

GPU-Parallel Linearization Error Bounds for Real-Time Robust Optimal Control of Nonlinear and Neural Network Dynamics

Jeffrey Fang*

Keyi Shen*

Anutam Srinivasan

Glen Chou

Abstract—This paper studies real-time robust optimal control for uncertain nonlinear systems, where linear time-varying (LTV) approximations make planning tractable but require sound linearization error bounds (LEBs) to guarantee robust constraint satisfaction. We develop tight, differentiable, GPU-parallel LEBs for LTV approximations of nonlinear and neural network (NN) dynamics. For analytic dynamics, we introduce path-based Hessian bounds that are tighter than standard interval methods. For NN dynamics, we derive certified LEBs using NN verifier-generated affine relaxations and local Jacobian corrections. We adapt a GPU-parallel system-level synthesis LTV-based robust control solver to be compatible with these LEBs by extending it to handle right-invertible disturbance matrices and non-zero-centered disturbance sets for tight zonotopic uncertainty propagation. Our method, GPUSLS-LEO, enables online optimization of robust feedback policies that account for linearization error, producing tight, formally verified reachable tubes. On complex nonlinear and NN dynamics up to 168 state dimensions, our method can compute robust control policies on the GPU at rates up to 67 Hz, reducing solve times and conservativeness relative to baselines while preserving formal guarantees and real-time performance.

I. INTRODUCTION

Safe real-time nonlinear control under uncertainty is essential for the resilient operation of robots, drones, and spacecraft. This motivates methods for solving the *robust nonlinear optimal control problem* (RNOCP), which seeks a control policy that *robustly* guarantees constraint satisfaction under worst-case disturbances. In practice, RNOCP solvers typically optimize (1) a nominal trajectory and (2) a stabilizing feedback controller to track it under disturbance. Robust constraint satisfaction is enforced by tightening the nominal constraints using an overapproximation of the closed-loop tracking-error *reachable tube*, ensuring that the closed-loop dynamics remain in a safe subset of the state and input space. However, exact reachability via error propagation through nonlinear dynamics [1]–[3] is generally intractable in real time. To address this, [4]–[7] approximate the nonlinear dynamics along a nominal trajectory with a linear time-varying (LTV) system and compute reachable sets using the LTV dynamics. System-level synthesis (SLS) [4], [8] is one such LTV robust control framework that can be solved in real time over long horizons for high-dimensional systems [9], [10].

Despite strong performance, LTV-based methods only guarantee robust constraint satisfaction given a bound on the *linearization error*, which captures the deviation between the true nonlinear dynamics and their LTV approximation. Existing methods often use global bounds [4], [11], which yield

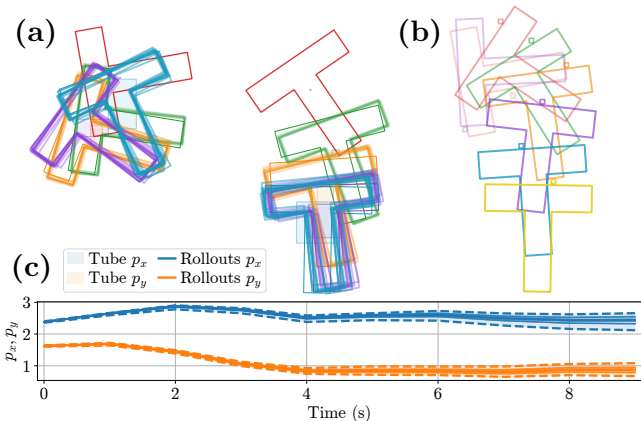


Fig. 1. (a): Robust tubes from GPUSLS-LEO for a neural T pusher system and disturbed rollouts for a rotation and a push trajectory against random disturbances. (b): MPC rollout using our method on real dynamics, successfully moving the T to the goal. (c): Robust tubes of x and y position showing tight tubes and all simulated rollouts staying within the tubes.

conservative (loose) reachable tubes. Tighter *local* bounds around the nominal trajectory are often possible, but must be computed *efficiently* for real-time planning and informative enough to *guide* the optimizer toward safe trajectories with minimal linearization error. Thus, the challenge is to develop bounds that are simultaneously (1) tight, (2) real-time computable, and (3) differentiable, so an RNOCP solver can use them to reshape the nominal trajectory and tighten the reachable tubes. As existing methods do not achieve all three, many LTV-based RNOCP solvers must omit linearization error, forgoing formal guarantees for performance [12], [13].

To close this gap, we propose a family of tight, differentiable linearization error bounds (LEBs) for RNOCP solvers with nonlinear dynamics described by analytic functions and neural networks (NNs), with a GPU-parallel implementation in JAX. For compatibility with the real-time SLS solver of [9], [10], we extend the algorithm with 1) right-invertible disturbance matrices, enabling direct zonotopic propagation of linearization errors and avoiding the conservative overapproximations in [9], and 2) disturbance sets that are not zero-centered. Finally, we show empirically that these bounds can enable real-time, formally-verified RNOCP on the GPU with limited conservativeness for systems with up to 168 states and solve times as low as 15 ms. Our contributions are:

- A set of tight, provably-sound linearization error bounding methods, including path-based interval bounds for analytic systems and NN verifier-based bounds for NN dynamics.
- A new nonlinear SLS formulation that includes linearization error through tight zonotopic propagation, including treatment of right-invertible disturbance matrices and non-zero-centered disturbance sets, guaranteeing robust con-

* Equal contribution. All authors are with the Georgia Institute of Technology, Atlanta, GA, USA, 30308. {jfang301, kshen84, asrinivasan350, chou}@gatech.edu

straint satisfaction for the original nonlinear system.

- A GPU-accelerated nonlinear SLS solver, **GPUSLS-LEO**, that uses a formulation of these linearization bounds in JAX to enable differentiable, GPU-parallel evaluation and gradient-based robust nonlinear optimal control.
- Evaluation on analytical and NN dynamics models, maintaining real-time control rates (up to 67 Hz) for systems up to 168D and problems with $\approx 2 \times 10^5$ decision variables.

II. RELATED WORK

While linearization-free reachability methods exist [1]–[3], direct error propagation through nonlinear dynamics is generally intractable in real time, especially beyond ≈ 10 states [1], [3]. Thus, RNOCP solvers often rely on linearized error propagation for efficiency [13], [14], but introduce linearization error that is difficult to tightly bound, and is often ignored [10], [13]–[15], underapproximated via sampling [4], [11], or handled by assuming access to a linearization bound oracle [16]–[20]. RNOCP solvers that explicitly compute LEBs typically rely on global Lipschitz estimates [11] or interval arithmetic [5]–[7], [21]. Interval arithmetic has been used both to compute reachable sets [22], [23] and to support reachability by constructing linear differential inclusion bounds (LDI) for nonlinear dynamics [24]. In particular, [25] uses Jacobian overbounding to obtain tighter LDIs for reachability, but is not used for control design or directly compute LEBs. Computationally, existing linearization bounding methods are offline [4], slow, or non-differentiable [7], limiting their use for planning. To close these gaps, our method uses interval arithmetic and JAX to compute tight differentiable LEBs for direct use in optimal control. Moreover, we also adapt neural-network (NN) verification tools [26] to derive affine bounds on linearization error for NN dynamics.

As LEBs can be slow to compute [4], they are often generated once offline [4], [11] but are loose if we wish to bound error around *one trajectory*, as in RMPC. Thus, practitioners often omit linearization error, yielding heuristic reachable tubes that trade formal guarantees for performance [12]. To address this, we show that tight, local, GPU-parallel LEBs can be computed online while providing both real-time performance on high-dimensional systems and hard guarantees.

III. PRELIMINARIES AND PROBLEM STATEMENT

Notation: We denote $I_n \in \mathbb{R}^{n \times n}$ as the identity matrix, the Frobenius norm of a matrix $A \in \mathbb{R}^{m \times n}$ as $\|A\|_{\mathcal{F}} := \text{Trace}(A^T A)$. Let $[N] := \{0, \dots, N-1\}$, and $[M, N] := \{M, \dots, N\}$ for $M, N \in \mathbb{N}$. For a matrix $A \in \mathbb{R}^{n \times p}$ we define the row-wise Euclidean norm $\|A\|_{1,r} \in \mathbb{R}^n$ by $\|A\|_{1,r} := [\|A_{1,:}\|_1, \dots, \|A_{n,:}\|_1]^T$. For $a \in \mathbb{R}^n$, we define $\text{diag}(a) \in \mathbb{R}^{n \times n}$ as its diagonal matrix. Given $A \in \mathbb{R}^{m \times n}$ with linearly independent rows, we define $A^\dagger \in \mathbb{R}^{n \times m}$ such that $AA^\dagger = I_m$, where $A^\dagger := A^T(AA^T)^{-1}$. We denote the Minkowski sum of two sets A and B as $A \oplus B := \{a+b \mid a \in A, b \in B\}$. For a function $p(x)$ with $x \in \mathbb{R}^n$ (or explicitly $p((x_1, \dots, x_n))$), we define $\partial_j p(\alpha) := \frac{\partial p}{\partial x_j}|_{x=\alpha}$ and $\partial_{jk} p(\alpha) := \frac{\partial^2 p}{\partial x_j \partial x_k}|_{x=\alpha}$ as the first and second partial derivatives with respect to the j -th and k -th components of the

vector input. We define $\mathbb{I}_n$ as the set of all n -dimensional intervals. Let, $\Delta, \hat{\Delta} \in \mathbb{I}_n$, then $\Delta \cdot \hat{\Delta} := \{\delta \cdot \hat{\delta} \mid \delta \in \Delta, \hat{\delta} \in \hat{\Delta}\}$.

Definitions: We consider uncertain discrete-time nonlinear dynamics

$$x_{k+1} = f(x_k, u_k) + E(x_k)w_k, \quad (1)$$

where $x_k \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ the system state at time step k , $u_k \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ denotes the control input, $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$ the dynamics function, $E : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x \times n_x}$ the disturbance scaling function, and $w_k \in \mathcal{E}_{n_x} := \{w \in \mathbb{R}^{n_x}, \|w\|_\infty \leq 1\}$ the disturbance, normalized to be contained in a unit ℓ_∞ -ball.

We aim to design a controller $\pi(\cdot)$ that solves the following robust nonlinear optimal control problem (RNOCP):

$$\min_{\pi(\cdot)} J(\bar{x}, \pi(\cdot)) \quad (2a)$$

$$\text{s.t. } x_{k+1} = f(x_k, u_k) + E(x_k)w_k, \quad \forall k \in [N], \quad (2b)$$

$$x_0 = \bar{x}_0, \quad (2c)$$

$$u_k = \pi_k(x_0, \dots, x_k), \quad \forall k \in [N], \quad (2d)$$

$$g_k(x_k, u_k) \leq 0, \quad \forall w_k \in \mathcal{E}_{n_x}, \quad \forall k \in [N], \quad (2e)$$

$$g^f(x_N) \leq 0, \quad \forall w_N \in \mathcal{E}_{n_x}. \quad (2f)$$

where $\pi := \{\pi_i\}_{i=0}^{N-1}$ is a sequence of causal control policies, $\bar{x}_0 \in \mathbb{R}^{n_x}$ is the initial state, $g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_c}$ denotes the stagewise state-input constraints, and $g^f : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_f}$ is the terminal constraint. We define the resulting closed-loop reachable set $\Omega_k := (\Omega_k^x, \Omega_k^u)$ of states and controls as

$$\Omega_k := \left\{ (x_k, u_k) \left| \begin{array}{l} x_{j+1} = f(x_j, u_j) + E(x_j)w_j, \quad \forall j \in [k], \\ \forall w_j \in \mathcal{E}_{n_x}, \quad \forall j \in [k], \\ u_j = \pi_j(x_0, \dots, x_j), \quad \forall j \in [k+1]. \end{array} \right. \right\}. \quad (3)$$

Because (2) is an intractable infinite-dimensional problem, RNOCP solvers often simplify it by computing (A) a nominal state-input trajectory and (B) a tracking controller around that trajectory. In this paper, we use one such method, GPUSLS [10], which adapts [9] for real-time solution on the GPU. We then extend [10] to account for linearization error in Sec. V. Our approach uses SLS, which we now review.

