Systematic, Lyapunov-Based, Safe and Stabilizing Controller Synthesis for Constrained Nonlinear Systems

Reza Lavaei, Member, IEEE, and Leila J. Bridgeman, Member, IEEE

Abstract—A controller synthesis method for state- and input-constrained nonlinear systems is presented that seeks continuous piecewise affine (CPA) Lyapunov-like functions and controllers simultaneously. Nonconvex optimization problems are formulated on triangulated subsets of the admissible states that can be refined to meet primary control objectives, such as stability and safety, alongside secondary performance objectives. A multistage design is also given that enlarges the region of attraction (ROA) sequentially while allowing different performance metrics for each stage. A boundary for the closed-loop system’s ROA is obtained from the resulting Lipschitz Lyapunov function. For control–affine nonlinear systems, the nonconvex problem is formulated as a series of conservative, but convex, well-posed optimization problems. These iteratively decrease the cost function until the design objectives are met. Since the resulting CPA Lyapunov-like functions are also Lipschitz control (or barrier) Lyapunov functions, they can be used in online quadratic programming to find minimum-norm control inputs. Numerical examples are provided to demonstrate the effectiveness of the method.

Index Terms—Constrained control, linear matrix inequalities (LMIs), optimization, safety, stability of nonlinear systems.

I. INTRODUCTION

LYAPUNOV theory has been instrumental to stable [1], [2], [3], [4], [5], [6], [7], [8], [9] and safe [10], [11], [12], [13], [14], [15], [16] control design. For linear systems, it provides straightforward stability criteria for analysis and design, since the existence of a Lyapunov function can be assured or denied simply by solving a set of linear matrix inequalities, but there is no systematic way to ensure Lyapunov stability for general, nonlinear systems. When physical limitations or operating considerations constrain the state and inputs, stability and feasibility must be considered in tandem, further complicating analysis. Ignoring constraints at best results in unexpected closed-loop behavior if the constraints are activated by physical boundaries, and at worst is hazardous if the system operates in unsafe regions. By limiting the controller and Lyapunov functions to a particular class, this article formulates controller synthesis for state- and input-constrained nonlinear systems as an offline optimization problem on a triangulated subset of the admissible states.

Lyapunov stability is verified by existence of a Lyapunov function and safety (state and input constraint adherence), is ensured if the system begins in a sublevel set of the Lyapunov function inside the set of admissible states where the control constraints are also respected. Many methods, therefore, seek out Lyapunov functions alongside controllers. However, decrease conditions on the Lyapunov function inevitably create nonconvex constraints involving products of unknown Lyapunov function and controller parameter, which introduce computationally taxing optimization problems. Therefore, most methods either find the Lyapunov-like function after controller design, or select it a priori [9], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23]. For instance, given a stabilizing controller, [20] can be used to find a continuous piecewise affine (CPA) Lyapunov function using a linear program (LP). While this provides a powerful stability analysis tool, it does not provide means to synthesize the required stabilizing controller. Conversely, given a continuously differentiable control Lyapunov function (CLF), Steentjes et al. [24] provided means to find a stabilizing CPA controller using an LP. However, it may not be easy to find a continuously differentiable CLF. Further, if Steentjes et al. [24] produced a controller with unsatisfactory performance, there may be a better controller and Lyapunov function pair that does satisfy the performance criteria. In either example, we are left with the challenge of choosing a stabilizing controller or CLF a priori.

Several methods avoid the problem of a priori controller or CLF design. For instance, model predictive control (MPC) mitigates the issue by only requiring a terminal stabilizing controller, but taxing invariant set computations are often needed and additional computations are needed to find the associated Lyapunov function through explicit nonlinear MPC [18], [19], [21]. Alternatively, Baier and Hafstein [25] formulated the simultaneous search for a CPA CLF, and controller as a mixed integer linear program (MILP). This avoids a priori design choices, but introduces an $\mathcal{NP}$-hard problem. Further, the resulting controller is constant on each simplex of a triangulation and,
therefore, discontinuous, which can be undesirable in practice. In this article, we give a method to search for controllers and Lyapunov functions in tandem for constrained state and input nonlinear systems. For input affine systems, a CPA controller and Lyapunov function pair is found using a sequence of well-posed, semidefinite programs (SDPs), which can be solved in polynomial time [26]. The proposed algorithms provide the region of attraction (ROA) and safe sets, and characterize the convergence rate. Like many other works, the proposed algorithms begin with candidate Lyapunov functions and controllers, but the major difference arises because even if the candidates are chosen poorly—even randomly—the proposed algorithms can improve them, thereby alleviating a priori design choices.

While both this article and Baier and Hafstein [25] give a stabilizing controller, this article aims to provide a more detailed control synthesis method that provides a practical controller and characterizes closed-loop performance. The deployed techniques are also different, where Baier and Hafstein [25] considered a finite set of possible inputs and solve an MILP to find discontinuous inputs, here iterative, but well-posed SDPs are used to find CPA controllers. To begin, a nonlinear program, which is feasible by construction, is posed. Then, a feasible candidate pair of controller and Lyapunov function is given that may not ensure stability. A sequence of SDPs select new candidates that iteratively reduce the degree to which stability criteria are violated. Once stability and desired closed-loop convergence rate for the state norm are met, a pair of stabilizing controller and Lyapunov function is provided. Because some of the feasible region may not be stabilizable, the ROA is found. At this point, existing CLF-based methods can be used to seek out controllers that improve performance in this ROA. In particular, Steentjes et al. [24] can accomplish this with an LP. However, these typically keep the Lyapunov function constant, which could limit performance. In this case, Algorithm 1 modifies both the controller and Lyapunov function via iterative SDPs, and Theorem 6 establishes that it improves the performance while preserving stability and the established convergence rate of the state norm. Moreover, two algorithms to increase the ROA are also provided.

This work improves upon preliminary results [27] that were focused on exponential stability by confronting safety and secondary performance objectives, increasing the ROA, and warm-starting triangulation refinement. Numerical examples demonstrate the effectiveness of the method.

II. PRELIMINARIES

Notation: The interior, boundary, and closure of \( \Omega \in \mathbb{R}^n \) are denoted by \( \Omega^\circ, \partial \Omega, \Omega, \) respectively. The set of all compact subsets \( \Omega \subset \mathbb{R}^n \) satisfying \( \Gamma \) \( \Omega^\circ \) is connected and contains the origin, and \( \Omega = \overline{\Omega^\circ} \), is denoted by \( \mathbb{R}^n \). The vectors of ones and zeros in \( \mathbb{R}^n \) are, respectively, denoted by \( \mathbf{1}_n \) and \( \mathbf{0}_n \). The set of integers between \( a \) and \( b \) inclusive is \( Z_{a,b} \). The \( i \)th element of a vector \( x \) is denoted by \( x^{(i)} \). The element in the \( i \)th row and \( j \)th column of a matrix \( G \) is denoted by \( G^{(i,j)} \). Its \( i \)th row is denoted by \( G^{(i,:)} \). The transpose and Euclidean norm of \( x \in \mathbb{R}^n \) are denoted by \( x^T \) and \( \|x\| \), respectively. Maximums, inequalities, and absolute values are taken elementwise and \( \preceq \) denotes negative semidefiniteness. The preimage of a function \( f \) with respect to a subset \( \Omega \) of its codomain is defined by \( f^{-1}(\Omega) = \{x \mid f(x) \in \Omega\} \).

In constrained systems, safety can be ensured in a positive-invariant subset of the feasible region for autonomous systems and a control-invariant set for controlled systems.

Definition 1 (Control invariance and positive invariance [28, Ch. 11]): A set \( A \subseteq \mathcal{X} \) is control-invariant with respect to dynamics and constraints if \( \dot{x} = g(x,u) \), \( x \in \mathcal{X} \subset \mathbb{R}^n \), and \( u \in U \subset \mathbb{R}^m \), if \( x(t_0) \in A \) implies existence of \( u^* : A \rightarrow U \) such that \( x(t) \in A \) for all \( t > t_0 \) if \( \dot{x} = g(x; u^*(x)) \). For autonomous dynamics, \( \dot{x} = g(x) \), such a set is positive invariant.

An ROA is a positive-invariant set in which \( x(t) \rightarrow 0 \) as \( t \rightarrow +\infty \). In this article, sublevel sets of Lipschitz Lyapunov-like functions are used to find safe sets and/or ROAs. These functions will be constructed on a triangulated subset of \( \mathbb{R}^n \). The required definitions are given next.

Definition 2 (Affine independence [20]): A collection of vectors \( \{x_0, \ldots, x_n\} \subset \mathbb{R}^n \) is called affinely independent if \( x_1 = x_0, \ldots, x_n = x_0 \) are linearly independent.

Definition 3 (n-simplex [20]): An \( n \) simplex is the convex hull of \( n + 1 \) affinely independent vectors in \( \mathbb{R}^n \) called vertices, denoted \( \sigma = \{x_j^{(1)}\}_{j=0}^n \).

In this article, simplex always refers to \( n \) simplex. By abuse of notation, \( T \) will refer to both a collection of simplexes and the set of points in all the simplexes of the collection.

Definition 4 (Triangulation [20]): A set \( T \subset \mathbb{R}^n \) is called a triangulation if it is a finite collection of \( m_T \) simplexes, denoted \( T = \{\sigma_i\}_{i=1}^{m_T} \), and the intersection of any of the two simplexes in \( T \) is either a face or the empty set.

The following two conventions are used throughout this article for triangulations and their simplexes. Let \( T = \{\sigma_i\}_{i=1}^{m_T} \). Further, let \( \{x_{i,j}^{(0)}\}_{j=0}^n \) be \( \sigma_i \)’s vertices, making \( \sigma_i = \sigma(\{x_{i,j}^{(0)}\}_{j=0}^n) \). The choice of \( x_{i,0} \) in \( \sigma_i \) is arbitrary unless \( 0 \in \sigma_i \), in which case \( x_{i,0} = 0 \). The vertices of the triangulation \( T \) that are in \( \Omega \subseteq \Omega \) is denoted by \( \overline{\Omega} \).

