits are 25 N·m. Thus, we cannot directly overcome gravity to swing the pendulum to x_0, and must iteratively pump energy into the system to reach x_0. For outputs, we are given a single keypoint at the tip of the pendulum, i.e., we have the observation function $y_k^* = [-\ell s, \ell c]^\top \in \mathbb{R}^2$. We train $\hat{h}_e$, represented as a CNN, from a dataset of 4000 labeled pairs of 64x64 grayscale images and corresponding keypoints. We synthesize a degree 2 dynamic-output-feedback controller, i.e., $d_k = d_l = 2$, with a single latent state $n_z = 1$, using the GD strategy (Method 2) in Sec. V-B. When controlling using the perfect keypoints y_k^*, we prove that the synthesized policy has a global ROA to a_0 using a degree 6 Lyapunov function V. When using the learned keypoints $y_k = \hat{h}_e(y)$, we bound the perception error as $\|w\|_2 \leq 0.003$, and certify global convergence to the $4 \cdot 10^{-5}$-sublevel set of V, Ω_{4e-5}, which we show in Fig. 4(B) in gray. This set is invariant, since it is compact and satisfies $\dot{V} < 0$ on its boundary. To empirically show the invariance of Ω_{4e-5}, we sample 150 ICs from Ω_{4e-5}^c and plot in Fig. 4(B) (blue) the states reached by the closed-loop system after 25s, which are all in Ω_{4e-5}. We also show snapshots of the images used to stabilize from the downward-facing equilibrium in Fig. 4(A), and example closed-loop trajectories (Fig. 4(C)). We also synthesize a controller ($d_k = d_l = 3$, $n_z = 1$) using alternations (Method 1). While this policy stabilizes near a_0 (see Fig. 5, red), it cannot swing up from the downward

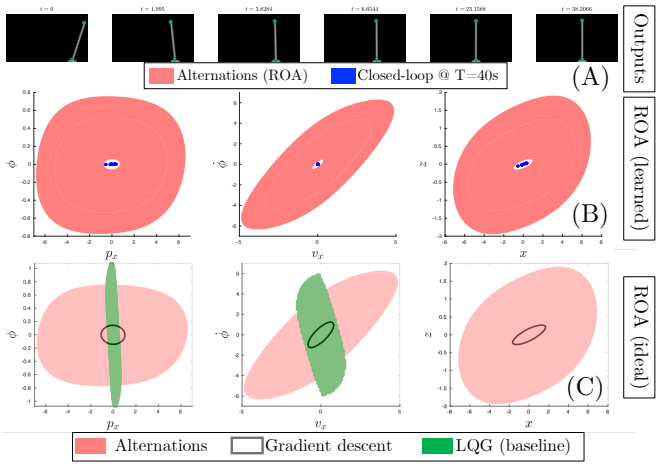

Fig. 6. Stabilizing the cart-pole using learned keypoints y_k **(A-B)** and perfect keypoints y_k^* **(C)**. **(A)**: Snapshots of images used by our alternations-based controller to stabilize from $a_0 = [5.2, -0.3, 0.6, 0.6, 1.3, -0.3]^\top$; y_k^* marked in green. **(B)**: ROA slice of our alternations-based controller when using y_k. **(C)**: ROA slices when using y_k^*. In red: Method 1 (alternations); black outline: Method 2 (gradient-based); in green: LQG.

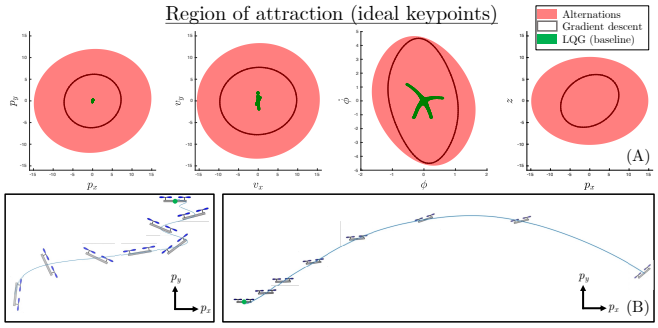

Fig. 7. Stabilizing the planar quadrotor with *perfect* keypoints y_k^*. **(A)**: ROA slices for a controller obtained via alternations (red), via GD (black), and sampled ICs in the ROA for LQG (green). **(B)**: Time-lapse of example trajectories of our closed-loop system, when controlling using the alternations-based policy driven by perfect keypoints.

equilibrium, as the alternations reach a local minimum in ROA volume. This is likely because the inscribed ellipsoid is a loose approximation of Ω_ρ for this system.