A. System Level Synthesis (SLS)

Consider the following uncertain LTV dynamics

$$x_{k+1} = A_k x_k + B_k u_k + E_k w_k. \quad (4)$$

SLS is a control design framework that optimizes over causal *disturbance feedback* controllers

$$u_k = v_k + \sum_{j=0}^{k-1} \Phi_{k,j}^u E_j w_j, \quad (5)$$

with nominal control $v_k \in \mathbb{R}^{n_u}$, i.e., we assign a disturbance feedback matrix $\Phi_{k,j}^u \in \mathbb{R}^{n_u \times n_x}$ for each disturbance $E_j w_j$ and control u_k for $k > j$. For LTV dynamics (4), it can be shown using algebraic manipulations [8] that the resulting closed-loop state sequence can be expressed as

$$x_k = z_k + \sum_{j=0}^{k-1} \Phi_{k,j}^x E_j w_j, \quad z_0 = \bar{x}_0, \quad (6)$$

where $z_k \in \mathbb{R}^{n_x}$ is the nominal state and the closed-loop response $\Phi_{k,j}^x \in \mathbb{R}^{n_x \times n_x}$ captures the effect of disturbance w_j on state x_k . Starting with $\Phi_{j+1,j}^x = I_{n_x}$, SLS propagates

the disturbance via $\Phi_{k+1,j}^x = A_k \Phi_{k,j}^x + B_k \Phi_{k,j}^u$, for all $j \in [N]$ and $k \in [j+1, N-1]$. Since (5)–(6) are the *true* closed-loop trajectory under a given disturbance sequence, the *exact* closed-loop reachable sets are $\Omega_x^k = z_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^x E_j \mathcal{E}_{n_x}$ and $\Omega_k^u = v_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u E_j \mathcal{E}_{n_x}$. To apply the methods of Sec. III-A to the nonlinear dynamics (1), we approximate (1) with the nominal dynamics $z_{k+1} = f(z_k, v_k)$ and LTV error dynamics along a nominal state-control trajectory $\mathbf{z} := \{z_k\}_{k=0}^N \in \mathbb{R}^{n_x \times (N+1)}$, $\mathbf{v} := \{v_k\}_{k=0}^N \in \mathbb{R}^{n_u \times (N+1)}$. For a nominal $(z_k, v_k) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$, we construct an LTV approximation of the form (4) by defining

$$A_k := \left. \frac{\partial f}{\partial x} \right|_{(z_k, v_k)}, B_k := \left. \frac{\partial f}{\partial u} \right|_{(z_k, v_k)}, E_k := E(z_k). \quad (7)$$

Denote $\delta z_k := x_k - z_k$ and $\delta v_k := u_k - v_k$. Then, the residual error $d_k(x_k, u_k, w_k)$ between (1) and its first-order Taylor approximation about (z_k, v_k) , i.e. $f(z_k, v_k) + A_k \delta z_k + B_k \delta v_k$, can be written as,

$$d_k(x_k, u_k, w_k) := E_k w_k + \underbrace{r_k^{\text{lin}} + r_k^{\text{dist}}}_{:= r(x_k, u_k, w_k)}, \quad (8a)$$

$$r_k^{\text{dist}} := (E(x_k) - E_k) w_k, \quad (8b)$$

$$r_k^{\text{lin}} := f(x_k, u_k) - f(z_k, v_k) - A_k \delta z_k - B_k \delta v_k. \quad (8c)$$

Specifically, (8c) captures the Taylor remainder of the nonlinear dynamics f , while (8b) accounts for the error induced by the state dependence of the disturbance matrix E . Given this setup, we study the following problems.

B. Problem Statement

Problem 1 (Linearization Error Bound). For all $k \in [N]$, find a set-valued overapproximation $\bar{\mathcal{E}}_k \subseteq \mathbb{R}^{n_x}$ and associated scaling matrix $\bar{R}_k \in \mathbb{R}^{n_x \times n_x}$ and offset $\bar{c}_k \in \mathbb{R}^{n_x}$ where

$$r_k(x_k, u_k, w_k) \in \bar{\mathcal{E}}_k \subseteq (\bar{R}_k \mathcal{E}_{n_x} \oplus \bar{c}_k), \quad (9)$$

$$\forall (x_k, u_k) \in \Omega_k, \quad \forall w_k \in \mathcal{E}_{n_x}.$$

Problem 2 (Real-time RNOCP). Using $\bar{\mathcal{E}}_k$ to account for the discrepancy between (1) and its LTV approximation, efficiently solve the RNOCP (2) using the LTV surrogate model (4) to robustly satisfy the constraints (2e) and (2f).

IV. GPU-PARALLEL LINEARIZATION ERROR BOUNDS

To streamline exposition we consider a vector-valued function $p(\alpha) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, over the augmented state $\alpha := [x^\top, u^\top]^\top$. Let $\mathcal{A} \subset \mathbb{R}^n$ be a compact set and $\tilde{\alpha} \in \mathcal{A}$ be a nominal point. The linearization error, i.e., the Lagrange remainder of the first-order Taylor expansion of p at $\tilde{\alpha}$ is,

$$r(\alpha; \tilde{\alpha}) := p(\alpha) - p(\tilde{\alpha}) - J(\tilde{\alpha})(\alpha - \tilde{\alpha}), \quad \alpha \in \mathcal{A}. \quad (10)$$

For the specific case when $p(\alpha) = f(x, u)$, (10) coincides with (8c). Our goal is to compute a sound componentwise interval enclosure $\mathcal{R} \in \mathbb{I}_m$ of $r(\mathcal{A}; \tilde{\alpha})$, i.e., $r(\mathcal{A}; \tilde{\alpha}) \subseteq \mathcal{R}$.

A. Linearization Error Bounds for Analytical Dynamics

By Taylor's theorem in Lagrange form, for each component $\ell \in \{1, \dots, m\}$, there exists $\xi \in \mathcal{A}$ such that

$$r_\ell(\alpha; \tilde{\alpha}) = \frac{1}{2} \delta^\top H_\ell(\xi) \delta, \quad \delta := \alpha - \tilde{\alpha}. \quad (11)$$

where $H_\ell(\xi)$ is the ℓ -th item of the Hessian H at ξ . To evaluate the interval enclosure of $r_\ell(\mathcal{A}; \tilde{\alpha})$, previous

work [5]–[7], [21], separately evaluate $\delta^\top, H_\ell(\xi)$ and δ in interval arithmetic, and then perform interval products as in (12), yielding the enclosure $\mathcal{R}_{\ell, \text{classic}}$:

$$r_\ell(\mathcal{A}; \tilde{\alpha}) \subseteq \mathcal{R}_{\ell, \text{classic}} := \frac{1}{2} \Delta^\top [H_\ell](\mathcal{A}) \Delta, \quad (12)$$

where $\Delta := \mathcal{A} \ominus \tilde{\alpha} \subset \mathbb{R}^n$ denotes the displacement set and $[H_\ell](\mathcal{A})$ is an interval enclosure of the Hessian over $\mathcal{A}$ with entries $[H_{\ell, jk}](\mathcal{A})$ (where $H_{\ell, jk}(\alpha) := \partial_{jk} p_\ell(\alpha)$). In practice, $\tilde{\alpha}$ is usually chosen in the center of $\mathcal{A}$ [7], leading to a symmetric Δ for smaller enclosure of $r_\ell(\mathcal{A}; \tilde{\alpha})$.

Retaining the tight quadratic form, (11) can also be first expanded componentwise, followed by performing interval arithmetic on the expansion. It yields (13), where interval Δ_j denotes the j -th slice of Δ , after the symmetry of the interval Hessian is applied. When Δ is symmetric, $\mathcal{R}_{\ell, \text{expand}} \subseteq \mathcal{R}_{\ell, \text{classic}}$ is guaranteed because (13) preserves the non-negativity of δ_k^2 , for all $\delta_k \in \Delta_k$ and for all $k \in [N]$:

$$r_\ell(\mathcal{A}; \tilde{\alpha}) \subseteq \mathcal{R}_{\ell, \text{expand}} = \sum_{1 \leq j < k \leq n} [H_{\ell, kj}](\mathcal{A}) \Delta_j \cdot \Delta_k + \frac{1}{2} \sum_{k=1}^n [H_{\ell, kk}](\mathcal{A}) \Delta_k^2. \quad (13)$$

However, the classical bounds in (12) and (13) evaluate all interval enclosures of the Hessian entries over the *full set* $\mathcal{A}$, causing conservativeness due to dependency and wrapping effects. To mitigate this, we use a *coordinate-wise path* from $\tilde{\alpha}$ to α that enables tighter bounds on second-order terms. For any $\alpha \in \mathcal{A}$, define the sequence of intermediate points

$$\alpha_\alpha^{(k)} := (\alpha_1, \dots, \alpha_k, \tilde{\alpha}_{k+1}, \dots, \tilde{\alpha}_n), \quad k \in [n+1], \quad (14)$$

with $\alpha_\alpha^{(0)} = \tilde{\alpha}$ and $\alpha_\alpha^{(n)} = \alpha$. This defines a *coordinate-wise path* [24], [25] that incrementally moves from $\tilde{\alpha}$ to α along each coordinate. Correspondingly, define the prefix sets $\mathcal{A}^{(k)}(\tilde{\alpha}) := \{\alpha \in \mathcal{A} \mid \alpha_\ell = \tilde{\alpha}_\ell, \forall \ell \in [k+1, n]\}$.

Theorem 1 (Path-Based Hessian Bound): Let $p : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be twice continuously differentiable on a set $\mathcal{A} \subset \mathbb{R}^n$, and let $\tilde{\alpha} \in \mathcal{A}$. Define the displacement set $\Delta := \mathcal{A} \ominus \tilde{\alpha}$. For each $\alpha \in \mathcal{A}$, define the coordinate-wise path $\alpha^{(k)}$ and the corresponding prefix sets $\mathcal{A}^{(k)}(\tilde{\alpha})$. Then, for each component $\ell \in \{1, \dots, m\}$, the linearization error satisfies

$$r_\ell(\mathcal{A}; \tilde{\alpha}) \subseteq \mathcal{R}_{\ell, \text{path}} := \sum_{1 \leq j < k \leq n} [H_{\ell, kj}](\mathcal{A}^{(j)}(\tilde{\alpha})) \Delta_j \cdot \Delta_k + \frac{1}{2} \sum_{k=1}^n [H_{\ell, kk}](\mathcal{A}^{(k)}(\tilde{\alpha})) \Delta_k^2. \quad (15)$$

Proof: Fix $\alpha \in \mathcal{A}$ and define $\delta := \alpha - \tilde{\alpha}$. Using telescoping along the coordinate-wise path, we have $p_\ell(\alpha) - p_\ell(\tilde{\alpha}) = \sum_{k=1}^n (p_\ell(\alpha_\alpha^{(k)}) - p_\ell(\alpha_\alpha^{(k-1)}))$. For each k , applying Taylor's theorem in 1D yields

$$p_\ell(\alpha_\alpha^{(k)}) - p_\ell(\alpha_\alpha^{(k-1)}) = \partial_k p_\ell(\alpha_\alpha^{(k-1)}) \delta_k + \frac{1}{2} \partial_{kk} p_\ell(\xi_k) \delta_k^2, \quad (16)$$

for some $\xi_k \in [\alpha_\alpha^{(k-1)}, \alpha_\alpha^{(k)}] \subseteq \mathcal{A}^{(k)}(\tilde{\alpha})$. Summing (16) over k and subtracting $\nabla p_\ell(\tilde{\alpha})^\top \delta$ yields $p_\ell(\alpha) - p_\ell(\tilde{\alpha}) - \nabla p_\ell(\tilde{\alpha})^\top \delta$ on the LHS which is $r_\ell(\alpha; \tilde{\alpha})$ and the RHS is in $\sum_{k=1}^n (\partial_k p_\ell(\alpha_\alpha^{(k-1)}) - \partial_k p_\ell(\tilde{\alpha})) \delta_k + \frac{1}{2} \sum_{k=1}^n \partial_{kk} p_\ell(\xi_k) \delta_k^2$. Expanding the gradient difference along the same path for k , we have $\partial_k p_\ell(\alpha_\alpha^{(k-1)}) - \partial_k p_\ell(\tilde{\alpha}) = \sum_{j=1}^{k-1} (\partial_k p_\ell(\alpha_\alpha^{(j)}) - \partial_k p_\ell(\alpha_\alpha^{(j-1)}))$. By the Mean Value Theorem, for each $j < k$, there exists $\eta_{j,k} \in [\alpha_\alpha^{(j-1)}, \alpha_\alpha^{(j)}] \subseteq \mathcal{A}^{(j)}$ such that

$\partial_k p_\ell(\alpha_\alpha^{(j)}) - \partial_k p_\ell(\alpha_\alpha^{(j-1)}) = \partial_{kj} p_\ell(\eta_{j,k}) \delta_j$. Substituting yields $\mathbf{r}_\ell(\alpha; \tilde{\alpha}) = \sum_{1 \leq j < k \leq n} \partial_{kj} p_\ell(\eta_{j,k}) \delta_j \delta_k + \frac{1}{2} \sum_{k=1}^n \partial_{kk} p_\ell(\xi_k) \delta_k^2$. Since $\eta_{j,k} \in \mathcal{A}^{(j)}(\tilde{\alpha})$, $\xi_k \in \mathcal{A}^{(k)}(\tilde{\alpha})$, and $\delta \in \Delta$, taking interval enclosures yields (15). ■

Thm. 1 implies that our path-based Hessian bound is never looser than the expanded interval Hessian bound, as each Hessian entry is evaluated over a subset of the full domain.