Definition 5 (CPA interpolation [20]): Consider a triangulation \( T = \{\sigma_i\}_{i=1}^{m_T} \), and a set \( \mathcal{W} = \{W_x\}_{x \in \mathbb{E}_x} \subset \mathbb{R}^d \). The unique, CPA interpolation of \( \mathcal{W} \) on \( T \) satisfying \( W_x(x) = W_x \forall x \in \mathbb{E}_T \), denoted \( W : T \rightarrow \mathbb{R}^d \), is \( W(x) = x^T \nabla W_i + \omega_i \), where \( \nabla W_i \) is linear in \( W \) and can be computed as follows. Let \( \sigma_i = \sigma(\{x_{i,j}^{(0)}\}_{j=0}^n) \), and \( x_j \in \mathbb{E}_{n \times n} \) be a matrix that has \( x_{i,j} = x_{i,0} \) as its \( j \)th row. By definition of affine independence, these rows are linearly independent, so \( x_j \) is invertible. Let \( W_j \in \mathbb{E}_{n}^{n} \) be a vector that has \( W_{x_{i,j}} - W_{x_{i,0}} \) as its \( j \)th element. Then, \( \nabla W_i = x_j^{-1} W_j \).

The set of real-valued functions with \( r \) times continuously differentiable partial derivatives over their domain is denoted by \( C^r \). A function is piecewise in \( C^2 \) on a triangulation \( T \), if it is in \( C^2 \) on all simplexes. When taking derivatives, if \( \xi \) is on the common face of some simplexes, the surrounding notation, \( \xi \in \sigma \), will clarify that the related limits should be evaluated in directions, \( y \in \mathbb{R}^n \), where \( \xi + h y \in \sigma \) as \( h \to 0 \). The Dini derivative of \( V \) at \( x \) is denoted \( D^+V(x) = \limsup_{h \to 0^+} (V(x+h \xi(x)) - V(x)) \), which equals \( V'(x) \) if \( V \in C^1 \). For \( g \in C^k \) on \( \Omega \subset \mathbb{R}^n \) and \( \nu = (v_1, \ldots, v_k) \in Z_2 \times \cdots \times Z_2 \), the partial
derivatives at \( x = \xi \) will be denoted

\[
D^\nu g(\xi) = \left. \frac{\partial^k g}{\partial x^{(\nu)}_i \ldots \partial x^{(\nu)}_k} \right|_{x=\xi}
\]

The following theorem from [20] bounds the time derivative of a CPA function above on a simplex using its values at the vertices of that simplex using Taylor’s theorem.

**Lemma 1 ([20]):** For \( g : \mathbb{R}^n \to \mathbb{R}^m \) in \( C^2 \), consider

\[
\dot{x} = g(x), \quad x \in \mathcal{X} \backslash \mathcal{R}^m.
\]

Let \( \mathcal{T} = \{\sigma_i\}_{i=1}^m \subset \mathcal{X} \) be a triangulation and \( V : \mathcal{T} \to \mathbb{R} \) be the CPA interpolation of \( V = \{V_x\}_{x \in \mathbb{R}^T} \). For any \( x \in \mathcal{T} \), there exists a \( \alpha_i = \{x_{ij}\}_{j=0}^n \in \mathcal{T} \) so that for small enough \( h > 0 \), \( co(x, x + h g(x)) \subset \sigma_i \). Let \( \alpha \) be the unique vector satisfying \( x = \sum_{j=0}^n \alpha_i x_{ij} \), where \( 0 < \alpha \leq \alpha_i \leq 1 + \frac{1}{2} \). Then

\[
D^+ V(x) \leq \frac{n}{2} \max_{k \in Z^0_1} \left( \max_{0 \leq k \leq 1} \left[ \|x_{i,k} - x_{i,0}\| + \|x_{i,j} - x_{i,0}\| \right] \right)
\]

**III. CONTROLLER CHARACTERIZATION**

The goal is to turn the analysis method of [20] into a design method for state- and input-constrained control systems by finding a state-feedback controller. While an optimization problem could simply be derived by directly applying [20, Th. 1] to the closed-loop of a plant and a parameterized controller structure, this does not readily lead to a well-posed, convex optimization problem and a synthesis method. Consequently, the theorems that follow parallel those of [20, Th. 1] with appropriate modifications. These changes are critical to the proposed iterative methods of Section V.

**Theorem 1:** Consider the system

\[
\dot{x} = g(x, u), \quad x \in \mathcal{X} \subset \mathbb{R}^n, \quad u \in \mathcal{U} = \mathcal{R}^m, \quad g(0, 0) = 0.
\]

Given a set \( \Omega \subset \mathcal{X} \), where \( \Omega \subset \mathbb{R}^n \), let \( \mathcal{T} = \{\sigma_i\}_{i=1}^m \) be its triangulation. Suppose that a class of Lipschitz controllers \( \mathcal{F} = \{u(\cdot, r)\}_{r \in \mathbb{R}^m} \) parameterized by \( r \) has at least one element and satisfies \( u(0, \lambda) = 0, u(\cdot, \lambda) \in C^2(T) \), and \( g(x, \cdot) := g(x, u(\cdot, \lambda)) \in C^2(T) \) is Lipschitz on \( \mathcal{T} \), and \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \) \( \forall x \in \mathcal{E_T} \) implies \( u(x, \lambda) \in \mathcal{U} \)

\[
\begin{align*}
\alpha &\geq 1, \quad b_1 > 0 \\
b_1 \|x\|_r^a &\leq V_x & \quad & \forall x \in \mathcal{E_T} \quad (5a) \\
|\nabla V_i| &\leq l_i & \quad & \forall i \in Z^r_1 \quad (5b) \\
u(x_{ij}, \lambda) &\in \mathcal{U} & \quad & \forall i \in Z^r_1, \forall j \in Z^n_0 \quad (5c) \\
D^+ V_{i,j} &\leq -b_2 V_{x_{ij}} & \quad & \forall i \in Z^r_1, \forall j \in Z^n_0 \quad (5d)
\end{align*}
\]

\[
D^+ V_{i,j} = g_k(x_{ij})^T \nabla V_i + c_{i,j} \beta_i \lambda_i \quad (5e)
\]

The inequalities (5) are feasible, and the CPA function \( V : \mathcal{T} \to \mathbb{R} \), constructed from \( V \), satisfies \( b_1 \|x\|_r^a \leq V(x) \) and \( D^+ V(x) \leq -b_2 V(x) \) for all \( x \in \mathcal{T} \).

**Proof:** To see that (5) is feasible, note that \( V_x = b_1 \|x\|_a \) with any \( a \geq 1 \) and \( b_1 > 0 \) satisfies (5a) and (5b) and can be used to compute a feasible solution \( l_i = |\nabla V_i| \) for (5c). By assumption, a feasible \( \lambda \) exists satisfying (5d). Using these feasible values, finite \( \beta_i \) satisfying (6) can be chosen and \( g_k(\cdot) \) is always finite because \( g_k(\cdot) \in C^2(T) \). Likewise, \( c_{i,j} \) is finite because each \( \sigma_i \) is compact, making the left-hand side of (5e) finite for each \( i \in Z^r_1 \) and \( j \in Z^n_0 \). Note that if \( x_{ij} = 0 \), then \( g_k(x_{ij}) = 0 \) and by convention, \( j = 0 \), so \( c_{i,j} = 0 \), making \( D^+ V_{i,j} = 0 \), making any \( b_2 \) feasible. Thus, there exists \( b_2 \in \mathcal{R} \) that satisfies (5e) for all \( i \in Z^r_1 \) and \( j \in Z^n_0 \).

In the next portion of the proof is devoted to showing that for the closed-loop system, \( \dot{x} = g_k(x) \), the solution \( V(x) \) satisfies \( b_1 \|x\|_a \leq V(x) \) and \( D^+ V(x) \leq -b_2 V(x) \) for all \( x \in \mathcal{T} \). By assumption, (5d) implies \( u(x, \lambda) \in \mathcal{U} \) for all \( x \in \mathcal{T} \). Constraints (5a) and (5b) ensure \( b_1 \|x\|_a \leq V(x) \) for all \( x \in \mathcal{T} \) since \( V : \mathcal{T} \to \mathbb{R} \) is a CPA function. It remains to show that (5c) and (5e) verify \( D^+ V(x) \leq -b_2 V(x) \) for all \( x \in \mathcal{T} \). For simplicity, let \( g(x) = g(x, u(\lambda, x)) \). The assumptions of Lemma 1 are verified by (5c) and (6). Applying (2) and (5e), and the fact that \( V(x) \geq 0 \) is affine on each \( \sigma_i \), shows that \( D^+ V(x) \leq \sum_{j=0}^n x_{ij} \alpha^{(j)} D^+ V_{i,j} \leq \sum_{j=0}^n \alpha^{(j)} D^+ V_{i,j} \leq -b_2 V(x) \), where \( x = \sum_{j=0}^n x_{ij} \in \mathcal{T} \) is arbitrary, and \( \alpha \) is the corresponding unique vector satisfying \( \alpha \beta = 1 \) and \( 0 < \alpha \leq \alpha_i \leq 1 \). Like [24], as a relaxation of Theorem 1, it is assumed that \( g_k(\cdot) \in C^2(T) \), not everywhere. Since \( x \in \mathcal{T} \) was an arbitrary point, \( D^+ V(x) \leq -b_2 V(x) \) for all \( x \in \mathcal{T} \) is verified.

**Remark 1:** To ensure a Lyapunov-like function is decreasing along all trajectories when seeking a solution to (5), \( b_2 > 0 \) is needed. If \( b_2 < 0 \), but there exists a subtriangulation \( \mathcal{T} \subset \mathcal{T} \), satisfying \( \mathcal{T} = \{\sigma_i\}_{i=1}^m \), where \( \mathcal{I} = \{i \in Z^r_1 \mid D^+ V_{i,j} < 0 \} \), then Theorem 1 can immediately be applied on \( \mathcal{T} \), replacing \( b_2 \) with \( \hat{b}_2 := \min \{\|D^+ V_{i,j} V_{-1,i,j} \mid i \in \mathcal{I}, j \in Z^n_0, x_{ij} \neq 0 \} \} \) to ensure the same properties on a smaller region.

**A. CPA Controller for Control-Affine Systems**

In practice, it may not be obvious how to apply Theorem 1 for control design. For one, finding a control structure in which point-wise feasibility on vertices of a triangulation implies feasibility at all points in the triangulation is not trivial. Moreover, once the control structure is chosen, its first and second derivatives may need to be constrained to compute \( \beta_i \) in (6). A more
practical characterization of (5) for control-affine systems with polytopic input constraints is given next.