For the baselines, we evaluate PPO by sampling 50 ICs from $[\phi, \dot{\phi}] \in [-10, 10]^2$ and computing the closed-loop 2-norm from x_0 after 60s have elapsed. PPO swings up the pendulum to a neighborhood of x_0 for all 50 samples, achieving a goal error of 0.07 ± 0.03 (mean + stdev). Despite this, we note that the PPO policy does not render x_0 an equilibrium point (i.e., $u(x_0, z = 0) \neq 0$, where z is the RNN hidden state), causing persistent chattering around x_0; thus, the goal error is nonzero. For the LQG baseline, we plot its certified ROA (obtained via SOS) in Fig. 5 (gray), which is smaller than the ROA achieved by both variants of our method. We note that for LQG to stabilize, the initial state estimate must be quite accurate; in contrast, our latent state z is far less sensitive to initialization. Overall, these results suggest our method can effectively stabilize despite underactuation, that both variants of our method outperform LQG, and that our method is competitive with PPO.

Cart-pole: To show our approach can control systems with rational nonlinear dynamics, we synthesize a stabilizing policy for a cart-pole [27, Ch. 3] This system has state $x = [p_x, s, c, \dot{p}_x, \dot{\phi}]^\top \in \mathbb{R}^5$. We wish to stabilize the system around the upright equilibrium, i.e., $x_0 = [0, 0, 1, 0, 0]^\top$. Using our alternations-based approach (Method 1), we synthesize a controller, where $d_k = 4$, $d_l = 1$, and $n_z = 1$. To avoid rational terms in the explicit dynamics, we use the implicit SOS variant of Alg. 1 discussed in Sec. V-A.5. For outputs, we are given two keypoints, one at the base and one at the tip of the pole, i.e., $y_k^* = [p_x, 0, p_x - \ell s, \ell c]^\top \in \mathbb{R}^4$ (see Fig. 2). We train $\hat{h}$, represented as a CNN, using 20000 pairs of labeled 56x96 grayscale images and keypoints, and bound their error as $\|w\| \leq 0.05$, for all $a \in \Omega_{2.75}$. Under this error, we can certify using a degree 4 Lyapunov function that $\Omega_{2.75} \setminus \Omega_{0.01}$ converges to $\Omega_{0.01}$ (shown in Fig. 6(B), white), and $\Omega_{0.01}$ is an invariant set. To show this invariance empirically, we plot the states reached after 40s have elapsed in blue (Fig. 6(B)) and images seen when stabilizing from a state in $\Omega_{2.75} \setminus \Omega_{0.01}$ in Fig. 6(A). When using perfect keypoint observations y_k^*, we can certify that the entirety of $\Omega_{2.75}$ is contained in the ROA (Fig. 6(C)). We also evaluate our GD variant (Method 2) with $d_k = 4$, $d_l = 1$, and $n_z = 1$; however, we cannot effectively descend on (11), yielding a controller with a small ROA (Fig. 6(C), black).

In terms of baselines, for PPO, we attempt to learn a stabilizing policy over $[p_x, \phi, \dot{p}_x, \dot{\phi}] \in [-0.5, 0.5]^4$. While the PPO policy can reliably maintain the pole's orientation near $\phi = 0$, the closed-loop cart position p_x continuously oscillates around zero and fails to stabilize to x_0 for all ICs, leading to large goal errors 6.1 ± 0.8 after 40s have elapsed (averaged over 50 rollouts). For LQG, we plot states which empirically converge in closed-loop to x_0 (Fig. 6(C), green), since computing the ROA using SOS is prohibitive (due to there being 20 indeterminates: 5 original states x, 5 implicit variables b, and 10 for the estimates of those variables $\hat{x}$, $\hat{b}$). While the ROA of LQG is larger in the ϕ dimension, it overall has smaller volume compared to our alternations-based controller, which can overcome much larger perturbations to the cart position p_x and velocity $\dot{p}_x$. Overall, this experiment suggests that our model-based alternations approach can synthesize stronger controllers than the baselines for rational systems, and that alternations may more robustly improve the controller compared to gradient-based methods when the landscape of (11) is poorly-shaped.