Corollary 1 (Path-Based Hessian is Never Looser):

When Δ is symmetric, the enclosure in (15) is guaranteed to be no looser than (13) and (12). That is,

$$\mathcal{R}_{\ell, \text{path}} \subseteq \mathcal{R}_{\ell, \text{expand}} \subseteq \mathcal{R}_{\ell, \text{classic}}. \quad (17)$$

Proof: For each $j \in \{1, \dots, n\}$, the prefix set satisfies $\mathcal{A}^{(j)} \subseteq \mathcal{A}$ by construction. Then, for every Hessian entry,

$$[H_{\ell, kj}](\mathcal{A}^{(j)}) \subseteq [H_{\ell, kj}](\mathcal{A}), [H_{\ell, kk}](\mathcal{A}^{(k)}) \subseteq [H_{\ell, kk}](\mathcal{A}).$$

Multiplying by $\Delta_j \Delta_k$ and Δ_k^2 preserves inclusion and summing the resulting terms yields $\mathcal{R}_{\ell, \text{path}} \subseteq \mathcal{R}_{\ell, \text{expand}}$. ■

Unlike $\mathcal{R}_{\ell, \text{classic}}$, $\mathcal{R}_{\ell, \text{expand}}$ and $\mathcal{R}_{\ell, \text{path}}$ preserve componentwise expansion to produce tight, off-centered intervals (via the quadratic form rather than symmetric bounds). We exploit the added tightness in the RNOCP setting (Sec. V).

B. Linearization Error Bounds for Neural Dynamics

For NN dynamics f , computing a tight interval of the Hessian can be difficult. Instead, we use CROWN [26], [27] to obtain affine bounds of an NN over a box input set, and convert them into an interval enclosure of $\mathbf{r}(\alpha; \tilde{\alpha})$.

Theorem 2 (CROWN bounds (Thm. 3.2 [26])): Given an NN, $p: \mathbb{R}^n \rightarrow \mathbb{R}^m$, and a bounded input $\alpha \in \mathcal{A}$ where $\mathcal{A}$ is an interval, there exists linear lower/upper bounds on $p(\alpha)$:

$$\mathbf{W}\alpha + \mathbf{b} \leq p(\alpha) \leq \overline{\mathbf{W}}\alpha + \overline{\mathbf{b}}, \quad (18)$$

where $\mathbf{W}, \overline{\mathbf{W}} \in \mathbb{R}^{m \times n}$ and $\mathbf{b}, \overline{\mathbf{b}} \in \mathbb{R}^m$.

The bound in (18) does not directly bound $\mathbf{r}(\alpha; \tilde{\alpha})$, since the slopes $\mathbf{W}, \overline{\mathbf{W}}$ generally do not match the Jacobian $J(\tilde{\alpha})$. We address this via further relaxation in Thm. 3.

Theorem 3 (Linearization error bound from CROWN):

Let $p: \mathbb{R}^n \rightarrow \mathbb{R}^m$ be an NN, $\mathcal{A} \subset \mathbb{R}^n$ be a box set, $\tilde{\alpha} \in \mathcal{A}$, $J(\tilde{\alpha})$ be the Jacobian of p at $\tilde{\alpha}$, and let $(\mathbf{W}, \mathbf{b}, \overline{\mathbf{W}}, \overline{\mathbf{b}})$ satisfy the CROWN bounds (18) on $\mathcal{A}$. Then, $\forall \alpha \in \mathcal{A}$:

$$\mathbf{r}(\alpha) \leq \mathbf{r}(\alpha; \tilde{\alpha}) \leq \overline{\mathbf{r}}(\alpha), \quad (19)$$

$$\mathbf{r}(\alpha) := (\mathbf{W} - J(\tilde{\alpha}))\alpha + J(\tilde{\alpha})\tilde{\alpha} - p(\tilde{\alpha}) + \mathbf{b}, \quad (20)$$

$$\overline{\mathbf{r}}(\alpha) := (\overline{\mathbf{W}} - J(\tilde{\alpha}))\alpha + J(\tilde{\alpha})\tilde{\alpha} - p(\tilde{\alpha}) + \overline{\mathbf{b}}.$$

Then, an interval enclosure of $\mathbf{r}(\mathcal{A}; \tilde{\alpha}) \subseteq [\mathbf{r}, \overline{\mathbf{r}}]$ is given by

$$\mathbf{r}_i = \min_{\alpha \in \mathcal{A}} \mathbf{r}_i(\alpha), \quad \overline{\mathbf{r}}_i = \max_{\alpha \in \mathcal{A}} \overline{\mathbf{r}}_i(\alpha), \quad \forall i \in [1, m], \quad (21)$$

Proof: Substituting the CROWN lower bound (18) into the expression for $\mathbf{r}(\alpha; \tilde{\alpha})$ (10), we yield $\mathbf{r}(\alpha; \tilde{\alpha}) \geq \mathbf{W}\alpha + \mathbf{b} - p(\tilde{\alpha}) - J(\tilde{\alpha})(\alpha - \tilde{\alpha})$, which simplifies to $\mathbf{r}(\alpha; \tilde{\alpha}) \geq (\mathbf{W} - J(\tilde{\alpha}))\alpha + J(\tilde{\alpha})\tilde{\alpha} - p(\tilde{\alpha}) + \mathbf{b} = \mathbf{r}(\alpha)$. Similarly, we apply the CROWN upper bound. Thus, $\forall \alpha \in \mathcal{A}$: $\mathbf{r}(\alpha) \leq \mathbf{r}(\alpha; \tilde{\alpha}) \leq \overline{\mathbf{r}}(\alpha)$. For bounds valid over $\mathcal{A}$, we take the componentwise minimum of $\mathbf{r}(\alpha)$ and maximum of $\overline{\mathbf{r}}(\alpha)$ over $\mathcal{A}$, as in (21). ■

Thm. 3 reconciles the CROWN slopes and the local linearization slope $J(\tilde{\alpha})$, thereby producing a sound remainder bound around the nominal point. Since $\mathbf{r}(\alpha)$ and $\overline{\mathbf{r}}(\alpha)$ are affine in α , problem (21) is inexpensive when $\mathcal{A}$ is a box set.

C. Zonotope Bounds of $r_k(x_k, u_k, w_k)$

To obtain the zonotope bound for $r_k(x_k, u_k, w_k)$ (9), we compute the enclosure $[\underline{r}_k^{\text{lin}}, \overline{r}_k^{\text{lin}}]$ of the linearization error r_k^{lin} as in Sec. IV-A or IV-B and the enclosure $[\underline{r}_k^{\text{dist}}, \overline{r}_k^{\text{dist}}]$ of the disturbance error r_k^{dist} via interval arithmetic in (22), where $[E](\Omega_k^x)$ is the interval enclosure of E over Ω_k^x :

$$r_k^{\text{dist}} = (E(x_k) - E_k)w_k \in ([E](\Omega_k^x) \ominus E_k)\mathcal{E}_{n_x}. \quad (22)$$

Finally, we obtain $r_k(x_k, u_k, w_k) \in [\underline{r}_k, \overline{r}_k]$, where $\underline{r}_k = \underline{r}_k^{\text{lin}} + \underline{r}_k^{\text{dist}}$ and $\overline{r}_k = \overline{r}_k^{\text{lin}} + \overline{r}_k^{\text{dist}}$. Its zonotope bound is given by $(\bar{R}_k \mathcal{E}_{n_x} \oplus \bar{c}_k)$, where $\bar{c}_k = \frac{1}{2}(\underline{r}_k + \overline{r}_k)$ and $\bar{R}_k = \frac{1}{2}\text{diag}(\overline{r}_k - \underline{r}_k)$. We denote this mapping compactly as

$$\bar{R}_k = \Gamma_r(\Omega_k, z_k, v_k; f, E), \quad \bar{c}_k = \Gamma_c(\Omega_k, z_k, v_k; f, E). \quad (23)$$

D. Differentiable, Parallel Bound Implementation

We implement our LEBs in JAX, with the PBH using `immrax` for interval arithmetic and `jax.verify` providing CROWN-based bounds for NN dynamics. JAX offers differentiability and GPU parallelization, enabling *gradient-based refinement* of the nominal point $\tilde{\alpha}$ and scalable *input domain partitioning* to reduce conservativeness for SLS.

V. USING LINEARIZATION ERROR BOUNDS IN GPUSLS

We discuss the use of our LEBs in SLS (Sec. V-A) and describe an efficient GPU-based implementation (Sec. V-B).

A. Nonlinear SLS Formulation

Following [9, App. A], SLS can be extended to nonlinear systems by planning a nominal trajectory for the disturbance-free dynamics $z_{k+1} = f(z_k, v_k)$ and modeling tracking error dynamics via the LTV system of (4). To solve (2) approximately, we modify the nonlinear SLS formulation of [9, App. A] to be compatible with 1) non-zero-centered disturbance sets and 2) the LEBs given in Sec. IV-A-IV-B:

$$\min_{\mathbf{z}, \mathbf{v}, \Phi} J(\mathbf{z}, \mathbf{v}) + \tilde{H}_0(\Phi, \mathcal{D}) \quad (24a)$$

$$\text{s.t. } z_{k+1} = f(z_k, v_k), \quad \forall k \in [N], \quad z_0 = \bar{x}_0, \quad (24b)$$

$$\Phi_{k+1,j}^x = A_k \Phi_{k,j}^x + B_k \Phi_{k,j}^u, \quad \Phi_{j+1,j}^x = I_{n_x}, \quad (24c)$$

$$\forall j \in [N], \quad \forall k \in [j+1, N-1],$$

$$\bar{E}_k := \mathcal{Z}_r(z_k, v_k, \mu_k, \sigma_k), \quad \forall k \in [N], \quad (24d)$$

$$\bar{c}_k := \mathcal{Z}_c(z_k, v_k, \mu_k, \sigma_k), \quad \forall k \in [N], \quad (24e)$$

$$\sigma_k = h_k^\sigma(\Phi, \mathcal{D}), \quad \mu_k = h_k^\mu(\Phi, \mathcal{D}), \quad \forall k \in [N], \quad (24f)$$

$$g_k(z_k, v_k) + h_k(\Phi, \mathcal{D}) \leq 0, \quad \forall k \in [N], \quad (24g)$$

$$g^f(z_N) + h^f(\Phi, \mathcal{D}) \leq 0, \quad (24h)$$

where A_k, B_k are the linearized dynamics (7) at time k and functions $\mathcal{Z}_r(\cdot), \mathcal{Z}_c(\cdot)$ (24d)–(24e) are the Minkowski sums of the exogenous disturbance and linearization error as a function of the nominal (z_k, v_k) , reachable tube center offset $\mu_k \in \mathbb{R}^{n_x+n_u}$, and widths $\sigma_k \in \mathbb{R}^{n_x+n_u}$ respectively [4], [12], and $\mathcal{D} := \{(\bar{c}_j, \bar{E}_j)\}_{j=0}^{N-1}$. Formally, for all $k \in [N]$ in (24d)–(24e), $\bar{c}_k \in \mathbb{R}^{n_x}$ and $\bar{E}_k \in \mathbb{R}^{n_x \times 2n_x}$ are given as

$$\mathcal{A}_k = ([z_k^\top, v_k^\top]^\top + \mu_k) \oplus \text{diag}(\sigma_k) \mathcal{E}_{n_x+n_u}, \quad (25a)$$

$$\bar{E}_k := [E(z_k) \quad \bar{R}_k] = [E(z_k) \quad \underbrace{\Gamma_r(\mathcal{A}_k, z_k, v_k; f, E)}_{\mathcal{Z}_r(z_k, v_k, \mu_k, \sigma_k)}], \quad (25b)$$

$$\bar{c}_k := \underbrace{\Gamma_c(\mathcal{A}_k, z_k, v_k; f, E)}_{\mathcal{Z}_c(z_k, v_k, \mu_k, \sigma_k)}, \quad \mathcal{Z}_r(z_k, v_k, \mu_k, \sigma_k) \quad (25c)$$

where $\bar{c}_k$ is the offset and $\bar{E}_k$ is the generator matrix of a zonotope for the Minkowski sum of the exogenous disturbance and linearization error: $E(z_k)\mathcal{E}_{n_x} \oplus (\bar{c}_k \oplus \bar{R}_k\mathcal{E}_{n_x}) \equiv \bar{c}_k \oplus \bar{E}_k\mathcal{E}_{2n_x}$. Using (25b) and (25c), the resulting reachable tube overapproximation is,

$$\begin{aligned}\bar{\mathcal{O}}_k^x &:= z_k + \sum_{j=0}^{k-1} \Phi_{k,j}^x \bar{c}_j + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^x \bar{E}_j \mathcal{E}_{2n_x}, \\ \bar{\mathcal{O}}_k^u &:= v_k + \sum_{j=0}^{k-1} \Phi_{k,j}^u \bar{c}_j + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u \bar{E}_j \mathcal{E}_{2n_x}.\end{aligned}\quad (26)$$