**Instantiation 1:** Consider the constrained control system
\[
\dot{x} = f(x) + G(x)u, \quad x \in \mathcal{X} \subseteq \mathbb{R}^n, \quad u \in \mathcal{U} \subseteq \mathbb{R}^m
\]
where \( \mathcal{U} = \{ u \in \mathbb{R}^m \mid Hu \leq h_u \} \). Given a set \( \Omega \subseteq \mathcal{X} \) and its triangulation, \( T = \{ \sigma_i \}_{i=1}^{m_f} \), suppose that both \( f(\cdot), G(\cdot) \in C^2(T) \). Let \( u \) be CPA on \( T \), i.e., \( u: T \to \mathbb{R}^m \), where \( u^{(s)}_i = x^T \nabla u^{(s)}_i + u^{(s)}_i \forall s \in Z_i^m \forall i \in Z_i^m \). Let \( y = [V, I, u, z, a, b] \) be the unknowns, where \( V = \{ V_x \}_{x \in \mathcal{X}} \subseteq \mathbb{R}^n_1 \), \( I = \{ I_i \}_{i=1}^{m_f} \subset \mathbb{R}^n_1 \), \( u = \{ u_x \}_{x \in \mathcal{X}} \subseteq \mathbb{R}^m_1 \), \( z = \{ z_i \}_{i=1}^{m_f} \subseteq \mathbb{R} \), \( a \in \mathbb{R} \), and \( b = \{ b_1, b_2 \} \subset \mathbb{R} \). The following inequalities imply (5):
1. \( a \geq 1, b_1 > 0 \)
2. \( b_1 \|x\|^a \leq V_x \quad \forall x \in \mathcal{X} \)
3. \( \|\nabla V_x\| \leq l_i \quad \forall i \in Z_i^m \)
4. \( H u_x \leq h_e \quad \forall x \in \mathcal{X} \)
5. \( \|\nabla u^{(s)}_i\| \leq z_i \quad \forall i \in Z_i^m \)
6. \( D^t V_x \leq -b_2 V_x \quad \forall x \in Z_0 \)

Further, let \( A = V^{-1}(0, r) \subseteq \mathcal{R}^n \) for some \( r > 0 \). Then, the origin is locally exponentially stable on \( \mathcal{A} \), that is \( \|x(t)\| \leq \sqrt[r]{e^{-\frac{b_1}{r}(t-t_0)}} \forall x(t) \in \mathcal{A} \).

**Proof:** Lyapunov stability is verified by [20, Def. 2, Remarkful 5]. Since \( A \subseteq \mathcal{R}^n \), \( x(t_0) \in \mathcal{A} \) implies \( V(x(t)) \leq r \) and \( x(t) \in \mathcal{A} \) holds for all \( t \geq t_0 \). By the Comparison Lemma [30, Lemma 3.4], \( V(x(t)) \leq V(x(t_0))e^{-b_2(t-t_0)} \) for all \( t \geq t_0 \). So, \( \|x(t)\| \leq \sqrt[r]{e^{-\frac{b_1}{r}(t-t_0)}} \).

The following theorem modifies Instantiation 1 to find a stabilizing controller for (7), using Lemma 2.

**Theorem 2 (Stabilization):** In Instantiation 1, let \( \Omega \subseteq \mathcal{R}^n \). Assume that \( 0 \in \mathcal{U} \) and \( f(0) = 0 \). Let (8) be augmented with the constraints \( V_0 = u_0 = 0 \). If \( b_2 > 0 \), then \( V: T \to \mathcal{R} \) is the closed-loop system’s CPA Lyapunov function. Further, let \( A = V^{-1}(0, r) \subseteq \mathcal{R}^n \) for some \( r > 0 \). If \( x(t_0) \in \mathcal{A} \), then \( \|x(t)\| \leq \sqrt[r]{e^{-\frac{b_1}{r}(t-t_0)}} \) for all \( t \geq t_0 \).

**Proof:** Using Corollary 1, \( b_1 \|x\|^a \leq D^t \|x\|^a(V(x) + D^t V(x) \leq b_2 V(x) \) holds for all \( x \in \Omega \), verifying (10a) and (10b). Lemma 2’s other conditions are satisfied by \( \Omega \subseteq \mathcal{R}^n \), \( b_2 > 0 \), \( f(0) = 0 \), and \( V(0) = u(0) = 0 \).

Note that \( \mathcal{A} \) and \( \mathcal{A}^1 \) in Lemma 2 and Theorem 2 are a positive-invariant set and a ROA, respectively, since \( \mathcal{A} \) is a subset of the Lyapunov function, making the restriction of \( V \) to \( \mathcal{A}^1 \) a CLF, as described in the following.

**Definition 6 (CLF [25]):** The function \( V: \Omega \to \mathcal{R} \), where \( \Omega \subseteq \mathcal{R}^n \), is called a CLF for (7) if \( \exists a, b_1, b_2 > 0 \) satisfying
\[
\begin{align*}
   & b_1 \|x\|^a \leq V(x) \quad \forall x \in \Omega \quad \text{(11a)}
   \quad \inf_{u \in \mathcal{U}} (D^t V(x) + b_2 V(x)) \leq 0 \quad \text{(11b)}
\end{align*}
\]
for all \( x \in \Omega \), where \( D^t V(x) = \nabla V(x) \). \( f(x) + G(x)u \).

**Proof:** If \( b_2 > 0 \) and \( \mathcal{A} \in \mathcal{R} \) are found in Theorem 2, then \( V: \mathcal{A} \to \mathcal{R} \) is a CLF for system (7).

**Proof:** The claim is verified since Theorem 1 establishes that (11a) holds on \( \mathcal{A} \) and for each \( x \in \mathcal{A}^1 \), \( u(x) \in \mathcal{U} \) and \( D^t \mathcal{V}(x) + b_2 V(x) \leq 0 \), satisfying (11b).

**Remark 2:** If \( b_2 \leq 0 \) is found in Theorem 2, but \( \exists \mathcal{T} \in \mathcal{R}^n \) and \( b_2 > 0 \) defined in Remark 1, one can check if \( \mathcal{A} = V^{-1}(0, r) \subseteq \mathcal{T} \) is in \( \mathcal{R}^n \) for some \( r > 0 \). If so, \( \|x(t)\|/r \)’s upperbound in Theorem 2 holds with \( b_2 := b_2 \) for all \( x(t_0) \in \mathcal{A}^1 \).

**C. Safety**

This section modifies Instantiation 1 to search for control-invariant set of feasible states that can reach a target set, \( A_1 \subseteq \mathcal{R}^n \). The following definitions are needed.

**Definition 7 (Controllable/Stabilizable Sets, [28, Ch. 10]):** A set \( \mathcal{A} \) is controllable to target set \( A_1 \subseteq \mathcal{A} \) if \( \mathcal{A} \) is a control-invariant set where each state in it can be driven to \( A_1 \). If \( A_1 \) is control-invariant, the set \( \mathcal{A} \) is called stabilizable.

Note that \( \mathcal{A}^1 \) in Lemma 2 and Theorem 2 is a stabilizable set for \( A_1 = \{0\} \). For larger \( A_1 \), a controllable/stabilizable set will be found using a modified definition of barrier functions.

**Definition 8:** Consider system (1), where \( g: \Omega \to \mathcal{R}^n \) is a Lipschitz map. Let a Lipschitz function \( V: \Omega \to \mathcal{R} \), where \( \Omega \subseteq \mathcal{X} \) is in \( \mathcal{R}^n \), and the sets \( \mathcal{A}, A_1 \subseteq \mathcal{R}^n \), where \( A \supseteq A_1 \) is a
sublevel set of $V$, and $a, b_1, b_2 > 0$ satisfy
\begin{align}
 b_1 \| x \|^a & \leq V(x) \leq V_{\partial A} \quad \forall x \in A \quad (12a) \\
 D^+ V(x) & \leq -b_2 V(x) \quad \forall x \in (A \setminus A_1)^c \quad (12b)
\end{align}
where $V_{\partial A}$ is the constant value of $V(x)$ on $\partial A$. Then, the restriction of $V(\cdot)$ to $A^c$, that is $V : A^c \to \mathbb{R}$, is a barrier function for (1).

Definition 8 modifies the zeroing barrier function definition in [16] by requiring a positive value for $V$ on the boundary of the set, and allowing $V$ to have positive time derivatives inside $A_1$, which means the decrease condition is not required everywhere in $A$. In fact, the following theorem ensures positive-invariance of $A$ and reachability of $\partial A_1$ from $(A \setminus A_1)^c$. Moreover, it provides a clear upper bound on the decay rate of the state norm in $(A \setminus A_1)^c$.

**Lemma 3:** If there exist a function $V(x)$, and sets $A$ and $A_1$ satisfying Definition 8, then $A$ is a positive-invariant set for (1). Further, if $x(t_0)\in (A \setminus A_1)^c$, the state reaches $\partial A_1$ and $\|x(t)\| \leq \sqrt{V(x(t_0))} b_1 e^{-\frac{b_2}{2}(t-t_0)}$ holds as long as $x(t)$ remains in $(A \setminus A_1)^c$.

**Proof:** Positive-invariance of $A$ and reachability of $\partial A_1$ in case $x(t_0) \in (A \setminus A_1)^c$ follows as a special case of [31, Th. 2.6] because $b_2 V(x)$ is a positive number and $A, \bar{A}_1 \in \mathbb{R}^n$. The convergence bound on $\|x(t)\|$ then follows from (12a) and (12b) by using the comparison lemma, [30, Lemma 3.4].

Given a target set $A_1 \in \mathbb{R}^n$, Lemma 3 gives sufficient conditions for finding $A^c$, a controllable/stabilizable set for it. The following theorem modifies Instantiation 1 to synthesize a controller that finds a controllable/stabilizable set for a target set. It formulates a feasibility problem to search for a controller and a function $V(\cdot)$ that has a constant, but unknown, value on $\partial A$ and decreases on $A \setminus A_1$, making $V(\cdot)$ a barrier function.

**Theorem 3 (Reaching a target):** Given $A, A_1 \in \mathbb{R}^n$, where $A_1 \subset A$, let $\Omega := A$ in the Instantiation 1, and suppose that $\partial A_1 \subset \cup_{\sigma \in \partial E} \partial \sigma$. Let $I_1 = \{ i \in Z_{P}^{m_r} \mid \sigma_i \in A_1 \}$ and $I_0 = Z_{P}^{m_r} \setminus I_1$. Further, let $V_x = V_{\partial A}$ for all $x \in \mathbb{E}_{\partial T}$, and $b = \{ b_1, b_2 \} \in \mathbb{R}$. The modified inequalities remain feasible when imposing $V_x \leq V_{\partial T} \forall x \in \mathbb{E}_{\partial T}$ and imposing (8) only for $i \in I_0 \cup \partial E$. If $b_2 > 0$, then the CPA function $V : \mathbb{T} \to \mathbb{R}$ constructed from the elements of $V$ is a barrier function for the closed-loop system, making $A^c$ positive-invariant. Further, if $x(t_0) \in (A \setminus A_1)^c$, then $x(t)$ reaches $\partial A_1$, and $\|x(t)\| \leq \sqrt{V_{\partial A} b_1} b_1 e^{-\frac{b_2}{2}(t-t_0)}$ while $x(t)$ remains in $(A \setminus A_1)^c$.