Planar quadrotor: We demonstrate that our approach can stabilize a planar quadrotor from images. This system has state $x = [p_x, p_y, s, c, \dot{p}_x, \dot{p}_y, \dot{\phi}]^\top$, with dynamics as in [27, Ch. 3]. We wish to stabilize to $x_0 = [0, 0, 0, 1, 0, 0, 0]^\top$ with input limits $u \in [0, 2mg]^2 \subseteq \mathbb{R}^2$, where $m = 1$kg. We synthesize a linear dynamic controller, i.e., $d_k = d_l = 1$, with a single latent state $n_z = 1$, using alternations (Sec. V-A). For outputs, we have two keypoints, one at the center of the quadrotor, and one on the right propeller (see Fig. 2), i.e., $y_k^* = [p_x, p_y, p_x + \ell c, p_y + \ell s]^\top \in \mathbb{R}^4$. We train $\hat{h}_e$, represented as a CNN, from 40000 labeled pairs of 128x96 grayscale images and keypoints. When controlling with the ideal keypoints y_k^*, we can certify using a degree-2 Lyapunov function V that $\Omega_{1.3}$ is an inner approximation

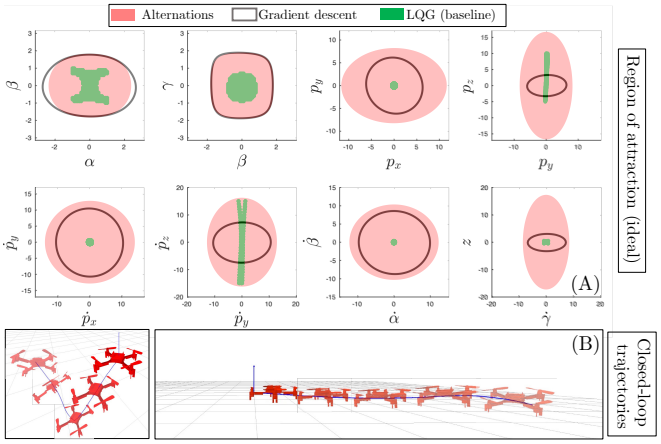

Fig. 8. Stabilizing the 3D quadrotor from *perfect* keypoints y_k^*. **(A)**: ROA slices for controller from alternations (red), from GD (black), and sampled ICs in the ROA for LQG (green). **(B)**: Time-lapse of closed-loop trajectories, when controlling using the alternations-based policy driven by y_k^*.

of the controller's ROA (shown in Fig. 7). We note that $\Omega_{1.3}$ contains states as distant as 15m from the origin and orientations beyond $\pi/2$ (Fig. 7, B). We also evaluate our GD approach (Method 2) with $d_k = d_l = n_z = 1$, which also achieves a large ROA (Fig. 7(A), black), though smaller than that which is achieved by alternations. This is because we run into difficulties in descending on (11) when expanding the ROA to distant states, due to the degradation of the landscape of (11) for long-horizon trajectories (see Sec. VII).

When controlling from images using $\hat{h}_e$, we reuse the same V and controller obtained from alternations, and aim to certify a smaller sublevel set of V, $\Omega_{0.4}$, due to the challenge of training an accurate, high-resolution keypoint extractor over a large range of states from a low-resolution image. We bound the error in the learned keypoints $\hat{h}_e(y)$ as $\|w\| \leq 0.003$ for all $a \in \Omega_{0.4}$, and certify global convergence of ICs in $\Omega_{0.4} \setminus \Omega_{0.0035}$ to $\Omega_{0.0035}$, which is an invariant set, and is plotted in Fig. 1(C), white). To empirically show the invariance of $\Omega_{0.0035}$, we plot in Fig. 1(C) (blue) the states reached after rolling out the policy for 10s, showing that states indeed reach and remain in $\Omega_{0.0035}$. We also plot example rollouts in Fig. 1(A-B) and snapshots of the images used to stabilize the system in Fig. 1(A-B) (left).