Robust constraint satisfaction is enforced through constraint tightenings $h_k(\Phi, \mathcal{D})$ and $h^f(\Phi, \mathcal{D})$. In short, these tightenings capture the propagation of unit-normalized disturbances in (26), which are then scaled by the disturbance bounds $\bar{E}_j$ and shifted by offset $\bar{c}_j$. For simplicity, we derive the tightenings assuming linear constraints, i.e., $g_k(z_k, v_k) = G_k [z_k, v_k]^\top$ and $g^f(z_N) = G^f z_N$, for $G_k \in \mathbb{R}^{n_c \times (n_x + n_u)}$, $G_N \in \mathbb{R}^{n_c \times n_x}$. Valid tightenings can be obtained for nonlinear g_k and g^f by including additional error terms [12, Eq. 22]. Formally, we define the tube center (27) and width (28), stacked (terminal) constraint tightenings (29)–(30), and surrogate system-level response cost (31) as,

$$h_k^\mu(\Phi, \mathcal{D}) = \sum_{j=0}^{k-1} \Phi_{k,j} \bar{c}_j \quad (27)$$

$$h_k^\sigma(\Phi, \mathcal{D}) = \sum_{j=0}^{k-1} \|\Phi_{k,j} \bar{E}_j\|_{1,r} \quad (28)$$

$$h_k(\Phi, \mathcal{D}) = \sum_{j=0}^{k-1} G_k \Phi_{k,j} \bar{c}_j + \|G_k \Phi_{k,j} \bar{E}_j\|_{1,r} \quad (29)$$

$$h^f(\Phi, \mathcal{D}) = \sum_{j=0}^{N-1} G_N \Phi_{N,j}^x \bar{c}_j + \|G_N \Phi_{N,j}^x \bar{E}_j\|_{1,r} \quad (30)$$

$$\begin{aligned}\bar{H}_0(\Phi, \mathcal{D}) &= \sum_{j=0}^{N-1} (\sum_{k=j}^{N-1} (\|\bar{Q}^{1/2} \Phi_{k,j}^x \bar{E}_j\|_{\mathcal{F}}^2 \\ &\quad + \|\bar{R}^{1/2} \Phi_{k,j}^u \bar{E}_j\|_{\mathcal{F}}^2) + \|\bar{Q}_N^{1/2} \Phi_{N,j}^x \bar{E}_j\|_{\mathcal{F}}^2),\end{aligned}\quad (31)$$

where $\Phi_{k,j} := [\Phi_{k,j}^x, \Phi_{k,j}^u]^\top$, Φ collects $\Phi_{k,j}$ for all $k, j \in [N]$, and $\bar{Q}, \bar{Q}_N \in \mathbb{S}_{++}^{n_x}$, $\bar{R} \in \mathbb{S}_{++}^{n_u}$. The constraint tightenings and tube widths in (28)–(30) are derived for the worst case disturbances using the dual norm property [28]. The cost $\bar{H}_0(\Phi, \mathcal{D})$ (31) penalizes uncertainty (the tube widths (28)). Following [9], [10], we solve (24) via an iterative process that alternates between finding a (A) nominal trajectory and a (B) robust controller. To solve (A), we find (z, v) by solving a constraint-tightened NOCP (32),

$$\min_{z, v} J(z, v) \quad (32a)$$

$$\text{s.t. } z_{k+1} = f(z_k, v_k), \quad \forall k \in [N], \quad z_0 = \bar{x}_0, \quad (32b)$$

$$g_k(z_k, v_k) + h_k(\Phi, \mathcal{D}) \leq 0, \quad \forall k \in [N], \quad (32c)$$

$$g^f(z_N) + h^f(\Phi, \mathcal{D}) \leq 0. \quad (32d)$$

Then, (32) mirrors the structure of (2) and can be solved using the GPU-parallel method of [10, Sec. IV]. After solving (A) (32), we linearize around the resulting (z, v) and solve for a robust controller (B) by finding a Φ that optimizes (31):

$$\min_{\Phi^x, \Phi^u} \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\mathcal{Q}_{k,j} \Phi_{k,j} \bar{E}_j\|_{\mathcal{F}}^2 + \|\mathcal{Q}_{N,j} \Phi_{N,j}^x \bar{E}_j\|_{\mathcal{F}}^2 \quad (33a)$$

$$\text{s.t. } \Phi_{k+1,j}^x = A_k \Phi_{k,j}^x + B_k \Phi_{k,j}^u, \quad (33b)$$

$$\Phi_{j+1,j}^x = I_{n_x}, \quad \forall j \in [N], \quad \forall k \in [j+1, N-1], \quad (33c)$$

where $\mathcal{Q}_{k,j}, \mathcal{Q}_{N,j}$ are defined according to [9, Eq. 20, 23] (see [29, App. V]) for consistency between (32) and (33).

Notably, the right-multiplication by $\bar{E}_j$ prevents the cost in (33a) from being expressed as a standard LQR objective in Φ^x and Φ^u . Thus, it is not directly solvable using the Riccati-based methods of [9], [10]. Instead, we solve (34), which has an LQR cost and can be solved via the efficient solvers [9], [10]:

$$\min_{\Phi^x, \Phi^u} \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\mathcal{Q}_{k,j} \Phi_{k,j}\|_{\mathcal{F}}^2 + \|\mathcal{Q}_{N,j} \Phi_{N,j}^x\|_{\mathcal{F}}^2 \quad (34a)$$

$$\text{s.t. } \Phi_{k+1,j}^x = A_k \Phi_{k,j}^x + B_k \Phi_{k,j}^u, \quad (34b)$$

$$\Phi_{j+1,j}^x = I_{n_x}, \quad \forall j \in [N], \quad \forall k \in [j+1, N-1]. \quad (34c)$$

Note that (34) coincides with (33) if $\bar{E}_j = I$ for all $j \in [N]$. In Prop. 1, we prove that the set of minimizers of (33) and (34) are equivalent if $\bar{E}_j$ is right-invertible for all $j \in [N]$ (see [29, App. III] for the proof).

Proposition 1: Let $(\tilde{\Phi}^x, \tilde{\Phi}^u)$ be an optimizer of (33), and let $(\hat{\Phi}^x, \hat{\Phi}^u)$ be an optimizer of (34). If $\bar{E}_j$ is right-invertible for all $j \in [N]$, i.e., there exists $\bar{E}_j^\dagger$ such that $\bar{E}_j \bar{E}_j^\dagger = I_{n_x}$, then the two optimizers coincide: $\tilde{\Phi}^x = \hat{\Phi}^x, \tilde{\Phi}^u = \hat{\Phi}^u$.

Note that since $\bar{R}_k$ is always invertible as generated by the procedure in Sec. IV, $\bar{E}_j$ as given by (25) is always right-invertible for all $j \in [N]$. After solving (34), we compute the constraint tightening terms in (27)–(30). This updates the reachable tube centers μ_k and widths σ_k : $\sigma_k \leftarrow h_k^\sigma(\Phi, \mathcal{D})$ and $\mu_k \leftarrow h_k^\mu(\Phi, \mathcal{D})$, which are used as constraint tightenings in the next iteration of nominal trajectory optimization (32). This iterates until convergence (as in [9, Alg. 1]); in this paper, we terminate after a fixed number of iterations or if the change in the iterates is below a threshold (see [29, App. II] for an algorithm block).

Formulation Novelty: Here, we summarize the changes made to prior nonlinear SLS formulations [9, App. A] [4]:

- 1) Zonotopes enable exact tightening computation (27)–(30) by concatenating the generators of the exogenous disturbance and linearization error, unlike prior ellipsoidal methods [9, App. A], but this yields right-invertible $\bar{E}_j$ incompatible with existing solvers [9], [10]. We address this by setting $\Phi_{j+1,j}^x = I_{n_x}$ and including the disturbance scaling directly into the tightenings and cost (31), with a compatible reformulation (34) justified by Prop. 1.
- 2) Finally, we extend [4], [9] to handle non-zero-centered disturbances, enabling direct use of interval bounds from (23) without extra conservativeness due to zero-centering.

B. Efficient SLS Implementation on the GPU

We propose GPUSLS-Linearization Error Optimization (GPUSLS-LEO), which builds on GPUSLS [10] to explicitly account for linearization error propagation. We first replace the ellipsoidal disturbance representation of [10] with a zonotope, modifying the constraint tightenings and cost terms via (29)–(31). At each controller update, we modify GPUSLS by including a zonotopic bound on the linearization error via (25). We then modify the initialization of Φ^x via (34c).

All other steps closely follow the GPUSLS procedure. In particular, GPUSLS-LEO solves (24) iteratively via sequential quadratic programming (SQP). At each iteration, GPUSLS-LEO quadraticizes the cost and linearizes $x_{k+1} =$

$f(x_k, u_k)$ around the current $(\mathbf{z}, \mathbf{v})$ to obtain A_k, B_k (4). It then solves (A) a local QP approximation of (32) to update $(\mathbf{z}, \mathbf{v})$, (B) (34) to get Φ , and updates (C) the tightenings via (24f). Since our LEBs are differentiable, we can penalize error accumulation during step (A) by modifying the nominal objective (32a) to $\tilde{J}(\mathbf{z}, \mathbf{v}) := J(\mathbf{z}, \mathbf{v}) + \lambda J_{\text{err}}(\mathbf{z}, \mathbf{v})$, where

$$J_{\text{err}}(\mathbf{z}, \mathbf{v}) = \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\Phi_{k,j} \mathcal{Z}_r(z_j, v_j, \mu_j, \sigma_j)\|_{\mathcal{F}}^2, \quad (35)$$

penalizes the propagated residual error (8a). Here, Φ, μ_j, σ_j are from the previous iteration and λ is a weighting term. Notably, (35) biases $(\mathbf{z}, \mathbf{v})$ toward regions of low propagated exogenous and linearization error disturbances. Finally, we show that by solving (24), GPUSLS-LEO ensures containment in the reachable tubes (26), i.e., $x_k \in \Omega_k^x$ and $u_k \in \Omega_k^u$, for all $k \in [N]$. Formally, we have (proof in [29, App. IV]):

Theorem 4: Let $\mathbf{z}, \mathbf{v}$, and Φ be a feasible solution for (24). Then, (26) is guaranteed to overapproximate the true closed-loop reachable set (3), i.e., $\Omega_k \subseteq \tilde{\Omega}_k$ for all $k \in [N]$.

VI. RESULTS

We evaluate LEB tightness (Sec. VI-A) and the robust controllers given by GPUSLS-LEO (Sec. VI-B). Runtime tests use an NVIDIA RTX 4090, except the multi-quadrotor and long-horizon experiments, which require an NVIDIA H200. System definitions appear in [29, App. I].

A. Linearization Error

We compare the LEB from our path-based Hessian (PBH) method on the *Satellite* ($n_x = 7, n_u = 3$) and *Quadrotor* ($n_x = 12, n_u = 4$) systems ([29, App. I]) with several baselines. **Global sampling** [4] estimates worst-case curvature *offline* (taking over 7 minutes [4]) over the *global* set $\mathcal{X} \times \mathcal{U}$, yielding an *empirical* over-approximation. **Interval Hessian (IH) (classic)** [6], [7] evaluates (12) on a *local* set $\mathcal{T} \subset (\mathcal{X} \times \mathcal{U})$. **IH (classic)-CORA** denotes the standard IH implementation in CORA [6], [30]. **Random sampling** uses 10K samples from the local set and gives an *under-approximation*. We sample nominal points (z, v) from $\mathcal{X} \times \mathcal{U}$ and define local sets $\mathcal{T} := (z, v) \oplus \epsilon \mathcal{E}_{n_x+n_u}$ with $\epsilon = 0.01, 0.05, 0.1, 0.15, 0.2$. Fig. 2 shows the mean interval width of (23). For *Satellite*, where nonlinearity is mild, all over-approximation methods remain close to the random-sampling lower bound as ϵ grows, though PBH is still the tightest. For *Quadrotor*, PBH consistently gives the tightest bounds. By contrast, global sampling is loose at small ϵ , suggesting global curvature misses local behavior, while the interval Hessian becomes overly conservative at $\epsilon = 0.20$. Table I compares *online* runtime. PBH runs in under 1 ms, enabling real-time use. Although slower than the global bound, which is efficiently calculated online after slow offline computation, PBH is 60% and 27% (satellite and quadrotor respectively) faster than dense random sampling and negligible running time compared to CORA IH.