**Proof:** To see that imposing $V_x \leq V_{\partial T} \forall x \in \mathbb{E}_{\partial T}$ does not harm feasibility, note that with any $a \geq 1$ and $b_1 > 0$, $V$ can be obtained from $V_x = b_1 \| x \|^{a}$ for $x \in \mathbb{E}_{\partial T}$, and $V_x = \max \{ b_1 \| x \|^a \mid x \in \mathbb{E}_{\partial T} \}$ for $x \in \mathbb{E}_{\partial T}$. Paralleling the arguments proving Theorem 1, feasible $I, \sigma$, and finite $b_2$ can be found. It remains to be shown that $V$ verifies Definition 8 and Lemma 3 if $b_2 > 0$. Letting $\Omega := A$ in Definition 8, since $V$ is CPA, $b_1 \| x \|^a \leq V_x \leq V_{\partial T} \forall x \in \mathbb{E}_{\partial T}$ implies (12a). By Corollary 1, assuming $\Omega := (A \setminus A_1)^c$, (8c)–(8f) imply (12b). The upper bound on the decay rate of the state norm and reachability of $\partial A_1$ follow from Lemma 3.

**Remark 3:** If $b_2 \leq 0$ is found in Theorem 3, a subtriangulation $\bar{T}$ and $\bar{b}_2 > 0$, both defined in Remark 1, might exist. If so, the controllable/stabilizable set is defined as any $\bar{A} = \mathbb{V}^{-1}(\{ r_1, r_2 \}) \subset \bar{T}$, where $r_1 < r_2$, satisfying $\bar{A} \in \mathbb{R}^n$ and $\bar{A} \supset A_1$. Moreover, the barrier function is the restriction of $V$ to $\bar{A}$, that is, $V : \bar{A} \to \mathbb{R}$, and $\|x(t)\|$’s upperbound in Theorem 3 holds with $b_2 := \bar{b}_2$ and $\bar{A} := \bar{A}$.

**IV. ENLARGING THE REGION OF ATTRACTION**

As established by Theorem 2 (stabilization), $A^c$, the interior of a sublevel set of the obtained Lyapunov function, is an ROA for the closed-loop system. However, larger ROAs might exist. On the other hand, the inequalities in Theorem 3 (reaching a target) make the boundary of the triangulation a positive-invariant set. This section uses this idea to enlarge the ROA when the target set is the origin. The ROA can then be used as the control-invariant target set to find a larger stabilizable set that contains it by Theorem 3 (reaching a target).

**A. Single-Stage Design**

By assuming $A_1 = \emptyset$ and $V_0 = u_0 = 0$ in Theorem 3 (reaching a target), the following theorem tries to make the triangulation’s interior a ROA.

**Theorem 4 (Single-stage ROA):** Given $A \in \mathbb{R}^n$, let $\Omega := A$ in the Instantiation 1. Assume that $0 \in U$, and $f(0) = 0$, and augment (8) with the constraints $V_0 = u_0 = 0$. Let $V_x = V_{\partial T} \forall x \in \mathbb{E}_{\partial T}$ and $V_x \leq V_{\partial T} \forall x \in \mathbb{E}_{\partial T}$. If $b_2 > 0$, then $V : \mathbb{T} \to \mathbb{R}$ is the corresponding CPA Lyapunov function for the closed-loop system, making $x = 0$ locally exponentially stable for the closed-loop system, where $\|x(t)\| \leq \mathbb{V}_{\partial T} b_1 e^{-\frac{b_2}{2}(t-t_0)}$ for all $x(t_0) \in A^c$.

**Proof:** The claims follow from Theorems 2 and 3.

If $b_2 \leq 0$ is found in Theorem 4, a smaller ROA inside the triangulation can be found following Remark 2. As will be shown in Section VIII, the proposed single-stage design might be used to automatically search for a large ROA in a set, but it usually can do so on refined triangulations. Thus, expanding the ROA via a multistage design is proposed next.

**B. Multistage Design**

When $A_1 \neq \emptyset$ in Theorem 3 (reaching a target), a large positive-invariant set can be found because its inequalities ignore the time derivative of $V(\cdot)$ inside $A_1$. This independence from what happens inside $A_1$ means that reaching $\partial A_1$ from the points in $(A \setminus A_1)^c$ can be still ensured even if (8) is solved on a hollowed triangulated region that has $\partial A_1$ as its interior boundary. If $A_1^c$ is itself a ROA found by Theorem 2 (stabilization) or Theorem 4 (single-stage ROA), finding $A \supset A_1$ provides a larger ROA. This motivates a multistage design where in each stage, a hollowed region surrounds last-stage’s positive-invariant set. In this multistage design, the CPA choice for Lyapunov-like functions and controllers makes it possible to either stitch them to their corresponding functions found previously in the inner
region, or let each/both of them be piecewise CPA functions if discontinuous functions along the boundary of the inner region are preferred.

There are two main advantages for a multistage design. Note that as the number of simplexes in a triangulation increases, so does the number of constraints in Instantiation 1, complicating the optimizations in Theorems 2 and 4. However, each stage of the multistage design may have far fewer simplexes, so the multistage design can be used to seek out progressively larger sets with the assurance that a valid ROA will be available even if the available computation time elapses mid-computation. Moreover, each stage of the multistage design can ensure its own secondary objective, for instance imposing different convergence rates at each stage.

The following theorem describes each design stage. The variables related to stage \( k \) are denoted by superscript \([k]\).

**Theorem 5 (Multistage ROA):** Let \( y^{[k]} = [V^{[k]}, u^{[k]}, a^{[k]}, b^{[k]}] \), where \( k \in \mathbb{N} \), satisfy the inequalities in Theorem 2 (stabilization) or Theorem 4 (single-stage ROA) on \( T^{[k]} \), making the origin exponentially stable in the interior of \( A^{[k]} \subseteq \mathbb{R}^n \). Let \( T A^{[k]} = T^{[k]} \cap A^{[k]} \), and \( \Omega_{\text{temp}} \supset A^{[k]} \) be in \( \mathbb{R}^n \). In the Instantiation 1, let \( \Omega := A = \Omega_{\text{temp}} \setminus A^{[k]} \), and denote its triangulation by \( T \). Denote the set of vertices on the outer and inner boundaries of \( \Omega \) by \( \mathbb{E}_{\partial \Omega} \) and \( \partial \mathcal{A}^{[k]} \), respectively. Impose \( V_x \leq V_{\partial \Omega} \forall x \in \mathbb{E}_{\partial \Omega} \), and \( V_x = V_{\partial \Omega} \) on \( \partial \mathcal{A}^{[k]} \). Moreover, augment (8) with the constraints \( V_x = V_x^{[k]} \) and \( u_x = u_x^{[k]} \forall x \in \partial \mathcal{A}^{[k]} \). Let \( [V_x^{[k+1]}, a^{[k+1]}, b^{[k+1]}] := \{ u_x \} x \in \mathbb{E}_{\partial \Omega} \cup \{ V_x \} x \in \mathcal{A}^{[k]} \), \( V_x^{[k+1]} := \{ V_x \} x \in \mathbb{E}_{\partial \Omega} \cup \{ V_x \} x \in \mathcal{A}^{[k]} \), \( a^{[k+1]} := \min \{ a^{[k]}, a \} \), and \( b^{[k+1]} := \min \{ b_1^{[k]}, b \} \), \( \min \{ b_2^{[k]}, b \} \} \). If \( b_2 > 0 \), then \( y^{[k+1]} \) satisfies Theorem 4 on \( T^{[k+1]} := TA^{[k+1]} \cup T \), making the origin locally exponentially stable in the interior of \( A^{[k+1]} := A^{[k]} \cup A \). Moreover, \[ \| x(t) \| \leq V_{\partial \Omega} x^{[k+1]} e^{-\rho_{\Omega}^{[k+1]}(t-t_0)} \] if \( x(t_0) \in (A^{[k+1]})^o \).

**Proof:** Using the definition of \( y^{[k+1]} \), the proof follows from that of Theorem 4 since \( V_{\partial \Omega} \) is a level set of the CPA Lyapunov function constructed from the elements of \( V^{[k]} \). The given upper bound on the decay rate of the state norm follows from Theorem 3.

As the sets \( A^{[k]} \) and \( T \) are put together in stage \( k + 1 \) in Theorem 5 (multistage ROA), the result may not look like a triangulation, because the vertices on the inner boundary of \( \Omega \) and \( \partial \mathcal{A}^{[k]} \) may not match. However, since these boundaries are same, and both \( V(\cdot), u(\cdot) \) are CPA functions, there is no ambiguity in determining their values on \( \partial \mathcal{A}^{[k]} \). In fact, the apparent mismatch between \( T \) and the triangulation of \( \mathcal{A}^{[k]} \) can be easily resolved by connecting appropriate vertices of the two to generate some redundant simplexes that carry no new information, as shown in Fig. 1. Also, if \( b_2 \leq 0 \) is found in Theorem 5, a smaller positive-invariant set can be sought following Remark 3.

**Remark 4:** Theorem 5 (multistage ROA) ensures that the CPA functions of consecutive stages remain continuous since they are stitched together. Relaxations of Theorem 5 can be expressed by allowing CPA functions of the Lyapunov-like functions or controllers of consecutive stages to be discontinuous along the boundary of \( A^{[k]} \)'s, making them piecewise CPA. This is easily done by eliminating either or both of the constraints \( V_x = V_x^{[k]} \) \( \forall x \in \mathbb{E}_{\partial \Omega} \) and \( u_x = u_x^{[k]} \forall x \in \mathcal{A}^{[k]} \). Since each \( A^{[k]} \) contains \( A^{[k-1]} \), having finite number of stages means finite number of switches in the Lyapunov function (or in the controller), ensuring overall stability.