In contrast, PPO is unable to learn a stabilizing policy to x_0 over $\Omega_{1.3}$, leading to a goal error of 9.9 ± 2.9 over 25 sampled ICs. We believe this is because the controls taken by PPO rapidly destabilize the quadrotor, providing poor signal in improving the controller; reward or observation clipping could possibly improve performance. The LQG controller also has poor performance (see Fig. 7, green), for ICs where the closed-loop system is stable), and is particularly sensitive to incorrect position estimates. Like before, the ROA of LQG is prohibitive to compute using SOS due to the doubling of the number of states required by full-order state estimation. Overall, this experiment suggests that both variants of our method can yield stronger controllers than the baselines.

3D quadrotor: Finally, to demonstrate our scalability to higher-dimensional systems, we synthesize a stabilizing visuomotor policy for a full 3D quadrotor. This system has state $x = [q_w, q_x, q_y, q_z, p_x, p_y, p_z, \dot{p}_x, \dot{p}_y, \dot{p}_z, \dot{\alpha}, \dot{\beta}, \dot{\gamma}]^\top \in$

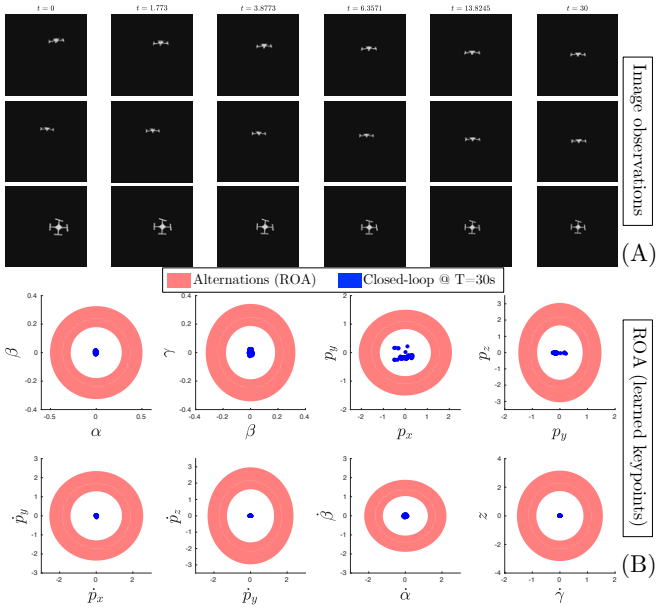

Fig. 9. Stabilizing the 3D quadrotor from *learned* keypoints y_k, using our alternations-based controller. **(A)**: snapshots of the three-view 128x128x3 images provided to the controller. **(B)**: verified ROA under keypoint error (red), which is guaranteed to converge to the white invariant set. We also plot (blue) the states reached in closed-loop, to show empirical invariance.

$\mathbb{R}^{13}$, where the quaternion $q_{(\cdot)}$-represented dynamics are given in [28] (we enforce the unit quaternion constraint via the S-procedure). We wish to stabilize to the origin $x_0 = [1, \mathbf{0}_{12}]^\top$, with input limits $u \in [0, 2.5mg/4]^4 \subseteq \mathbb{R}^4$. We synthesize a linear dynamic controller, i.e., $d_k = d_l = 1$, with a single latent state $n_z = 1$, using alternations (Method 1). For outputs, we have three keypoints, one on three of the four propellers (see Fig. 2), i.e., $y_k \in \mathbb{R}^9$. We train $\hat{h}_e$, represented as a CNN, from 150000 labeled pairs of 128x128x3 depth images and keypoints. Here, we stack three depth images, each recorded at different angles, offering views of the p_xp_z, p_xp_y, and p_yp_z planes. When controlling with the ideal keypoints y_k^*, we certify using a degree-2 Lyapunov function V that $\Omega_{0.3}$ is contained in the controller's ROA (shown in Fig. 8). We note that $\Omega_{0.3}$ contains states as distant as 15m from the origin and yaw/pitch/roll angles well beyond $\pi/2$ (Fig. 7, B). We also evaluate GD (Method 2) with $d_k = d_l = n_z = 1$, which also achieves a large ROA (Fig. 8(A), black), which while overall smaller in volume than the alternations ROA, has a larger ROA in the orientation states.