TABLE I

RUNTIME (IN MS) COMPARISON OF LINEARIZATION BOUND METHODS.

System	Random	Global	IH-classic	IH-classic-CORA	Path-based
Satellite	0.894	0.143	0.218	11570	0.361
Quadrotor	1.278	0.156	0.453	227200	0.936

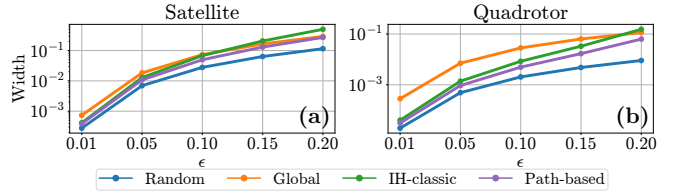


Fig. 2. (a): LEBs on the satellite system. (b): LEBs on the quadrotor system. In both systems, our method achieves the tightest over-approximation.

TABLE II

LINEARIZATION ERROR UNDER NOMINAL OPTIMIZATION VIA PGD.

System	Initial	Sampling	PGD	PGD better (%)
Satellite	0.2680	0.2365	0.2241	100
Quadrotor	0.02253	0.01639	0.01590	100

To quantify the benefit of differentiability in our path-based Hessian implementation, we search for a nominal point within a fixed perturbation radius ϵ that minimizes the LEB size. Starting from random nominal points, we run projected gradient descent (PGD) within a local neighborhood of radius ϵ_{opt} to reduce the error interval width. We compare PGD with random sampling of nominal points under the same time budget, where both methods use the PBH to evaluate linearization error. Table II shows that across 100% of test cases, PGD yields smaller error interval widths than random sampling across both systems, showing the value of gradient-based optimization for finding nominal points with tighter LEBs, which can benefit downstream SLS.

B. Using Linearization Bounds for RNOCP

In this section, we evaluate GPUSLS-LEO against a suite of analytical and neural dynamical systems against various RNOCP baselines. We further assess the impact of key ablations, including the effect of optimizing over the linearization error gradients and the use of nonzero-centered zonotopes. All benchmarked methods are evaluated under the same disturbance set for the given dynamical system and, where applicable, are run for a maximum of 100 SQP iterations.

1) *Satellite*: We evaluate our method on a 7D satellite attitude control task, where the objective is to stabilize the system to a desired orientation under bounded disturbances. We adopt the experimental setup of [4] and benchmark against their approach. In Fig. 3(a) we compare the tube widths σ_k for: **GPUSLS-LEO**, **GPUSLS-E** (ellipsoidal propagation as in [9, App. A]), **GPUSLS** (no linearization error propagation) [10], and **NL-SLS** ([4]). Our method achieves the tightest tubes among all baselines. We have on average 20% smaller tubes than NL-SLS, which relies on a global maximum Hessian bound and is therefore locally overly-conservative along a trajectory. Compared to GPUSLS-E, our method achieves 45% smaller tubes, as it involves overapproximating the Minkowski sum of ellipsoids, whereas our zonotope-based combination is exact. We observe that incorporating our LEB pipeline does not introduce significant conservativeness, as the tube widths increase by only 4% compared to GPUSLS. In Fig. 3(b), we show the rollouts of 20 random and 64 adversarial disturbances for the quaternion states, showing 100% containment under our method.

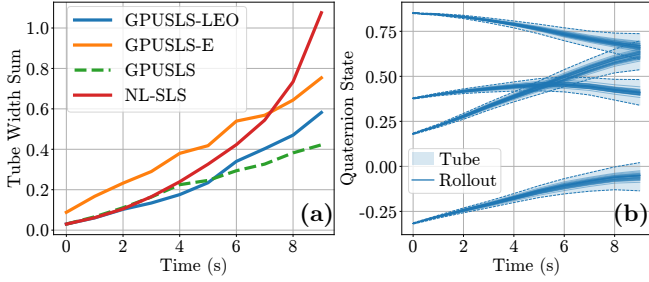


Fig. 3. **Satellite (7D)**. (a): Tube sizes for our method, GPUSLS-LEO, compared to baselines and the variant without linearization error. Our method is the least conservative, while minimally increasing tube size (b): Tubes for the first four state dimensions; our tubes capture 100% of sampled rollouts.

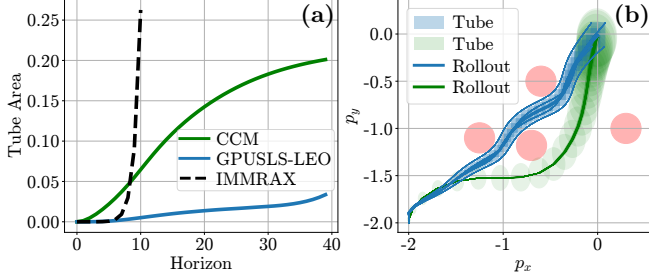


Fig. 4. **Planar quadrotor (6D)**. (a): Tube area comparison against CCM and *immrax*, showing our method is less conservative. (b): Tubes and rollouts for GPUSLS-LEO and CCM. Both methods achieve 100% safety, but the conservativeness of the CCM forces the system to take a suboptimal path.

2) *Planar quadrotor*: We evaluate our method on a 6D planar quadrotor against control contraction metrics (CCM) [31]. Both methods must navigate through a dense obstacle field over a horizon of 40 steps. Fig. 4(a) shows our method achieving 91% smaller tube area than the CCM for the (x, y) position states. This arises because the CCM certifies safety over the entire state space, introducing inherent conservativeness. GPUSLS-LEO, on the other hand, certifies safety along a specific trajectory, enabling tighter, trajectory-dependent bounds. We also evaluate the closed-loop system on an interval-based reachability analysis tool *immrax* [24], showing its calculated tube area diverging after a few steps.

In Fig. 4(b) we compare the calculated trajectories and tubes for our method and CCM. Our approach is less conservative, allowing a more direct, task-optimal path, while the CCM is forced to take a longer route. In Fig. 5(a), we show the trajectory and tubes over a long horizon of 7500 steps, which involves optimizing a state-control trajectory with 1.95×10^5 decision variables. Due to the tightness of our LEBs and the optimization of the closed-loop controller in SLS, our tubes remain well-bounded and finite over the horizon (Fig. 5(b)), demonstrating its scalability to long horizons.

3) *Coupled 3D Quadrotor*: To demonstrate scalability to high dimensional systems, we evaluate our method on a coupled system of 14 3D quadrotors (168D) over a horizon of 25. The system is tasked with navigating through an obstacle field while the quadrotors are coupled via a spring force toward the group centroid. We show the planned trajectories and the tubes in Fig. 6(a), and observe all rollouts remain in the tight tubes despite the high dimensional state space, demonstrating the method's scalability to state dimensionality. In Fig. 6(b), we plot the tubes and the corresponding

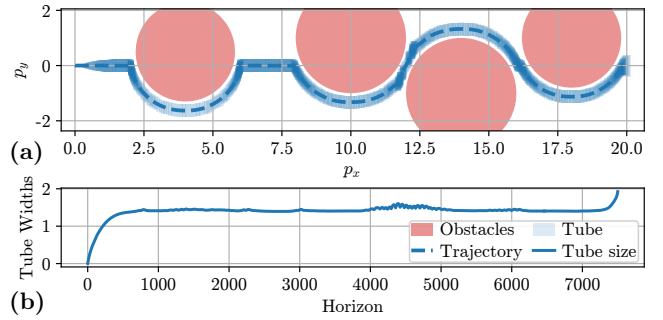


Fig. 5. **Planar quadrotor (6D)**. (a): System navigating through an obstacle field over a 20 m trajectory with a planning horizon length of 7500. (b): Tube widths along the horizon. Despite the long horizon, the tubes remain bounded due to the GPUSLS-LEO controller's optimization of tube sizes.

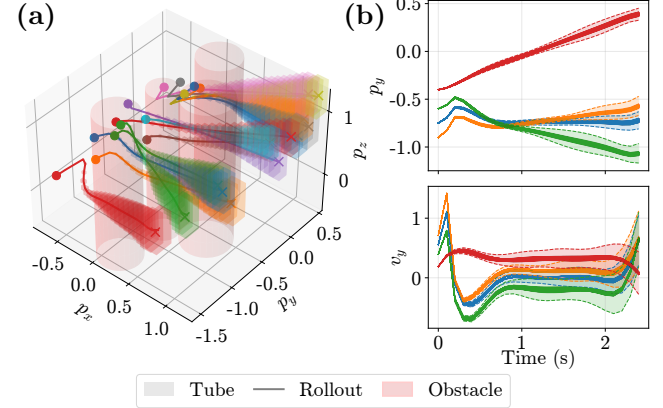


Fig. 6. **System of 14 coupled 3D quadrotors (168D)**. (a): Computed trajectories and tubes in an obstacle field. Even at high dimensions, our method is able to compute tight tubes. (b): Tubes and 100 rollouts for states p_y and v_y for 4 quadrotors. 100% of rollouts stay safely in the tubes.

rollouts for p_y and v_y for 100 adversarial disturbances.

4) *Neural T-pusher*: We demonstrate that our formulation can account for linearization error propagation through learned neural dynamics. We train an MLP-based model for the T-pusher system and task it with rotating and pushing the object to the goal configuration (Fig. 1). We evaluate our method on several ablations (Fig. 7(a)), including GPUSLS, GSL (shorthand for GPUSLS-LEO), *GSL-C* (zero-centered zonotopes) and *GSL* ($\lambda = 1$), which incorporates a penalty on the linearization error as defined in V-B. Compared to GPUSLS, GPUSLS-LEO produces tubes that are on average 99% larger. However, Fig. 7(b) shows that our method achieves 100% containment of all rollouts, while Fig. 7(c) illustrates how trajectories from GPUSLS fall out of the tube, showing the need to formally consider linearization error.

Fig. 7(a) shows that adding linearization error penalties can reduce conservativeness, with $\lambda = 1$ decreasing tube widths by 8%. We also show the importance of nonzero-centered zonotopes, as using zero-centered zonotopes leads to 25% larger tubes. We present rollouts and disturbance deviations in Fig. 1(a). In Fig. 1(b), we demonstrate the method's ability for closed-loop MPC on simulated contact dynamics, where the pusher successfully completes the task. We note that we do not formally consider learning error; however, it can be incorporated as in [32].

5) *Runtime*: We evaluate the runtime of our method across all systems. In Tab. III, we show the per iteration

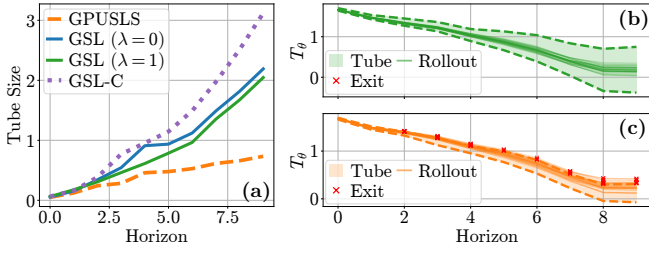


Fig. 7. **Neural T-pusher (5D).** (a): Tube size comparison across ablations of our method, showing nonzero-centered tubes and linearization error gradients reduce tube widths. (b): Rollouts of our method; all trajectories remain in the computed robust tubes. (c): Rollouts of GPUSLS. Without formally considering linearization error, rollouts leave the tube.

TABLE III

PER ITERATION RUNTIME BREAKDOWN (MS) ACROSS SYSTEMS.

System	GPUSLS	Lin. Error Grad.	Remainder	Total	DT
Satellite	11	3	< 1	15	1000
Planar Quadrotor	18	1	< 1	20	150
Coupled Quadrotor (96D)	40	—	48	88	100
Neural T-Dynamics	630	100	32	762	1000

breakdown in milliseconds of the different components of our methodology. In all systems aside from the coupled quadrotor, the main computation load is in the GPUSLS, showing that our proposed modifications do not drastically increase run time. We also show that all total runtimes fall below the discretization time step DT, demonstrating the viability of our method under a real-time iteration MPC scheme [10].

Finally, we evaluate the method’s scaling with growing state and control dimensions using an increasing number of coupled quadrotors. Fig. 8 shows that GPUSLS incurs higher initial runtime but scales favorably with state and control dimension due to its logarithmic scaling [10]. PBH on the other hand, exhibits low initial runtime, but can slow down in higher dimensions due to the cost of Hessian computations.

VII. CONCLUSION

This paper presents a framework for computing tight, differentiable, and sound linearization error bounds (LEBs) for nonlinear analytical and NN dynamics and for using them in GPUSLS-LEO, a real-time GPU-parallel RNOCP solver based on SLS. The approach reduces conservativeness, outperforming state-of-the-art reachability analysis tools, while maintaining formal guarantees and achieving performance on systems with up to 168 states, real-time control rates (up to 67 Hz), and problems with $\approx 2 \times 10^5$ decision variables.