**Remark 5:** Since \( A^{[k]} \) is already a ROA in Theorem 5, one can use any set \( A^{[k]} \subseteq A^{[k-1]} \), where \( A^{[k]} \) is in \( \mathbb{R}^n \), to substitute \( A^{[k]} \) and remove the corresponding constraints that equate the values of \( V(\cdot), u(\cdot) \) from stage \( k \). Since the state will eventually reach \( A^{[k]} \), the controller can be switched to \( u(x) \) constructed from \( u^{[k]} \) once the state is in \( A^{[k]} \). This can be helpful in practice to prevent the triangulation in stage \( k + 1 \) to have an unnecessarily large number of fine simplexes around the boundary of the inner set. An example is provided in Section VIII.

For the first stage in the proposed multistage design, three options are as follows. One, using Theorem 2 (stabilization). Second, using Theorem 4 (single-stage ROA). Third, if the state transition matrix of the linearized system around the origin is Hurwitz, one can design a linear controller for it. Then, using [20, Th. 5], find a sublevel set \( A \) for the closed-loop system’s Lyapunov function, making sure that the state and input constraints are satisfied in it.

**V. ITERATIVE IMPROVEMENT ALGORITHM**

As discussed in Theorems 2–5, formulating Lyapunov-like functions depends on finding \( b_2 > 0 \), providing a natural value to optimize when solving their constraints. Choosing a cost function that weighs increasing \( b_2 \) against other objectives is a bad choice because no useful controller is formulated unless \( b_2 > 0 \). Once it is found, it can be kept fixed during further optimizations to achieve other performance objectives. However, the constraints in Theorems 2–5 have nonlinear terms that increase in number alongside the simplexes. This section gives an iterative algorithm that implements the discussed strategy using iterative, well-posed SDPs, convexifying the optimizations with conservatism.

Since the constraints in Theorems 2–5 were formulated using Instantiation 1, this section gives an algorithm that iteratively...
searches for $b_2 > 0$ using a sequence of SDPs. If $b_2 > 0$ is found, the algorithm fixes it, and then optimizes other performance objectives. The following theorem formulates each iteration. The required modifications to formulize similar optimizations for Theorems 2–5 are inferred from their corresponding changes to Instantiation 1, but their initializations will be discussed separately. The theorem establishes that feasibility is maintained and the cost decreases at each iteration. Convergence is, therefore, assured if the cost chosen is bounded below. There is no assurance that $b_2 > 0$ can be obtained, but the authors in [20] and [24] provided conditions ensuring that a CPA Lyapunov function exists for a given controller and vice versa. When applicable, these could be used to initialize (13) and ensure $b_2 > 0$ is found. However, our aim is to provide an algorithm that can be used when stabilizability is not assured, for instance, identifying a stabilizable region when subsets of the feasible set are not stabilizable.

Theorem 6 (Iterative Optimization): Let $y = [V, 1, u, z, a, b]$, satisfy (8), and $J(\cdot)$ be a cost function. Consider

$$
\delta y^* = \text{argmin}_{\delta y = [\delta V, \delta l, \delta u, \delta z, 0, \delta b]} J(y + \delta y)
$$

s.t.

$$
\begin{align*}
& b_1 + \delta b_1 > 0 \quad (13a) \\
& (b_1 + \delta b_1) ||x||^a \leq V_x + \delta V_x \quad \forall x \in \mathbb{E}_T \quad (13b) \\
& |\nabla V_i + \delta \nabla V_i| \leq l_i + \delta l_i \quad \forall i \in Z_{1}^{mr} \quad (13c) \\
& H(u_x + \delta u_x) \leq h_c \quad \forall x \in \mathbb{E}_T \quad (13d) \\
& |\nabla u_i(s) + \delta \nabla u_i(s)| \leq z_i + \delta z_i \quad \forall i \in Z_1^{mr} \quad \forall s \in Z_0^n \quad (13e) \\
& P_{i,j} \leq 0 \quad \forall i \in Z_1^{mr} \quad \forall j \in Z_0^n \quad (13f)
\end{align*}
$$

where $\delta \nabla V_i = X^{-1}_i \delta \hat{V}_i$, $\delta \nabla u_i(s) = X^{-1}_i \delta \hat{u}_i(s)$ as in Definition 5.

$$
P_{i,j} = \begin{bmatrix}
\hat{\phi}_{i,j} & * & * & * & * & * \\
\delta \nabla V_j & -2I_n & * & * & * & * \\
G(x_{i,j}) \delta u_{x_{i,j}} & 0 & -2I_n & * & * & * \\
\delta V_{x_{i,j}} & 0 & 0 & -2 & * & * \\
\delta b_2 & 0 & 0 & 0 & -2 & * \\
\sqrt{\eta_i c_{i,j}} \delta z_i & 0 & 0 & 0 & 0 & -2 \\
\end{bmatrix}
$$

(14)

and $c_{i,j}$ is given in (6), and $\mu_i, \eta_i$ are given in (9). Then, $y + \delta y^*$ is a feasible point for (8), and $J(y + \delta y^*) \leq J(y)$. □

Proof: To see that (13) is feasible, observe that $\delta y = 0$ satisfies (13) since in this case, (13) is equivalent to (8). In fact, (13f) is the convexified version of (8f). To see this, recall that $w^T w \leq 0.5(w^T w + v^T v)$ for any vectors $v, w$ with the same dimension. Applying this fact with $(v, w) = (\delta \nabla V_i, G(x_{i,j}) \delta u_{x_{i,j}})$, $(v, w) = (\delta z_i, 1^T \delta l_i)$, and $(v, w) = (\delta V_{x_{i,j}}, \delta b_2)$ shows that by the Schur complement, (13f) is implied. Finally, $J(y + \delta y^*) \leq J(y)$ because otherwise $\delta y = 0$ would be a better, feasible solution. □

Remark 6: In two cases, the last two rows and columns of (14) are eliminated. First, when writing (14) in each simplex for $j = 0$ since $c_{i,j} = 0$ in (6). Second, when $G(x)$ in (7) is a constant matrix because $\eta_i = 0$ in (9). In the second case, (8e) and (13e) are not needed as well. □

Remark 7: If no particular $b_2 > 0$ is sought, one may modify (8f) to $D_{i,j}^+ V \leq 0$. This eliminates the fourth and fifth block-rows and -columns of (14), reducing computation effort. Note, however, that if used for Theorem 2 or 4, (8f) for $x = 0$ should be discarded because $D_{i,j}^+ V$ is zero at $x = 0$ in those theorems. □

Starting with a feasible point of (8), Theorem 6 (iterative optimization) can be used repeatedly to potentially decrease the values of the cost function. Note that by letting $b_1 + \delta b_1$ be greater than or equal to a small positive number, and a linear/quadratic cost function, (13) is a SDP in the standard format. The small positive number must be kept constant in the later iterations. Two feasible initialization points for (8) are given next. These provide only an initial feasible point for the optimization to begin the iterative algorithm. As will be shown in the Section VIII, they are often poor solutions with a large negative $b_2$. It is the iterative process that turns a poor initial candidate for the controller and Lyapunov function pair to a stabilizing one.

Initialization 1: Choose $a \geq 1$ and $b_1 > 0$. Compute $V_x = b_1 ||x||^a \forall x \in \mathbb{E}_T$ and then, $l_i = |\nabla V_i| |\in Z_1^{mr}$ using Definition 5. Find $u = \{u_x\}_{x \in \mathbb{E}_T}$ and $l = \{l_i\}_{i \in Z_1^{mr}}$ using the following LP:

$$
\begin{align*}
& \min_{b_1, u, l} \quad -b_2 \\
& \text{s.t.} \quad H u_x \leq h_c \quad \forall x \in \mathbb{E}_T \quad (16a) \\
& & |\nabla u_i(s)| \leq z_i \quad \forall i \in Z_1^{mr} \quad \forall s \in Z_0^n \quad (16b) \\
& & D_{i,j}^+ V \leq -b_2 V_{x_{i,j}} \quad \forall i \in Z_1^{mr} \quad \forall j \in Z_0^n \quad (16c)
\end{align*}
$$

where $H, h_c$, and $D_{i,j}^+ V$ are given in Instantiation 1. □

Initialization 2: Linearize (7) around the origin. Design a linear quadratic regulator (LQR) controller, and find the corresponding Lyapunov function, $x^T P x$. Sample $x^T P x$ at the vertices of $T$ to find $V$, and let $a = 2$ and $b_1$ be equal to the smallest eigenvalue of $P$. Sample the LQR controller at the vertices of $T$ to form $u^\text{LQR} = \{u_x^\text{LQR}\}_{x \in \mathbb{E}_T}$. Divide each element of $u^\text{LQR}$ by an appropriate positive number so that the result, $u = \{u_x\}_{x \in \mathbb{E}_T}$, has admissible values for all vertices. Compute $l_i = |\nabla V_i|$ and $z_i = |\nabla u_i(s)|$ for all $i \in Z_1^{mr}$ using the values of $V_x$ and $u_x$, respectively. Finally, find the largest $b_2$, satisfying (8f) in all simplices. □

Given a triangulation and a linear/quadratic cost function $J(V, 1, u, z, b_1)$, the procedure for finding a positive $b_2$ and minimizing $J(\cdot)$ using Instantiation 1 is given in Algorithm 1
Algorithm 1: Iterative Control Design on a Fixed Triangulation.

**Inputs:** Instantiation 1, and a linear/quadratic $\hat{J}(\hat{y})$, where $\hat{y} = [V, 1, \mathbf{u}, x, b_1]$

**Outputs:** Sufficiently large $b_2 > 0$ and $\hat{y}$, or warning

1. $y := \text{a feasible point of (8)}$ (using Initialization 1 or 2)
2. $J := -b_2$ ⨿ since $b_2$ is to be maximized
3. repeat
4. Use Theorem 6 (Iterative Opt.) to update $y$
5. until $b_2 > 0$ is large enough OR $b_2$ is not changing
6. if $b_2 > 0$ is found then
7. Fix $b_2$, and let $J := J(\hat{y})$
8. repeat
9. Use Theorem 6 (Iterative Opt.) to update $\hat{y}$
10. until $J$ is sufficiently small OR $J$ is not changing
11. Return $b_2$ and $\hat{y}$
12. else Issue warning
13. end if

via iterative pSDP. It increases $b_2$ until it is positive. Since $e^{-\frac{2\pi^2}{\Delta_t^2}}$ is proportional to the state norm’s upper bound when $\alpha$ is fixed, increasing $b_2 > 0$ can continue until a desired decay rate is ensured. Then, by fixing $b_2$’s value, $\hat{J}(\cdot)$ is iteratively minimized. Note that $\hat{J}(\cdot)$ is not a function of $b_2$ and $\alpha$. Thus, by keeping them fixed while improving the performance, the stability and established convergence rate for the state norm are preserved. Both of the loops can be terminated in lines 5 and 10 if a predefined maximum number of iterations is reached. If a sufficiently large positive $b_2$ satisfying $||x||$’s desired decay rate requirement cannot be found, triangulation refinement, discussed later, is needed.