When controlling from images using $\hat{h}_e$, we reuse the same V and controller from Method 1, and aim to certify a smaller sublevel set of V, $\Omega_{0.01}$ (shown in Fig. 9(B)), for the same reasons as for the planar quadrotor. We bound the error in the learned keypoints $\hat{h}_e(y)$ as $\|w\| \leq 0.003$ for all $a \in \Omega_{0.01}$, and certify global convergence of ICs in $\Omega_{0.01} \setminus \Omega_{0.003}$ to $\Omega_{0.003}$, which is an invariant set, and is plotted in Fig. 9(B), white). To empirically show the invariance of $\Omega_{0.003}$, we plot in Fig. 9(B) (blue) the states reached after rolling out the policy for 30s, showing that states indeed reach and remain in $\Omega_{0.003}$. We plot some images received along an example stabilization in Fig. 9(A).

In contrast, PPO is unable to learn a stabilizing policy to x_0 for ICs sampled from $\Omega_{0.01}$, leading to a goal error

of 8.6 ± 1.3 over 25 ICs; we believe it fails for similar reasons as the planar quadrotor. The LQG controller also has poor performance (see Fig. 8, green, where we plot sampled ICs where the closed-loop system is stable), and is highly sensitive to incorrect estimates in several state dimensions. Like before, the ROA of LQG is prohibitive to compute using SOS due to the doubling of states caused by full-order state estimation. Overall, this experiment suggests that both variants of our method can provide controllers with large ROA, even for high-dimensional systems, and provides huge computational savings compared to synthesizing/certifying a full-dimensional state estimator together with the controller.

VII. Discussion and Conclusion

We present two methods for synthesizing provably-stable reduced-order dynamic-output-feedback controllers from images: the first synthesizes stable-by-construction policies via bilinear alternations, and the second optimizes the controller via GD and certifies an ROA *a posteriori*. Our method stabilizes several systems more effectively than baselines.

Pros/cons of Method 1: Alternation provides several benefits. For the full-order case (i.e., $n_z = n_x$), if the linearization around the goal is stabilizable and detectable, we can reliably initialize Alg. 1 with the controller/observer from LQG. Moreover, for both the full- and reduced-order cases, Alg. 1 monotonically increases the (ellipsoidal approximation of the) ROA volume. This systematic controller improvement makes Method 1 well-suited for expanding the ROA to distant regions of $\mathcal{A}$. The scalability of Method 1 is also similar to that of state-feedback synthesis for the systems in Sec. VI, since a scalar z is sufficient for stabilization. The biggest drawback of Method 1 is the numerical instability of Alg. 1, though we find that orthogonal polynomial bases (like Chebyshev) greatly improve the numerics. Method 1 is also prone to local minima, especially when there is a large gap between the volume of Ω_ρ and its inscribed ellipsoid, i.e., for the inverted pendulum, which has a "spiral"-shaped ROA (Fig. 4(C)) that is poorly approximated by an ellipsoid.

Pros/cons of Method 2: GD is more scalable than SOS in high dimensions, though we may still need to solve a large SOS program to verify the controller. Method 2 also sidesteps local minima caused by the ellipsoid volume objective in Alg. 1. Overall, GD is quite reliable for obtaining an initial controller with a small ROA around the goal. For drawbacks, Method 2 may not find a controller with certifiable ROA (as verification is done post-hoc), though we find empirically that closed-loop stability on sampled ICs generalizes well to stability on nearby ICs, due to the simplicity of our controller. There are also many parameters which must be tuned for success. Moreover, the landscape of (11) often has high Lipschitz constant, especially when T is large [3], leading to myopic gradients that do not effectively descend (11). This makes it difficult to expand the ROA to distant parts of $\mathcal{A}$ (as in the cart-pole and quadrotor examples). We believe these issues are not tied to the polynomial policy parameterization, as we also tried GD on an NN policy and observed the same issues. These issues can be mitigated by damping the system or using a critic to reduce the effective horizon required [3].

Future work: We require labeled images and keypoints to train $\hat{h}_e$; to remove this limitation, we will explore unsupervised learning of polynomial latent dynamics from images, which would be amenable to SOS-based synthesis and verification tools.

References

[1] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *JMLR*, vol. 17, pp. 39:1–39:40, 2016.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, 2017.