REFERENCES

- [1] A. Majumdar and R. Tedrake, “Funnel libraries for real-time robust feedback motion planning,” *IJRR*, vol. 36, no. 8, 2017.
- [2] M. Althoff, “Reachability analysis of nonlinear systems using conservative polynomialization and non-convex sets,” in *HSCC*, 2013.
- [3] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, “Hamilton-jacobi reachability: A brief overview and recent advances,” in *CDC*, 2017.
- [4] A. Leeman, J. Köhler, A. Zanelli, S. Bennani, and M. Zeilinger, “Robust nonlinear optimal control via system level synthesis,” *TAC*, 2025.
- [5] L. Schäfer and M. Althoff, “Robust optimal control using set-based reachability analysis,” in *Eur. Control Conf. (ECC)*. IEEE, 2025.
- [6] D. D. Leister and J. P. Koeln, “Robust model predictive control for nonlinear discrete-time systems using iterative time-varying constraint tightening,” in *American Control Conference (ACC)*. IEEE, 2025.
- [7] M. Althoff, O. Stursberg, and M. Buss, “Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization,” in *CDC*. IEEE, 2008, pp. 4042–4048.

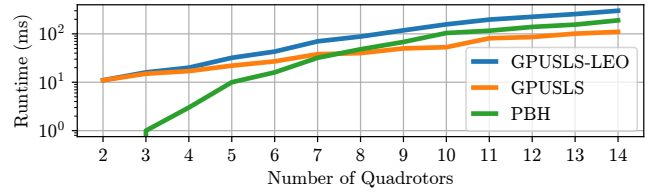


Fig. 8. **Coupled Quadrotor.** Per-iteration runtime evaluation across increasing numbers of coupled quadrotors. GPUSLS scales favorably with dimension state, growing approximately logarithmically with state and control space. Path-Based Hessians (PBH) achieves low runtimes at small dimensions but scales more poorly as the state dimension increases.

- [8] J. Anderson, J. C. Doyle, S. H. Low, and N. Matni, “System level synthesis,” *Annu. Rev. Control.*, vol. 47, pp. 364–393, 2019.
- [9] A. P. Leeman, J. Köhler, F. Messerer, A. Lahr, M. Diehl, and M. N. Zeilinger, “Fast system level synthesis: Robust model predictive control using riccati recursions,” *IFAC*, vol. 58, no. 18, 2024.
- [10] J. Fang and G. Chou, “Safe large-scale robust nonlinear mpc in milliseconds via reachability-constrained system level synthesis on the gpu,” in *Under review*, 2026.
- [11] T. Kim, P. Elango, and B. Açıkmeşe, “Joint synthesis of trajectory and controlled invariant funnel for discrete-time systems with locally lipschitz nonlinearities,” *IJRRNC*, vol. 34, no. 6, 2024.
- [12] S. Zhan, C.-Y. Chiu, A. Leeman, and G. Chou, “Robustly constrained dynamic games for uncertain nonlinear dynamics,” in *ICRA*, 2026.
- [13] F. Messerer and M. Diehl, “An efficient algorithm for tube-based robust nonlinear optimal control with optimal linear feedback,” 2021.
- [14] Z. Manchester and S. Kuindersma, “Dirtrel: Robust nonlinear direct transcription with ellipsoidal disturbances and lqr feedback,” 2017.
- [15] M. Diehl, H. G. Bock, and E. Kostina, “An approximation technique for robust nonlinear optimization,” *Math. Prog.*, vol. 107, no. 1, 2006.
- [16] B. Houska, F. Logist, J. Van Impe, and M. Diehl, “Robust optimization of nonlinear dynamic systems with application to a jacketed tubular reactor,” *Journal of Process Control*, vol. 22, no. 6, 2012.
- [17] M. Villanueva, R. Quirynen, M. Diehl, B. Chachuat, and B. Houska, “Robust mpc via min–max differential inequalities,” *Automatica*, 2017.
- [18] M. Cannon, D. Ng, and B. Kouvaritakis, “Successive linearization nmpe for a class of stochastic nonlinear systems,” in *NMPC*, 2009.
- [19] M. Cannon, J. Buerger, B. Kouvaritakis, and S. Rakovic, “Robust tubes in nonlinear model predictive control,” *TAC*, vol. 56, no. 8, 2011.
- [20] A. Richards, “Robust model predictive control for time-varying systems,” in *CDC*. IEEE, 2005, pp. 3747–3752.
- [21] M. Runger and M. Zamani, “Accurate reachability analysis of uncertain nonlinear systems,” in *HSCC*, 2018, pp. 61–70.
- [22] D. Limon, J. Bravo, T. Alamo, and E. Camacho, “Robust mpc of constrained nonlinear systems based on interval arithmetic,” *IEE Proceedings-Control Theory and Applications*, vol. 152, no. 3, 2005.
- [23] K. Shen and G. Chou, “Parallel differentiable reachability for learning and planning with certified neural dynamics and controllers,” 2026.
- [24] A. Harapanahalli, S. Jafarpour, and S. Coogan, “immrax: A parallelizable and differentiable toolbox for interval analysis and mixed monotone reachability in jax,” *ADHS*, vol. 58, no. 11, 2024.
- [25] A. Harapanahalli and S. Coogan, “A linear differential inclusion for contraction analysis to known trajectories,” *IEEE TAC*, 2025.
- [26] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, “Efficient neural network robustness certification with general activation functions,” *NeurIPS*, vol. 31, 2018.
- [27] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh, “Automatic perturbation analysis for scalable certified robustness and beyond,” *NeurIPS*, 2020.
- [28] W.-C. Li and G. Chou, “Certified gradient-based contact-rich manipulation via smoothing-error reachable tubes,” *arXiv*, 2026.
- [29] J. Fang, K. Shen, A. Srinivasan, and G. Chou, “Gpu-parallel linearization error bounds for real-time robust optimal control of nonlinear and neural network dynamics (extended),” <https://bit.ly/4c39Pai>, 2026.
- [30] M. Althoff, “An introduction to CORA 2015,” in *Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.
- [31] A. Sasfi, M. N. Zeilinger, and J. Köhler, “Robust adaptive mpc using control contraction metrics,” *Automatica*, vol. 155, p. 111169, 2023.
- [32] A. Srinivasan, A. Leeman, and G. Chou, “Safety beyond the training data: Robust out-of-distribution mpc via conformed system level synthesis,” *LADC*, 2026.

APPENDIX

In the appendix, we first discuss details on the system dynamics used in the experimental results (App. I). We then provide an algorithm block describing how sequential quadratic programming is used to solve GPUSLS-LEO (App. II). Next, we discuss and prove Proposition 1 and a supporting lemma in App. III. We then provide proofs for the remaining theoretical results that were omitted from the main manuscript in App. IV. Finally, we discuss in more detail the Lagrange multiplier variable τ used in our framework in App. V.

APPENDIX I DYNAMICAL SYSTEM DEFINITIONS

In this section, we give an overview of the discrete-time dynamical system definitions for each system. The discrete-time dynamics are obtained via forward Euler discretization with timestep Δt , $x_{k+1} = x_k + \Delta t(\dot{x}_k)$.

A. Satellite

We use the same satellite dynamics and experimental setup as [4]. Specifically, we define the dynamics as,

$$\dot{x} = \begin{bmatrix} \zeta(\omega)q \\ I_S^{-1}(v - w \times (I_S \omega)) \end{bmatrix} \quad (36a)$$

$$\zeta(\omega) := \frac{1}{2} \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \quad (36b)$$

with states $x := (q, \omega) \in \mathbb{R}^7$, where $q \in \mathbb{R}^4$ is the attitude quaternion, $\omega \in \mathbb{R}^3$ the angular rotation rate, $v \in \mathbb{R}^3$ the input control torque, and $I_S = \text{diag}(5, 2, 1)$ the symmetric inertia matrix. We define a constant disturbance scaling of $E = 5 \cdot 10^{-3} \cdot \text{diag}(0, 0, 0, 0, 1, 1, 1)$.

B. Planar Quadrotor

We consider a planar quadrotor with the following dynamics

$$\dot{x} = \begin{bmatrix} v_x \\ v_y \\ \dot{\phi} \\ -\frac{1}{m}(u_1 + u_2) \sin(\phi) \\ \frac{1}{m}(u_1 + u_2) \cos(\phi) - g \\ \frac{L}{J}(u_2 - u_1) \end{bmatrix} \quad (37a)$$

The state is defined as $x := (p_x, p_y, \phi, v_x, v_y, \dot{\phi}) \in \mathbb{R}^6$ and input $u := (u_1, u_2) \in \mathbb{R}^2$. The position is denoted as (p_x, p_y) , ϕ the pitch angle, (v_x, v_y) the translational velocities, and $\dot{\phi}$ the angular velocity. The inputs (u_1, u_2) are the individual rotor thrusts. We define the mass $m = 2.0576$, gravitational acceleration $g = 9.81$, arm length $L = 0.25$ and moment of inertia $J = 0.01$. We define the disturbance scaling as $E = 5 \cdot 10^{-2} \cdot \text{diag}(0, 0, 0, 1, 1, 0)$.

C. Coupled 3D Quadrotor

We consider a system of N quadrotors. For a given quadrotor i , we define its dynamics as

$$\dot{\xi}_i = \begin{bmatrix} \dot{p}_{x,i} \\ \dot{p}_{y,i} \\ \dot{p}_{z,i} \\ \dot{\phi}_i \\ \dot{\theta}_i \\ \dot{\psi}_i \\ \dot{v}_{x,i} \\ \dot{v}_{y,i} \\ \dot{v}_{z,i} \\ \dot{p}_i \\ \dot{q}_i \\ \dot{r}_i \end{bmatrix} = \begin{bmatrix} v_{x,i} \\ v_{y,i} \\ v_{z,i} \\ p_i + (q_i \sin \phi_i + r_i \cos \phi_i) \tan \theta_i \\ q_i \cos \phi_i - r_i \sin \phi_i \\ \frac{q_i \sin \phi_i + r_i \cos \phi_i}{\cos \theta_i} \\ \frac{T_i}{m} (\cos \psi_i \sin \theta_i \cos \phi_i + \sin \psi_i \sin \phi_i) \\ \frac{T_i}{m} (\sin \psi_i \sin \theta_i \cos \phi_i - \cos \psi_i \sin \phi_i) \\ \frac{T_i}{m} (\cos \theta_i \cos \phi_i) - g \\ \frac{J_y - J_z}{J_x} q_i r_i + \frac{\tau_{\phi,i}}{J_x} \\ \frac{J_z - J_x}{J_y} p_i r_i + \frac{\tau_{\theta,i}}{J_y} \\ \frac{J_x - J_y}{J_z} p_i q_i + \frac{\tau_{\psi,i}}{J_z} \end{bmatrix}. \quad (38a)$$

The state of quadrotor i is defined as $\xi_i := (p_{x,i}, p_{y,i}, p_{z,i}, \phi_i, \theta_i, \psi_i, v_{x,i}, v_{y,i}, v_{z,i}, p_i, q_i, r_i) \in \mathbb{R}^{12}$ where $(p_{x,i}, p_{y,i}, p_{z,i})$ denotes position, $(\phi_i, \theta_i, \psi_i)$ the roll, pitch, and yaw angles, $(v_{x,i}, v_{y,i}, v_{z,i})$ the translational velocities, and (p_i, q_i, r_i) denote body angular velocities. The input is defined as $v_i := (T_i, \tau_{\phi,i}, \tau_{\theta,i}, \tau_{\psi,i}) \in \mathbb{R}^4$, where T_i is the collective thrust and $(\tau_{\phi,i}, \tau_{\theta,i}, \tau_{\psi,i})$ are the body torques.