Once Algorithm 1 is terminated at line 5 or 10, if no other improvement is needed, the returned $\hat{y}$ can serve as an initial guess for a nonconvex optimization, formulated by a nonlinear cost function $J(\hat{y})$ and (8) as constraints, as a final attempt to boost the performance.

By appropriate modifications that account for differences in the constraints of Instantiation 1 and those of Theorems 2–5, iterative optimizations for Theorem 2 (stabilization), Theorem 3 (reaching a target), Theorem 4 (single-stage ROA), and Theorem 5 (multistage ROA) are obtained. Then, provided that they are correctly initialized, Algorithm 1 can be used to find a large enough $b_2 > 0$ and achieve further objectives for any of them. At line 4, $\mathcal{T}$ and $b_2 > 0$ can be sought following Remark 1 whenever $b_2 \leq 0$ is found. Once desired objectives are met, the positive-invariant set $\mathcal{A}$ can be also found. Modifying Initializations 1 and 2, the next section gives feasible points for Theorems 2–5 to be used in line 1 of Algorithm 1 when designing by any of them.

A. Specialized Initializations

Here, for each of Theorems 2–5, two feasible points are given, allowing their iterative versions to be used in Algorithm 1 in lines 4 and 9 once properly initialized in line 1.

1) For Theorem 2 (Stabilization): Initialization 3: Use Initialization 1, and let $u_0 = 0$.

2) Initialization 4: Same as Initialization 2.

3) For Theorem 3 (Reaching a Target): Initialization 5:

Let $V_x = b_1 ||x||^\alpha \forall x \in \mathcal{E}_T \mathcal{\setminus} \partial \mathcal{E}_T$, and $V_{\partial \mathcal{E}_T}^x = \max_{x \in \mathcal{E}_T} b_1 ||x||^\alpha$ in Initialization 1.

Initialization 6: Use Initialization 2 to find $V = \{V_x \}_{x \in \mathcal{E}_T}$. Replace the elements of $V$ corresponding to boundary vertices with $V_{\partial \mathcal{E}_T}^x = \max_{x \in \mathcal{E}_T} V_x$. The rest is identical to Initialization 2.

4) For Theorem 4 (Single-Stage ROA): Initialization 7:

Use Initialization 5, and let $u_0 = 0$.

Initialization 8: Same as Initialization 6.

5) For Theorem 5 (Multistage ROA): Initialization 9:

Let $\alpha := a[k]$, and $b_1 := \min_{x \in \mathcal{E}_T \setminus \partial \mathcal{E}_T} \mathcal{A}[k] V_{\partial \mathcal{A}[k]}^x ||x||^\alpha$. Assign $V_{\partial \mathcal{A}[k]}^x$ to all elements of $\{V_x \}_{x \in \mathcal{E}_T \setminus \partial \mathcal{E}_T}$, and let $V_x = b_1 ||x||^\alpha$ for all $x \in \mathcal{E}_T \setminus \partial \mathcal{E}_T$ to obtain $V_{\text{temp}}^x = \{V_x \}_{x \in \mathcal{E}_T}$. Now replace all elements in $V_{\text{temp}}$ corresponding to vertices in $\mathcal{E}_T \setminus \partial \mathcal{E}_T$ with $V_{\partial \mathcal{E}_T}^x := \max_{x \in \mathcal{E}_T} V_x$. The result is $\{V_x \}_{x \in \mathcal{E}_T}$. The rest is identical to Initialization 1.

Initialization 10: Linearize (7) around the origin. Design a LQR controller, and find the corresponding quadratic Lyapunov function, $x^T P x$. Sample $x^T P x$ at the vertices of $\mathcal{T}$ to find $V_{\text{temp}}$, and let $\alpha = 2$ and $b_1$ be equal to the smallest eigenvalue of $P$. Replace all elements of $V_{\text{temp}}$ corresponding to vertices in $\partial \mathcal{E}_T$ and $\mathcal{A}[k]$ by $V_{\partial \mathcal{E}_T} = \max_{x \in \mathcal{E}_T} V_x$ and $V_{\mathcal{A}[k]} = \min_{x \in \mathcal{E}_T} V_x$, respectively, to obtain $V = \{V_x \}_{x \in \mathcal{E}_T}$. The rest is identical to Initialization 2 except for finding $u$, where its elements that correspond to vertices in $\partial \mathcal{A}[k]$ should be replaced by corresponding values from the CPA controller $u[k](\cdot)$.

While the proposed iterative pSDP scheme mitigates some of the computational challenges of nonlinear optimization, when it converges, the algorithm is expected to at best arrive at a local optimum, and the one it arrives at will depend on the initialization.

VI. Minimum-Norm Controllers

Using Algorithm 1, a stabilizing controller that ensures a desired upper bound for the decay rate of the state norm can be found at line 5 using iterative versions of any of Theorems 2, 4, or 5. Although such a controller is guaranteed to be optimal at all times when the system is initialized in the corresponding positive-invariant set, its norm can further be reduced without losing stability or safety. This section suggests two notions of achieving the minimum-norm for the controller. The first one continues the offline design in Algorithm 1 by minimizing a quadratic cost function. The second one takes the CLF obtained at line 5 or 10 of Algorithm 1, and seeks a point-wise minimum-norm controller using online quadratic programming (QP). Having minimum-norm property at all points and online computations poses a tradeoff between the two proposed minimum-norm controllers. Moreover, while Lipschitz property for the offline designed controller is guaranteed, it is not the case for the QP-based controller [16].
A. Continuing Offline Design

By Definition 5, the CPA controller is fully defined by its value at the vertices of a triangulation. After a stabilizing controller results from $b_2 > 0$ in any of Theorems 2–5, finding a minimum-norm controller can be a secondary objective of further offline optimizations. This can be done by fixing the obtained $a$ and $b_2$, and minimizing the quadratic cost function, $J = u^T u$. Since the $u$ contains the values of the CPA $u(\cdot)$ at all vertices, minimizing $u^T u$ results a controller that has minimum-norm property at all the vertices. Since $u(\cdot)$ is CPA, at each point in the interior of any simplex, $u(x)$ will be a linear interpolation of minimum-norm values at the vertices of that simplex. Minimizing $u^T u$ can be done as a nonconvex optimization or by iterative SDPs as follows, making it a candidate for lines 6–13 of Algorithm 1.

Let $y = [V, 1, u, z, a, b]$ be a feasible point of (8), where $b_2 > 0$ is found by any of Theorems 2–5. Let $\hat{y} = [V, 1, u, z, b_1, \phi]$, where $\phi \in \mathbb{R}$, be the unknowns on the same triangulation in which $y$ was found. Fixing $b_2$ and $a$, the search for the controller that is minimum-norm at all vertices can be formulated as the following SDP:

$$\begin{align*}
\min_y \phi & \quad \text{s.t. } \phi \geq 0 \\
& \quad \begin{bmatrix}
-\phi & * \\
0 & -1
\end{bmatrix} \preceq 0 \\
c(\hat{y}) & \leq 0
\end{align*}$$

where (17c) encapsulates all the constraints in Instanation 1 together with any modifications or added constraints suggested by any of Theorems 3, 4, or 5. Note that (17b) is the Schur complement of $u^T u - \phi \leq 0$. Since $y$ is a feasible point of (17), the convex-overbounding technique used in Theorem 6 can be applied to solve (17) iteratively. This gives an example for continuing Algorithm 1 after line 5.

B. QP-Based Online Application

As discussed, the restriction of the obtained Lyapunov-like functions to their respective positive-invariant sets in Theorems 2–5 are either CLFs or control barrier functions (CBFs). Thus, online QP can be implemented to find a point-wise minimum-norm controller as the state trajectory evolves similar to what is proposed in [16].

Suppose that $b_2 > 0$ is found by Algorithm 1, and $V : \mathcal{A} \to \mathbb{R}$ is the restriction of the corresponding CPA Lyapunov function to one of its sublevel sets inside the triangulation as specified in any of Theorems 2–5, that is $\mathcal{A} = V^{-1}(0, r)$, $r > 0$, where $\mathcal{A} \subseteq \mathcal{T}$ and $\mathcal{A} \subseteq \mathbb{R}^n$. Starting at any $x \in \mathcal{A}$, the minimum-norm online QP controller can be written as

$$\begin{align*}
\hat{u}^* &= \arg\min_u \quad u^T \hat{H}(x) u + \hat{h}(x)^T u \\
& \quad \text{s.t. } H u \leq h_c \\
& \quad \nabla V^T f(x) + G(x) u + b_3 V(x) \leq 0 \quad \forall i \in \mathcal{I}
\end{align*}$$

Algorithm 2: Control Design with Triangulation Refinement

**Inputs:** System (7), a triangulable set $\Omega \subseteq \mathcal{X}$, cost function, simplex size function $\rho : \Omega \to \mathbb{R}_{>0}$, where $\Omega \subseteq \Omega$, minimum simplex size $\rho_{\min}$, triangulation $\mathcal{T} = \{\sigma_k\}_{i=1}^m$.

**Outputs:** $y = [V, 1, u, z, a, b]$

1. Generate $\mathcal{T}' = \{\sigma_k\}_{i=1}^m$, triangulating $\Omega$ and respecting $\rho$ with $\cup_{i=1}^m \partial \sigma_k \subseteq \cup_{i=1}^m \partial \sigma_k$
2. repeat
3. Solve (8) with the cost function or use Algorithm 1
4. if desired objectives are met then
5. Return $y$
6. end if
7. Refine $\mathcal{T}$ and update $y$ using Remark 8
8. until $\rho_{\min}$ is reached

where $\mathcal{I} = \{i \in \mathbb{Z}_{1}^{n_{\mathcal{T}}} \mid x \in \sigma_i\}$ and $\hat{H}(x)$ is positive definite. The set $\mathcal{I}$ has more than one element if $x$ is on the common face of some simplexes. The optimization (18) is feasible for all $x \in \mathcal{A}$, because the corresponding CPA controller of $V$ is a feasible point for it. Therefore, the convergence inequality $\|x(t)\| \leq (r_{\frac{1}{2}})^{\frac{1}{2}}e^{-\frac{\beta}{2}(t-t_0)}$ that holds for the CPA controller, also holds for the QP-based controller.