[3] J. Xu, V. Makoviychuk, Y. S. Narang, F. Ramos, W. Matusik, A. Garg, and M. Macklin, "Accelerated policy learning with parallel differentiable simulation," in *ICLR*, 2022.

[4] E. Banijamali, R. Shu, M. Ghavamzadeh, H. Bui, and A. Ghodsi, "Robust locally-linear controllable embedding," in *AISTATS*, 2018.

[5] G. Blekherman, P. A. Parrilo, and R. R. Thomas, *Semidefinite Optimization and Convex Algebraic Geometry*. SIAM, 2012.

[6] A. Majumdar, A. A. Ahmadi, and R. Tedrake, "Control design along trajectories with sums of squares programming," in *ICRA*, 2013.

[7] S. Dean, A. J. Taylor, R. K. Cosner, B. Recht, and A. D. Ames, "Guaranteeing safety of learned perception modules via measurement-robust control barrier functions," in *CoRL*, 2020.

[8] D. Henrion and M. Korda, "Convex computation of the region of attraction of polynomial control systems," *TAC*, vol. 59, no. 2, 2014.

[9] K. Åström, *Introduction to Stochastic Control Theory*. Dover, 2006.

[10] D. Bernstein and D. Hyland, "The optimal projection equations for fixed-order dynamic compensation of distributed-parameter systems," in *Structures, Structural Dynamics and Materials*, 1984.

[11] A. Isidori and A. Astolfi, "Disturbance attenuation and $\mathcal{H}_\infty$-control via measurement feedback in nonlinear systems," *TAC*, vol. 37, 1992.

[12] A. Astolfi and P. Colaneri, "A hamilton-jacobi setup for the static output feedback stabilization of nonlinear systems," *IEEE Trans. Autom. Control.*, vol. 47, no. 12, pp. 2038–2041, 2002.

[13] S. Baldi, "An iterative sum-of-squares optimization for static output feedback of polynomial systems," in *CDC*, 2016.

[14] S. K. Nguang, M. Krug, and S. Saat, "Nonlinear static output feedback controller design for uncertain polynomial systems: An iterative sums of squares approach," in *IEEE IEA*, 2011.

[15] W. Tan, "Nonlinear control analysis and synthesis using sum-of-squares programming," in *PhD thesis, UC Berkeley*, 2006.

[16] Q. Zheng and F. Wu, "Nonlinear output feedback $\mathcal{H}_\infty$ infinity control for polynomial nonlinear systems," *ACC*, pp. 1196–1201, 2008.

[17] I. R. Manchester and J. E. Slotine, "Output-feedback control of nonlinear systems using control contraction metrics and convex optimization," in *Australian Control Conference*. IEEE, 2014.

[18] C. Dawson, B. Lowenkamp, D. Goff, and C. Fan, "Learning safe, generalizable perception-based hybrid control with certificates," *IEEE Robotics Autom. Lett.*, vol. 7, no. 2, pp. 1904–1911, 2022.

[19] G. Chou, N. Ozay, and D. Berenson, "Safe output feedback motion planning from images via learned perception modules and contraction theory," in *WAFR*, 2022.

[20] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," in *CoRL*, 2018.

[21] A. Laguna, E. Riba, D. Ponsa, and K. Mikolajczyk, "Key.net: Keypoint detection by handcrafted and learned CNN filters," in *ICCV*, 2019.

[22] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "KPAM: keypoint affordances for category-level robotic manipulation," in *ISRR*, 2019.

[23] M. Fazel, H. A. Hindi, and S. P. Boyd, "Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices," in *ACC*. IEEE, 2003, pp. 2156–2162.

[24] C. Knuth, G. Chou, J. Reese, and J. Moore, "Statistical safety and robustness guarantees for feedback motion planning of unknown underactuated stochastic systems," in *ICRA*, 2023.

[25] A. Raffin, https://github.com/DLR-RM/rl-baselines3-zoo.

[26] B. Legat, C. Coey, R. Deits, J. Huchette, and A. Perry, "Sum-of-squares optimization in Julia," in *JuMP-dev Workshop*, 2017.

[27] R. Tedrake, "Underactuated robotics: Learning, planning, and control for efficient and agile machines," *Course notes for MIT 6.832*, 2009.

[28] E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor," in *ECC*. IEEE, 2013, pp. 3864–3869.