For a system of N coupled quadrotors, we stack the states and inputs, giving the overall state and control vectors,

$$x := (\xi_1, \dots, \xi_N) \in \mathbb{R}^{12N} \quad u := (v_1, \dots, v_N) \in \mathbb{R}^{4N}. \quad (39)$$

We additionally incorporate a centroid spring-damper coupling in the translational dynamics. We define the centroid position and velocity as

$$\bar{p}_k = \frac{1}{N} \sum_{i=1}^N p_{i,k}, \quad \bar{v}_k = \frac{1}{N} \sum_{i=1}^N v_{i,k}. \quad (40)$$

The coupling acceleration applied to quadrotor i is

$$a_{i,k}^{\text{coup}} = -\alpha N(p_{i,k} - \bar{p}_k) - \gamma(v_{i,k} - \bar{v}_k) \quad (41)$$

where α and γ are the coupling gains, with α specifying the stiffness of the spring term and γ controlling the damping coefficient. In our experiments, we set $\alpha = 0.75$ and $\gamma = 0.25$. Therefore, the discrete-time dynamics become

$$\xi_{i,k+1} = \xi_{i,k} + \Delta t \begin{bmatrix} \dot{p}_{x,i,k} \\ \dot{p}_{y,i,k} \\ \dot{p}_{z,i,k} \\ \dot{\phi}_{i,k} \\ \dot{\theta}_{i,k} \\ \dot{\psi}_{i,k} \\ \dot{v}_{i,k} + a_{i,k}^{\text{coup}} \\ \dot{p}_{i,k} \\ \dot{q}_{i,k} \\ \dot{r}_{i,k} \end{bmatrix}. \quad (42)$$

We use a disturbance scaling matrix of $E = 2 \cdot 10^{-2} \cdot I_{12N}$

APPENDIX II
ALGORITHM BLOCK

Algorithm 1 GPUSLS-LEO

Require: $\bar{x}_0, J, f(x, u), E(x), g_k(x, u), g^f(x)$

- 1: **while** SQP NOT CONVERGED **do**
- 2: $\mathbf{A}, \mathbf{B}, \mathbf{G} \leftarrow \text{SQP: LINEARIZE SYSTEM}$
- 3: $\mathbf{Q}, \mathbf{M}, \mathbf{R}, \mathbf{q}, \mathbf{r} \leftarrow \text{SQP: QUADRATICIZE COST}$
- 4: **while** SLS NOT CONVERGED **do**
- 5: $\mathbf{z}, \mathbf{v}, \leftarrow \text{OPTIMIZE NOMINAL TRAJECTORY}$
- 6: $\boldsymbol{\sigma}, \boldsymbol{\mu} \leftarrow \text{COMPUTE INTERVAL}$
- 7: $\bar{\mathbf{E}}, \bar{\mathbf{c}} \leftarrow \text{COMPUTE RESIDUAL ERROR ZONOTOPE}$
- 8: $\boldsymbol{\tau}, \mathbf{Q} \leftarrow \text{UPDATE DUALS AND COST}$
- 9: $\boldsymbol{\Phi}^x, \boldsymbol{\Phi}^u \leftarrow \text{OPTIMIZE CONTROLLER}$
- 10: $h(\boldsymbol{\Phi}, \mathcal{D}), h^f(\boldsymbol{\Phi}, \mathcal{D}) \leftarrow \text{UPDATE TIGHTENING}$
- 11: **end while**
- 12: **end while**
- 13: **return** $\mathbf{z}, \mathbf{v}, \boldsymbol{\Phi}^x, \boldsymbol{\Phi}^u$

Here, $\mathbf{Q}, \mathbf{M}, \mathbf{R}, \mathbf{q}, \mathbf{r}$ correspond to the quadratic approximation of the nominal cost.

APPENDIX III
ON SOLVING FOR $\boldsymbol{\Phi}$

Solving (33) yields optimal values for $\boldsymbol{\Phi}^x$ and $\boldsymbol{\Phi}^u$ which penalize the size of the tightenings (31). However, this leads to terms in (33a) that scale the response matrices $\boldsymbol{\Phi}^x$ and $\boldsymbol{\Phi}^u$ by $\bar{E}_j$ in the cost (33a). This remains a quadratic cost in $\boldsymbol{\Phi}$ but no longer maintains the structure of an LQR cost function. To address this, we instead solve (34) which has an LQR structure and can be solved via Ricatti recursions. For self-containment, we restate the original problem (33) and the reformulated problem (34), and the equivalent optimizer Proposition (Prop. 1) below:

Original Problem:

$$\min_{\boldsymbol{\Phi}^x, \boldsymbol{\Phi}^u} \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\mathcal{Q}_{k,j} \boldsymbol{\Phi}_{k,j} \bar{E}_j\|_{\mathcal{F}}^2 + \|\mathcal{Q}_{N,j} \boldsymbol{\Phi}_{N,j}^x \bar{E}_j\|_{\mathcal{F}}^2 \quad (33a)$$

$$\text{s.t. } \boldsymbol{\Phi}_{k+1,j}^x = A_k \boldsymbol{\Phi}_{k,j}^x + B_k \boldsymbol{\Phi}_{k,j}^u, \quad (33b)$$

$$\boldsymbol{\Phi}_{j+1,j}^x = I_{n_x}, \quad \forall j \in [N], \quad \forall k \in [j+1, N-1], \quad (33c)$$

Reformulated Problem:

$$\min_{\boldsymbol{\Phi}^x, \boldsymbol{\Phi}^u} \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\mathcal{Q}_{k,j} \boldsymbol{\Phi}_{k,j}\|_{\mathcal{F}}^2 + \|\mathcal{Q}_{N,j} \boldsymbol{\Phi}_{N,j}^x\|_{\mathcal{F}}^2 \quad (34a)$$

$$\text{s.t. } \boldsymbol{\Phi}_{k+1,j}^x = A_k \boldsymbol{\Phi}_{k,j}^x + B_k \boldsymbol{\Phi}_{k,j}^u, \quad (34b)$$

$$\boldsymbol{\Phi}_{j+1,j}^x = I_{n_x}, \quad \forall j \in [N], \quad \forall k \in [j+1, N-1] \quad (34c)$$

In Prop. 1, we prove that the optimizer sets of (33c) and (34c) are equivalent.

Proposition 1 (Equivalent Optimizers): Let $(\tilde{\boldsymbol{\Phi}}^x, \tilde{\boldsymbol{\Phi}}^u)$ be an optimizer of (33), and let $(\hat{\boldsymbol{\Phi}}^x, \hat{\boldsymbol{\Phi}}^u)$ be an optimizer of the reformulated problem (34). Suppose that $\bar{E}_j$ is right-invertible for all $j \in [N]$, i.e., there exists $\bar{E}_j^\dagger$ such that $\bar{E}_j \bar{E}_j^\dagger = I_{n_x}$. Then the two optimizers coincide: $\tilde{\boldsymbol{\Phi}}^x = \hat{\boldsymbol{\Phi}}^x, \tilde{\boldsymbol{\Phi}}^u = \hat{\boldsymbol{\Phi}}^u$.

Proof: Observe that (33) can be rewritten into (43) with the variable change $\boldsymbol{\Psi}_{k,j} = \boldsymbol{\Phi}_{k,j} \bar{E}_j$, since by using (33b) and (33c) we have that,

$$\boldsymbol{\Phi}_{k+1,j}^x \bar{E}_j = A_k \boldsymbol{\Phi}_{k,j}^x \bar{E}_j + B_k \boldsymbol{\Phi}_{k,j}^u \bar{E}_j \iff \text{Eq. (43b)},$$

and

$$\boldsymbol{\Phi}_{j+1,j}^x \bar{E}_j = I_{n_x} \bar{E}_j \iff \text{Eq. (43c)}.$$

$$\min_{\boldsymbol{\Psi}^x, \boldsymbol{\Psi}^u} \sum_{j=0}^{N-1} \sum_{k=j}^{N-1} \|\mathcal{Q}_{k,j} \boldsymbol{\Psi}_{k,j}\|_{\mathcal{F}}^2 + \|\mathcal{Q}_{N,j} \boldsymbol{\Psi}_{N,j}^x\|_{\mathcal{F}}^2 \quad (43a)$$

$$\text{s.t. } \boldsymbol{\Psi}_{k+1,j}^x = A_k \boldsymbol{\Psi}_{k,j}^x + B_k \boldsymbol{\Psi}_{k,j}^u, \quad (43b)$$

$$\boldsymbol{\Psi}_{j+1,j}^x = \bar{E}_j, \quad \forall j \in [N], \quad \forall k \in [j+1, N-1] \quad (43c)$$

From [10], (43) can be solved via the parallel Ricatti recursions,

$$\begin{aligned} \mathcal{G}_{N,j} &= \mathcal{Q}_{N,j}^x, & \mathcal{K}_{k,j} &= -\mathcal{G}_{k,j} \mathcal{B}_{k,j}, \\ \mathcal{P}_{k,j} &= \mathcal{Q}_{k,j}^x + (A_k)^\top \mathcal{P}_{k+1,j} A_k + (\mathcal{K}_{k,j})^\top \mathcal{B}_{k,j}, \\ \mathcal{G}_{k,j} &= (\mathcal{Q}_{k,j}^u + (B_k)^\top \mathcal{P}_{k+1,j} B_k)^{-1}, \\ \mathcal{B}_{k,j} &= \mathcal{Q}_{k,j}^{ux} + (B_k)^\top \mathcal{P}_{k+1,j} A_k, \end{aligned} \quad (44)$$

and parallel forward propagations

$$\begin{aligned} \boldsymbol{\Psi}_{j+1,j}^x &= \bar{E}_j, & \boldsymbol{\Psi}_{k,j}^u &= \mathcal{K}_{k,j} \boldsymbol{\Psi}_{k,j}^x, \\ \boldsymbol{\Psi}_{k+1,j}^x &= (A_k + B_k \mathcal{K}_{k,j}) \boldsymbol{\Psi}_{k,j}^x. \end{aligned} \quad (45)$$

Then, since $\bar{E}_j$ is right-invertible and (43) represents the system level constraint (SLC) assumed in Lemma 1, the controller gains $\boldsymbol{\Psi}$ obey,

$$\begin{aligned} \boldsymbol{\Psi}_{j+1,j}^x &= \bar{E}_j, & \boldsymbol{\Psi}_{k,j}^u \bar{E}_j^\dagger &= \mathcal{K}_{k,j} \boldsymbol{\Psi}_{k,j}^x \bar{E}_j^\dagger \\ \boldsymbol{\Psi}_{k+1,j}^x \bar{E}_j^\dagger &= (A_k + B_k \mathcal{K}_{k,j}) \boldsymbol{\Psi}_{k,j}^x \bar{E}_j^\dagger. \end{aligned} \quad (46)$$

Then, by unraveling the recursion (46), we have for $k > j$,

$$\boldsymbol{\Psi}_{k,j} \bar{E}_j^\dagger = \begin{bmatrix} I_{n_x} \\ \mathcal{K}_{k,j} \end{bmatrix} \left(\prod_{m=j+1}^{k-1} (A_m + B_m \mathcal{K}_{m,j}) \right) \boldsymbol{\Psi}_{j+1,j}^x \bar{E}_j^\dagger, \quad (47)$$

while the remaining gains ($\boldsymbol{\Psi}_{k,j}$ where $k \leq j$) are zero, where we define the ordered matrix product $\prod_{i=1}^n R_i := R_n R_{n-1} \cdots R_1$.

To recover the optimal gains for (33), we use the right-invertibility property of $\bar{E}_j$ and substitute $\boldsymbol{\Psi}_{k,j} \bar{E}_j^\dagger = \tilde{\boldsymbol{\Phi}}_{k,j}^x$ into (47):

$$\begin{aligned} \tilde{\boldsymbol{\Phi}}_{k,j}^x &= \begin{bmatrix} I_{n_x} \\ \mathcal{K}_{k,j} \end{bmatrix} \left(\prod_{m=j+1}^{k-1} (A_m + B_m \mathcal{K}_{m,j}) \right) \tilde{\boldsymbol{\Phi}}_{j+1,j}^x \\ &= \begin{bmatrix} I_{n_x} \\ \mathcal{K}_{k,j} \end{bmatrix} \left(\prod_{m=j+1}^{k-1} (A_m + B_m \mathcal{K}_{m,j}) \right) I_{n_x}. \end{aligned} \quad (48)$$

The second equality stems from $\tilde{\boldsymbol{\Phi}}_{j+1,j}^x$ satisfying (33c). Now, observe (34) has the same Ricatti recursions as (44),

thus resulting in an optimal gain, similarly structured to (47) where $E_j^\dagger$ is replaced with I_{n_x} , $\hat{\Phi}$:

$$\hat{\Phi}_{k,j} = \begin{bmatrix} I_{n_x} \\ \mathcal{K}_{k,j} \end{bmatrix} \left(\prod_{m=j+1}^{k-1} (A_m + B_m \mathcal{K}_{m,j}) \right) \hat{\Phi}_{j+1,j}^\times, \quad (49)$$

where $\hat{\Phi}_{j+1,j}^\times = I_{n_x}$, as constrained by (34c). Hence $\forall k, j \in [N]$ $\hat{\Phi}_{k,j} = \hat{\Phi}_{k,j}$, so $\hat{\Phi} = \hat{\Phi}$. ■