VII. TRIANGULATION REFINEMENT

Theorems 2–5 and their iterative implementation via Algorithm 1 work on given, fixed triangulations. If $b_2 > 0$ is not found, the triangulation can be refined, introducing more simplexes. In (6), $c_{i,j}$, roughly representing the length of an edge in each simplex squared, is multiplied by $\beta_l$, which compensates for higher order terms in the Taylor’s theorem. Thus, reducing the length of edges tightens upper bounds on $D^1 V(\cdot)$. Refinement also increases design freedom in $V(\cdot)$ and $u(\cdot)$. Refinement can be done globally, or locally, targeting simplexes where $b_2 < 0$ is most negative or performance metrics are worst. The refined triangulation is guaranteed to have a feasible initialization through Initialization 1 or 2. However, warm starting with values from the previous triangulation as discussed next often reduces computation time.

Remark 8: Let $y = [V, 1, u, z, a, b]$ be a feasible point of Instanation 1 with associated CPA function $V(x)$ and controller $u(x)$. Retriangulate $\Omega$ with $\mathcal{T}' = \{\sigma'_k\}_{k=1}^m$, where each simplex of $\mathcal{T}'$ is a subset of a simplex in the original triangulation, $\mathcal{T}$. On each vertex, $x'$, of $\mathcal{T}'$, set $V_{x'} = V(x')$ and $u' = u(x')$. With this, on $\sigma_k$, $l_k$ and $z_k$ can be found from Definition 5 and $\beta_k$ and $\epsilon_{kj}$ from 6 for $j \in \mathbb{Z}_0^r$, $a = a$ and $b_l = b_l$. Finally, compute $b_2' \leq \min\{-(D^1 V_{x'})^T V_{x',j} \mid j \in \mathbb{Z}_0^r, i, j \neq 0\}$. □

Algorithm 2 describes how to use Remark 8. Similar warm starts can be found for Theorems 2–5 considering their additional constraints to those of Instanation 1.
VIII. NUMERICAL EXAMPLES

This section designs controllers for a constrained nonlinear system. The single- and two-stage designs proposed here were compared with dynamic programming (DP). Like the methods proposed here, DP can avoid difficult a priori design steps using an equilibrium point as a terminal set. The goal compare the computational costs of the nonconvex optimizations of DP to the repeated SDPs proposed here, and compare the ROAs and performance metric obtained by each method.

While Baier and Hafstein [25] seek a stabilizing controller and Lyapunov function, a comparison was not performed because its results depend heavily on how an autonomous set map is formulated a priori to discretize the input space.

While Steenjes et al. [24] can also be used to seek out stabilizing controllers, our goal is to consider examples were the difficulty of choosing an initial, feasible Lyapunov function precludes implementing [24]. In fact, once \( b_2 > 0 \) is found, the output of Algorithm 1, Line 5 could initialize [24] to save computational time compared with lines 6–10. Likewise, once [24] terminates, Algorithms 1 and 2 could be used to improve performance or increase the ROA. Hence, a comparison was not made to [24] because it and Algorithms 1 and 2 should be used in complement.

Calculations were conducted in MATLAB 2019b on a desktop computer with an AMD Ryzen 5 2600 six-core CPU, and 8 GB DDR4 RAM, running the 64-bit version of Windows 10.

To aid visualization, all examples were conducted on a 2-D, lumped-mass pendulum connected to a fixed revolute joint with a mass-less rod. Its unstable, unforced equilibrium is \( (x, u) = (0, 0) \). The model and constraints with international system (SI) units were

\[
\dot{x} = g(x, u) = [x^{(2)} \sin x^{(1)} - 0.3x^{(2)} + u]^T \quad (19a)
\]

\[
\|x\|_\infty \leq 1, \ |u| \leq 5. \quad (19b)
\]

A. Controllers

The design methods for (19) are as follows.

1) Dynamic Programming: DP solving a finite-horizon optimal control problem is probably the closest design method to the proposed methods in terms of its generality, independence from a priori design choices, constraint enforcement, low online computational costs, and computing control-invariant sets that provide ROAs [28, Ch. 10]. However, in practice, gridding the state space and discretizing the equations of motion reduces accuracy, and therefore, makes the performance and the ROAs approximate and potentially invalid. Moreover, finding a suitable sampling-time and grid size to balance computation time and accuracy are nontrivial.

Assuming an \( N \) step horizon, and \( \mathcal{X}^{N-N}_N \) a control-invariant terminal set, \( \mathcal{X}^{N-N}_k \) denotes the set of feasible states that can be driven to \( \mathcal{X}^{N-N}_N \) in \( k \) steps using a sequence of \( k \) admissible inputs [28, Ch. 10]. Having \( \{\mathcal{X}^{N-N}_k\}_{k=0}^N \), the following optimization is solved for \( k \in \mathbb{Z}_+^N \) backwards \( \forall x \in \mathcal{X}^{N-N}_k \) to determine the time-varying DP controller

\[
J^*_k(x) = \min_{u(x)} x^T x + J^*_{k+1-N}(x^+) \quad (20a)
\]

s.t. \( x^+ = x + T_s g(x, u(x)) \)

\[
x^+ \in \mathcal{X}_N, u(x) \in \mathcal{U} \quad (20b)
\]

\[
x^+ \in \mathcal{X}^{N-N}_{k+1-N} \quad (20c)
\]

where \( J^*_N(x) = x^T x, J^*_k(\cdot) \) is the optimal cost-to-go of the step \( k \), (20a) denotes the Euler discretization of (19), and \( T_s \) is the sampling time. The cost function in (20) encourages fast convergence by not penalizing \( u(x) \), making it competitive to CPA designs of this article.

To approximate \( \{\mathcal{X}^{N-N}_N\}_{k=0}^{N-1} \) and \( \{J^*_k(\cdot)\}_{k=0}^{N-1} \) a uniform grid, \( \delta x(1) = \delta x(2) = 0.02 \), was chosen. To avoid a priori design choices, the origin was selected as the terminal set in (20). However, due to inevitable inaccuracies in finding \( k \)-step stabilizable sets, \( \|x\|_\infty \leq 0.01 \) was selected as \( \mathcal{X}^{N-N}_N \) instead of \( \mathcal{X}^{N-N}_N = 0 \) to avoid numerical issues. Each element of \( \{\mathcal{X}^{N-N}_k\}_{k=0}^{N-1} \) was then approximated as the convex hull of all the grid points \( x \) at which (20) was feasible. This involved solving a feasibility problem with linear constraints for each \( k \in \mathbb{Z}_+^{N-1} \) at each grid point. This was done with SeDuMi [32]. To find \( \{J^*_k(\cdot)\}_{k=0}^{N-1} \), (20) was solved backwards using the nonlinear optimization solver “fmincon” at the grid points. Inside the stabilizable sets the “spline” function interpolated the optimal cost-to-go.

The parameters \( T_s = 0.1 \) s and \( N = 40 \) were chosen so that any \( x \in \mathcal{X}_0^{N=40} \) was expected to reach \( \mathcal{X}^{N-N}_N \) in 4 s while satisfying state and input constraints, but inaccuracies caused slight violations. These choices balanced accuracy and computation time: coarser grid sizes made more constraint violations, while finer ones increased computation time. For instance, \( \mathcal{X}^{N-N}_k \) should always hold, but was violated at \( k \in \{28, 12, 4, 0\} \).

2) CPA Controllers: Two CPA controllers were designed. The software package Mesh2d [33], [34] was used to generate initial triangulations, in which the maximum edge size function \( \rho : \mathbb{R}_+^n \to \mathbb{R}_+ \) was used to produce nonuniform triangulations. All SDPs were solved by SeDuMi. The refinements were implemented following Algorithm 2, dividing each simplex in four using [35]. For SDP initializations, LQR with the cost function \( 2x^T x + u^2 \) was used.

2.1) Two-stage design For the first stage, Theorem 2 was implemented. Simplexes had a fan-like shape around the origin, as in Fig. 2(a). The polytope, \( \mathcal{X}_N \) was chosen to crudely resemble a level set of the quadratic Lyapunov function associated with an LQR design with \( x^T x + 2u \) as the cost function for the linearized, unconstrained system. To generate the initial triangulation, \( \rho = 0.25 \), and 0.08, and 0.2 were used for the boundary, and \( \|x\| \leq 0.1 \), and elsewhere in the polytope, respectively. After eight SDP iterations, the triangulation was refined. Another eight iterations on the refined triangulation resulted in \( b_2 = 0.27 \). Fig. 3 shows the progress of \( b_2 \). The refined triangulation, and
LAVAEI AND BRIDGEMAN: SYSTEMATIC, LYAPUNOV-BASED, SAFE AND STABILIZING CONTROLLER SYNTHESIS 3021

Fig. 2. Triangulations and CPA Lyapunov-like functions in the two-stage design. The triangulation and (19)'s symmetry were not preserved by the level set of the Lyapunov-like function due to numerical errors. The thick gray lines in (a) and (b) were boundaries imposed on the triangulation, and the blue lines represent the level sets of the Lyapunov-like functions. Combined, the discontinuous CPA Lyapunov function and its level set are depicted in (c) and (d). The simplexes marked by asterisks have $D_{ij}^+ V \geq 0$ in one or more vertices. (a) First stage. (b) Second stage. (c) Combined (Top view). (d) Combined (3D view).

Fig. 3. Sequence of $b_2$'s in the multistage design's first stage. After eight iterations, a refinement causes the increase between the dashed lines. At iteration 11, $b_2 > 0$ is found and stability is achieved. Further iterations hasten convergence.

Fig. 4. Single-stage design compared with 40-step DP. In (a), thick gray lines represent the boundaries of the triangulation, and asterisks mark the simplexes with $D_{ij}^+ V \geq 0$ in one or more vertices. In (b), light and dark gray areas are $X$ and $X_{0} \rightarrow 40$ in DP, respectively. The blue curves represent a level set of the CPA Lyapunov function. This ROA is 79.3% of 40-step DP's ROA, but it was computed in 3.1% of the time. (a) Single-stage design. (b) Comparing the ROA.

2.2) Single-stage design: Here, Theorem 4, and a similar argument to Remarks 27 were used. The set $X$ in (19) was triangulated to maximize the ROA found. The simplex boundaries were forced to include $\partial_X$ and the two surfaces passing through the origin, as shown in Fig. 4(a). For simplex sizes, $\rho(\cdot) = 0.02$ was used when $\|x\| \leq 0.1$ or $0.9 \leq \|x\| \leq 1$, while elsewhere $\rho(\cdot) = 0.04$. After five iterations, the level set shown in Fig. 4 was found.