Lemma 1: Let, $\bar{E} := \text{blk_diag}(I_{n_x}, \bar{E}_0, \dots, \bar{E}_{n-1})$, where for all $i \in [N]$, $E_i \in \mathbb{R}^{n_x \times n_w^{(i)}}$ is right-invertible and $n_w^{(i)} \geq n_x$. Enforcing the uncertain LTV dynamics

$$x_{k+1} = A_k x_k + B_k u_k + \bar{E}_k w_k, \quad w_k \in \mathcal{E}_{n_w^{(k)}}, \quad (50)$$

is equivalent to enforcing the system level constraint (51), i.e., (50) $\iff$ (51),

$$\begin{bmatrix} (I_{Nn_x} - ZA) & -ZB \end{bmatrix} \begin{bmatrix} \Phi^\times \\ \Phi^u \end{bmatrix} = \bar{E}. \quad (51)$$

Furthermore, the controller gain mapping for the SLC in (51) is,

$$\mathcal{K} = (\Phi^u E^\dagger)(\Phi^\times \bar{E}^\dagger)^{-1} = (\Phi^u \bar{E}^\top)(\Phi^\times \bar{E}^\top)^{-1}, \quad (52)$$

where $\bar{E}^\dagger = \text{blk_diag}(I, \bar{E}_0^\dagger, \dots, \bar{E}_{n-1}^\dagger)$. Lastly, given $\mathcal{K}$, the forward propagation of disturbance gains obeys:

$$\begin{aligned} \Phi_{j+1,j}^\times &= \bar{E}_j, & \Phi_{k,j}^u \bar{E}_j^\dagger &= \mathcal{K}_{k,j} \Phi_{k,j}^\times \bar{E}_j^\dagger \\ \Phi_{k+1,j}^\times \bar{E}_j^\dagger &= (A_k + B_k \mathcal{K}_{k,j}) \Phi_{k,j}^\times \bar{E}_j^\dagger. \end{aligned} \quad (53)$$

Proof: First, let $\delta_k = E_k w_k$ be such that

$$x_{k+1} = A_k x_k + B_k u_k + \delta_k. \quad (54)$$

Now, let $\mathbf{x} = [x_0^\top, \dots, x_{n-1}^\top]^\top$, $\mathbf{u} = [u_0^\top, \dots, u_{n-1}^\top]^\top$, and $\boldsymbol{\delta} = [\delta_0^\top, \dots, \delta_{n-1}^\top]^\top$. Then, defining the gain mappings $\hat{\Phi}^\times$ and $\hat{\Phi}^u$ such that,

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \hat{\Phi}^\times \\ \hat{\Phi}^u \end{bmatrix} \boldsymbol{\delta}, \quad (55)$$

we have from [8]:

$$\begin{bmatrix} (I_{Nn_x} - ZA) & -ZB \end{bmatrix} \begin{bmatrix} \hat{\Phi}^\times \\ \hat{\Phi}^u \end{bmatrix} = I_{Nn_x} \quad (56)$$

Defining $\mathbf{w} = [w_0^\top, \dots, w_{n-1}^\top]^\top$, we observe:

$$\boldsymbol{\delta} = \bar{E} \mathbf{w} \quad (57)$$

Then, the gain mapping from $\mathbf{w}$ to $\mathbf{x}$ and $\mathbf{u}$ via $\Phi^\times$ and Φ^u follows from (55) and (57):

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \Phi^\times \\ \Phi^u \end{bmatrix} \mathbf{w} = \begin{bmatrix} \hat{\Phi}^\times \bar{E} \\ \hat{\Phi}^u \bar{E} \end{bmatrix} \mathbf{w} = \begin{bmatrix} \hat{\Phi}^\times \\ \hat{\Phi}^u \end{bmatrix} \bar{E} \mathbf{w}. \quad (58)$$

This establishes that $\Phi^\times = \hat{\Phi}^\times \bar{E}$ and $\Phi^u = \hat{\Phi}^u \bar{E}$. Hence, by multiplying (56) by $\bar{E}$, we arrive at (51). Thus we have shown (51) $\implies$ (50). To show the equivalence between (50) and (51), we can show (50) $\implies$ (51), by directly recovering the matrices $A_k, B_k, \bar{E}_k \forall k \in [N]$ from (51) to produce (50).

To prove (52), we observe that, $K = \hat{\Phi}^u (\hat{\Phi}^\times)^{-1}$, [8]. Then using the substitutions $\Phi^\times \bar{E}^\dagger = \hat{\Phi}^\times$ and $\Phi^u \bar{E}^\dagger =$

$\hat{\Phi}^u$ from (58) we arrive at the first equality in (52). The second equality arises from algebraic manipulation using the definition $\bar{E}^\dagger = \bar{E}^\top (\bar{E} \bar{E}^\top)^{-1}$. Lastly, manipulating (52) to see $\mathcal{K}(\Phi^\times \bar{E}^\dagger) = (\Phi^u \bar{E}^\dagger)^{-1}$, and expanding the system level constraint in (51) we have that

$$\begin{aligned} \Phi_{j+1,j}^\times &= \bar{E}_j, & \Phi_{k,j}^u \bar{E}_j^\dagger &= \mathcal{K}_{k,j} \Phi_{k,j}^\times \bar{E}_j^\dagger \\ \Phi_{k+1,j}^\times \bar{E}_j^\dagger &= (A_k + B_k \mathcal{K}_{k,j}) \Phi_{k,j}^\times \bar{E}_j^\dagger. \end{aligned} \quad (59)$$

■

APPENDIX IV PROOFS

Theorem 5: Let $\mathbf{z}$, $\mathbf{v}$, and Φ be a feasible solution for (24). Then, (26) is guaranteed to overapproximate the true closed-loop reachable set (3), i.e., $\Omega_k \subseteq \bar{\Omega}_k$ for all $k \in [N]$.

Proof: We will first prove (26) is correct assuming valid disturbances. Let $\mathcal{W}_k$ specify the disturbance $\forall k \in [N]$. Then, from SLS [4], we have that the overapproximation of the reachable set is,

$$\begin{aligned} \bar{\Omega}_k^x &:= z_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^\times \mathcal{W}_j, \\ \bar{\Omega}_k^u &:= v_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u \mathcal{W}_j. \end{aligned} \quad (60)$$

Now, suppose $\mathcal{W}_k$ can be expressed as a zonotope, $\bar{c}_k \oplus \bar{E}_k \mathcal{E}_{2n_x}$. Making this substitution in (60) we arrive at,

$$\begin{aligned} \bar{\Omega}_k^x &:= z_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^\times (\bar{c}_j \oplus \bar{E}_j \mathcal{E}_{2n_x}), \\ \bar{\Omega}_k^u &:= v_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u (\bar{c}_j \oplus \bar{E}_j \mathcal{E}_{2n_x}). \end{aligned} \quad (61)$$

Using the associativity and commutativity of the Minkowski sum (61) can be rewritten into,

$$\begin{aligned} \bar{\Omega}_k^x &:= z_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^\times \bar{c}_j \oplus \bigoplus_{j=0}^{k-1} \Phi_{k,j}^\times \bar{E}_j \mathcal{E}_{2n_x}, \\ \bar{\Omega}_k^u &:= v_k + \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u \bar{c}_j \oplus \bigoplus_{j=0}^{k-1} \Phi_{k,j}^u \bar{E}_j \mathcal{E}_{2n_x}. \end{aligned} \quad (62)$$

Then recalling that $\bigoplus_{j=0}^{k-1} \Phi_{k,j}^\times \bar{c}_j$ is a Minkowski sum of singleton sets, we can replace it with $\sum_{j=0}^{k-1} \Phi_{k,j}^\times \bar{c}_j$ (and similarly do this for $\bigoplus_{j=0}^{k-1} \Phi_{k,j}^u \bar{c}_j$), and arrive at at the disturbance reachable set (26). Now, we will show the disturbance bounds, $\mathcal{Z}_r(z_j, v_j, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)$, $\mathcal{Z}_c(z_j, v_j, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)$, in (24d)-(24e) are valid overapproximations $\mathcal{E}_k$ (9) and certify that the closed-loop reachable sets $\bar{\Omega}_k^x$ and $\bar{\Omega}_k^u$ overapproximate the true reachable set Ω_k^x and Ω_k^u (3).

Base Case: The disturbance at timestep $k = 0$ is precisely $E(z_0) = E(\bar{x}_0)$, which is captured by (25), and thus (24d)-(24e) provides overapproximate disturbance bounds.

Inductive Step: Suppose that for all timesteps $j \in [k]$, $d_j(x_j, u_j, w_j) \in E_j \mathcal{E}_{n_x} \oplus \bar{R}_j \mathcal{E}_{n_x} \oplus \bar{c}_j$. Then, we have a overapproximate reachable sets, $\Omega_k^x \subseteq \bar{\Omega}_k^x$ and $\Omega_k^u \subseteq \bar{\Omega}_k^u$, at timestep k . Define $\bar{\Omega}_k := \{[x^\top, u^\top]^\top \mid x \in \Omega_k^x, u \in \Omega_k^u\}$. Then observe that $\bar{\Omega}_k \subseteq \mathcal{A}_k := ([z_k^\top, v_k^\top]^\top + \boldsymbol{\mu}_k) \oplus \text{diag}(\boldsymbol{\sigma}_k) \mathcal{E}_{n_x + n_u}$, as defined in (25a). Then, by construction, for all $[x_k^\top, u_k^\top]^\top \in \bar{\Omega}_k$ and for all $w_k \in \mathcal{E}_{n_x}$, we have that $r(x_k, u_k, w_k) \in \bar{c}_k \oplus \bar{R}_k \mathcal{E}_{n_x}$, where $(\bar{c}_k, \bar{R}_k)$ is computed using $\mathcal{A}_k$ in (25). Similarly, the nominal disturbance $E_k w_k \in E(z_k) \mathcal{E}_{n_x}$. Thus, the total disturbance (8a), $r_k^{\text{lin}} + r_k^{\text{dist}} + E_k w_k = d(x_k, u_k, w_k) \in \bar{c}_k \oplus \bar{R}_k \mathcal{E}_{n_x} \oplus E(z_k) \mathcal{E}_{n_x}$.

Concatenating the two generator matrices produces (25b). Thus, the disturbance bound at timestep k in (24d)-(24e) is an overapproximation and contains the true residual error, i.e., $d_k(x_k, u_k, w_k)$ (8a).

Having established valid disturbance bounds for all steps $k \in [N]$, SLS guarantees that the over approximation of the closed-loop reachable set (26), $\bar{\Omega}_k^x$ and $\bar{\Omega}_k^u$, are valid, i.e., $\Omega_k^x \subset \bar{\Omega}_k^x$ and $\Omega_k^u \subset \bar{\Omega}_k^u$ for all $k \in [N]$. ■

APPENDIX V GPUSLS DUALS τ

As discussed in Sec. V-A, the RNOCP problem is solved by iteratively alternating between solving a nominal trajectory optimization and a controller update. Since these two optimizations are solved separately, we use the dual variable τ to enforce consistency between the two optimizations as discussed in [10, App. B]. Due to our reformulation, we modify the procedure as follows. As with [10, App. B], we define the auxiliary terms $\beta_{k,j} \in \mathbb{R}^{n_c}$ and $\beta_{N,j} \in \mathbb{R}^{n_f}$, where

$$\beta_{k,j} = [G_k \Phi_{k,j} \bar{c}_j + \|G_k \Phi_{k,j} \bar{E}_j\|_{1,r}]^{\circ 2} \quad (63a)$$

$$\forall j \in [N], \quad \forall k \in [j, N] \quad (63b)$$

$$\beta_{N,j} = [G_N \Phi_{N,j}^x \bar{c}_j + \|G_N \Phi_{N,j}^x\|_{1,r}]^{\circ 2} \quad (63c)$$

$$\forall j \in [N] \quad (63d)$$

where $[\cdot]^{\circ 2}$ denotes element-wise square. We can then calculate our dual variables τ by

$$\tau_{k,j} = \frac{\bar{\lambda}_k}{\sqrt{\beta_{k,j}} + \epsilon} \quad \forall j \in [N], \quad \forall k \in [j, N] \quad (64)$$

where we denote $\bar{\lambda}_k$ as the dual variable of the nominal trajectory optimization (corresponding to the dual variable λ_k in [10]), introduced here to avoid notation conflict. Finally, we can define our cost terms as

$$\begin{aligned} \mathcal{Q}_{k,j} &= \left(\text{diag}(\sqrt{\tau_{k,j}}) G_k, \begin{bmatrix} \tilde{Q}^{1/2} & 0 \\ 0 & \tilde{R}^{1/2} \end{bmatrix} \right), \\ \mathcal{Q}_{N,j} &= \left(\text{diag}(\sqrt{\tau_{N,j}}) G_N, \tilde{Q}_N^{1/2} \right). \end{aligned} \quad (65)$$

Following [9] and [10], this formulation preserves dual consistency and yields an equivalent reformulation of the RNOCP under the proposed decomposition.