B. Comparison

Qualitatively, since the time-varying DP controller is obtained by solving (20) only on grid points, it has a suboptimal performance. Moreover, the stabilizable set boundaries are approximate. This can lead to slight violations of the state or input constraints. On the other hand, the CPA controllers are conservative since they bound the maximum element of the closed-loop system’s Hessian above, and they are obtained by convex overbounding. However, once a solution is found, the ROA’s boundary is exact and no constraint violations are possible for trajectories starting in it. The DP and CPA controllers were compared based on the offline synthesis time and the closed-loop settling time to $\|x\| \leq 0.1$, starting at identical points. The DP was implemented by zero-order hold.

1) Synthesis Time: For CPA controllers, the synthesis time is the time required to solve all the SDP iterations and generate triangulations. The synthesis time for DP equals the required time for computing stabilizable sets plus solving optimizations. This omits the tedious trial-and-error process on both finer and coarser grids, giving an advantage to DP.

With both techniques, it is possible to increase the resulting ROA at the cost of greater computation times. For DP, this is the corresponding level set of the CPA Lyapunov function are also shown in Fig. 2(a).

The second stage was designed using Theorem 5 and Remark 4, where both the Lyapunov function and controller were allowed to be discontinuous along the level set of the first stage. The set boundaries in the second stage are shown in Fig. 2(b) with thick gray lines. The inner one bounds a polytope inside the level set found in the first stage following Remark 5. The outer one crudely resembles a slanted elliptic level set of the LQR initialization. For the initial triangulation $\rho = 0.04$ and 0.2 were used for the outer boundary and elsewhere, respectively. We allowed two iterations on it and then another two iteration on its refinement. The refined triangulation, and the level set found at the last iteration are shown in Fig. 2(b). This level set was found by Remark 3 since a few simplexes (red asterisks) had to be excluded to find $b_2 > 0$. The combined triangulation from the two stages and the resulting discontinuous CPA Lyapunov function are shown in Fig. 2(c) and (d), respectively.
TABLE I
SYNTHESIS TIME AND AREA OF ROA COMPARISON

| Controller          | Synthesis time (min) | ROA area |
|---------------------|----------------------|-----------|
| Two-stage CPA       |                      |           |
| (first stage)       | 2.05                 | 0.30      |
| DP (8 steps)        | 54.90                | 0.38      |
| (second stage)      | 3.39                 | 0.62**    |
| DP (the next 11 steps) | 111.02             | 0.52**    |
| Two-stage CPA (Combined) | 5.44                | 0.87      |
| DP (19 steps)       | 165.92               | 0.90      |
| Single-stage CPA    | 10.93                | 0.73      |
| DP (40 steps)       | 351.69               | 0.92      |

* area of $A_A$

** area of 19-step stabilizable set minus 8-step’s

Fig. 5. Multistage design and DP’s pROA. The light gray area is $X$ and the blue lines are each stage’s level sets. The dark gray areas represent 8-step and 19-step stabilizable sets in (a) and (b), respectively. Two-stage ROA is 96.7% of the 19-step DP’s ROA, but it was computed in 3.3% of the time. (a) First stage. (b) Second stage.

Fig. 6. Difference between the settling time to $\|x\| \leq 0.1$, in seconds using the DP controller and the two-stage design, initialized at identical points. The two-stage CPA design settles faster for most of the initial points in its ROA.

C. Setting Time

The settling time to $\|x\| \leq 0.1$ using the two-stage design and DP was compared by initializing them on a uniform grid with $\delta x^{(1)} = \delta x^{(2)} = 0.1$ inside the ROA of the two-stage controller. Fig. 6 shows the difference in the settling time, showing that the closed-loop system with the two-stage CPA design converges faster than DP in most regions.

IX. CONCLUSION

A systematic controller synthesis method for state- and input-constrained nonlinear systems was developed using CPA Lyapunov functions and controllers on triangulated subsets of the admissible states. The method is distinguished by its generality, complete offline design, and independence from typical unclear a priori design choices. For control-affine systems, the method was formulated as iterative SDPs that can be solved using available software. Safety and stability were guaranteed by finding CBF or CLFs, and the controller simultaneously. Therefore, it can also be viewed as a systematic approach to find CBFs and CLFs. Numerical examples showed the efficiency and effectiveness of the method.

REFERENCES

[1] R. E. Kalman and J. E. Bertram, “Control System Analysis and Design via the “Second Method” of Lyapunov: I–Continuous-Time Systems,” J. Basic Eng., vol. 82, no. 2, pp. 371–393, 1960.
[2] S. Prajna, P. A. Parrilo, and A. Rantzer, “Nonlinear control synthesis by convex optimization,” IEEE Trans. Autom. Control, vol. 49, no. 2, pp. 310–314, Feb. 2004.
[3] J. Dafouz, P. Riedinger, and C. Jung, “Stability analysis and control synthesis for switched systems: A switched Lyapunov function approach,” IEEE Trans. Autom. Control, vol. 47, no. 11, pp. 1883–1887, Nov. 2002.
[4] R. Freeman and P. V. Kokotovic, Robust Nonlinear Control Design: State-Space and Lyapunov Techniques, Berlin, Germany: Springer Sci. and Bus. Media, 2008.
[5] S. Nitzravesh, Nonlinear Systems Stability Analysis: Lyapunov-Based Approach, Boca Raton, FL, USA: CRC Press, 2018.
[6] M. Malisoff and F. Mazenc, Constructions of Strict Lyapunov Functions, New York, NY, USA: Springer Sci. Bus. Media, 2009.
[7] P. Giesl and S. Hafstein, “Review on computational methods for Lyapunov functions,” Discrete Contin. Dyn. Syst.-B, vol. 20, no. 8, 2015, Art. no. 2291.
[8] Y. Orlov, Nonsmooth Lyapunov Analysis in Finite and Infinite Dimensions, Berlin, Germany: Springer, 2020.
[9] P. Mhaskar, N. H. El-Farra, and P. D. Christofides, “Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control,” IEEE Control Syst. Lett., vol. 5, no. 8, pp. 650–659, Aug. 2020.
[10] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, “Robustness of control barrier functions for safety critical control,” IFAC-PapersOnLine, vol. 48, no. 27, pp. 54–61, 2015.
[11] Q. Nguyen and K. Sreenath, “Exponential control barrier functions for enforcing high relative-degree safety-critical constraints,” in Proc. IEEE Amer. Control Conf., 2016, pp. 322–328.
[12] L. Liu, Y. Liu, A. Chen, S. Tong, and C. P. Chen, “Integral barrier Lyapunov function-based adaptive control for switched nonlinear systems,” Sci. China Inf. Sci., vol. 63, no. 3, 2020, Art. no. 132203.
[13] A. J. Taylor and A. D. Ames, “Adaptive safety with control barrier functions,” in Proc. Amer. Control Conf., 2020, pp. 1399–1405.
[14] P. Wieland and F. Allgöwer, “Constructive safety using control Lyapunov–barrier function,” IFAC Proc. Volumes, vol. 40, no. 12, pp. 462–467, 2007.
[15] M. Z. Romdlony and B. Jayawardhana, “Stabilization with guaranteed safety using control Lyapunov–barrier function,” Automatica, vol. 66, pp. 39–47, 2016.
[16] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” IEEE Trans. Autom. Control, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.
[17] H. Chen and F. Allgöwer, “Nonlinear model predictive control schemes with guaranteed stability,” in Nonlinear Model Based Process Control, Dordrecht: The Netherlands: Springer, 1998, pp. 465–494.
[18] F. A. Bayer, F. D. Brunner, M. Lazar, M. Wijnand, and F. Allgöwer, “A tube-based approach to nonlinear explicit MPC,” in Proc. IEEE Conf. Decis. Control, 2016, pp. 4059–4064.
[19] A. Grancharova and T. A. Johansen, Explicit Nonlinear Model Predictive Control: Theory and Applications, New York, NY, USA: Springer Sci. Bus. Media, 2012.

[20] P. A. Giesl and S. F. Hafstein, “Revised CPA method to compute Lyapunov functions for nonlinear systems,” J. Math. Anal. Appl., vol. 410, no. 1, pp. 392–306, 2014.

[21] I. Pejcic, M. Korda, and C. N. Jones, “Control of nonlinear systems with explicit-MPC-like controllers,” in Proc. IEEE Conf. Decis. Control, 2017, pp. 4970–4975.

[22] A. Jadabaie and J. Hauser, “On the stability of receding horizon control with a general terminal cost,” IEEE Trans. Autom. Control, vol. 50, no. 5, pp. 674–678, May 2005.

[23] J. A. Acosta, A. Doria-Cerezo, and E. Fossas, “Stabilisation of state-and-input constrained nonlinear systems via diffeomorphisms: A Sontag’s formula approach with an actual application,” Int. J. Robust Nonlinear Control, vol. 28, no. 13, pp. 4032–4044, 2018.

[24] R. Baier and S. F. Hafstein, “Numerical computation of control Lyapunov functions in the sense of generalized gradients,” in Proc. Int. Symp. Math. Theory Netw. Syst., 2014, pp. 1173–1180.

[25] Y. Nesterov and A. Nemirovskii, Interior point polynomial algorithms in convex programming. Philadelphia, PA, USA: SIAM, 1994.

[26] J. Burkardt, triangulation_refine. Accessed: Sep. 19, 2022. https://people.math.sc.edu/Burkardt/m_src/triangulation_refine/triangulation_refine.html, 2009.

Reza Lavaei (Member, IEEE) received B.Sc. and M.Sc. degrees in mechanical engineering from the Sharif University of Technology and University of Tehran, Tehran, Iran, in 2012 and 2017, respectively, and the Ph.D. degree in mechanical engineering from Duke University, Durham, NC, USA, in 2022. His research interests include constrained control, switched systems, and robust model predictive control.

Leila J. Bridgeman (Member, IEEE) received the B.Sc. and M.Sc. degrees in applied mathematics, and the Ph.D. degree in mechanical engineering from McGill University, Montreal, QC, Canada, in 2008, 2010, and 2016, respectively. She is currently an Assistant Professor with the Thomas Lord Department of Mechanical Engineering and Materials Science, Duke University, Durham, NC, USA. Her research interests include dissipativity-based control, controller synthesis via LMIs, and model predictive control with applications to robotic systems.