Safe Output Feedback Motion Planning from Images via Learned Perception Modules and Contraction Theory

Glen Chou, Necmiye Ozay, and Dmitry Berenson

Dept. of Electrical Engineering and Computer Science,
University of Michigan, Ann Arbor, MI 48105, USA
{gchou, necmiye, dmitryb}@umich.edu

Abstract. We present a motion planning algorithm for a class of uncertain control-affine nonlinear systems which guarantees runtime safety and goal reachability when using high-dimensional sensor measurements (e.g., RGB-D images) and a learned perception module in the feedback control loop. First, given a dataset of states and observations, we train a perception system that seeks to invert a subset of the state from an observation, and estimate an upper bound on the perception error which is valid with high probability in a trusted domain near the data. Next, we use contraction theory to design a stabilizing state feedback controller and a convergent dynamic state observer which uses the learned perception system to update its state estimate. We derive a bound on the trajectory tracking error when this controller is subjected to errors in the dynamics and incorrect state estimates. Finally, we integrate this bound into a sampling-based motion planner, guiding it to return trajectories that can be safely tracked at runtime using sensor data. We demonstrate our approach in simulation on a 4D car, a 6D planar quadrotor, and a 17D manipulation task with RGB(-D) sensor measurements, demonstrating that our method safely and reliably steers the system to the goal, while baselines that fail to consider the trusted domain or state estimation errors can be unsafe.

Keywords: Motion planning · machine learning · perception-based control.

1 Introduction

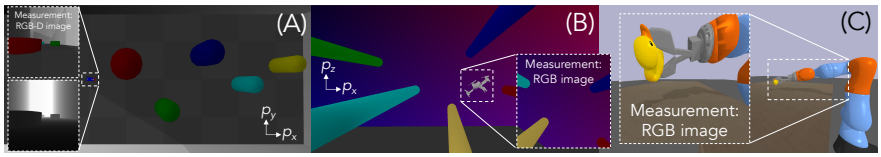

Fig. 1. For a 4D car, a 6D quadrotor, and a 14D arm, we compute plans that can be safely stabilized to reach goals at runtime using rich sensor observations in the form of RGB(-D) images.

Safely and reliably deploying an autonomous robot requires a systematic analysis of the uncertainties that it may face across its perception, planning, and feedback control modules. State-of-the-art methods largely analyze each module separately; e.g., by first certifying perception [29], finding a safe plan under a nominal dynamics model [15], and then using a stable tracking controller [22]. However, this ignores how the errors in each module can propagate. Inaccuracies in the dynamics and perception can destabilize the downstream feedback controller and lead to failure, revealing a need to unify perception, planning, and control to guarantee safety for the end-to-end autonomy pipeline.

To address this gap, we consider one such unified approach: the Output Feedback Motion Planning problem (OFMP) [21], which jointly plans nominal trajectories and designs feedback controllers which safely stabilize the system to some goal when using imperfect state information (i.e., output feedback). A concrete way to solve the OFMP

is to bound the set of states that the system may reach while tracking a plan using output feedback, that is, a closed-loop output feedback trajectory tracking tube, and ensure it is collision-free. Practical robots present challenges in solving the OFMP:

1. The tracking tubes should be efficiently computable for arbitrary trajectories so that they can be used in the planning loop to restrict the set of states that can be safely visited. However, solving this reachability problem is computationally demanding.
2. Processing rich sensor data (e.g., images, depth maps, etc.) at runtime is often done via deep learning-based perception modules, which are powerful but error-prone. Bounding this error and bounding its effect on trajectory tracking error is difficult.

To address the first challenge, we use contraction theory, which is of specific interest for the OFMP as it enables the 1) design of stabilizing feedback controllers [18] and convergent state estimators [7] and 2) fast computation of tracking/estimation tubes, given a bound on the disturbances that the controller and observer are subjected to [17]. Estimating this bound is central to our solution of the second challenge, where we use data to 1) estimate a bound on the error of a learned perception module which is valid with high probability and 2) bound the level to which incorrect state estimates can destabilize the controller. Combining these solutions provides accurate tubes that can be used in planning. In summary, we develop a contraction-based output feedback motion planning algorithm for control-affine systems stabilized from image observations, which retains guarantees on safety and goal reachability. Our specific contributions are:

- A learning-based framework for integrating high-dimensional observations into contraction-based control and estimation that can generalize across environments
- A trajectory tracking error bound for contraction-based feedback controllers in output feedback, subjected to a disturbance that accurately reflects the perception error
- A sampling-based planner which solves the OFMP, returning plans that can be safely tracked and that reliably reach the goal at runtime using image observations
- Validation in simulation on a 4D nonholonomic car, a 6D planar quadrotor, and a 17D manipulation task, guaranteeing safety whereas baseline approaches fail

2 Related Work

First, our work is related to and draws from contraction-based control and estimation. Contraction-based robust control [5, 22, 25] can ensure safety for uncertain systems using perfect state feedback, but the guarantees are lost if using imperfect state estimates. Other work has studied contraction-based convergence guarantees for state estimation without control input, e.g., [3, 7, 26]; however, solving the OFMP requires jointly analyzing the controller and observer. Most closely related is [17], which studies output feedback control via contracting controllers and observers; however, it considers simple measurement models and does not derive the tracking tubes needed for the OFMP.

Our work also relates to control from rich observations. Differentiable filtering [12] learns state estimators from images in an end-to-end fashion, which while empirically successful, do not provide guarantees. Other work focuses on safety: [9] safely controls linear systems using learned observation maps; other methods use Control Barrier/Lyapunov Functions (CBF/CLFs) to guarantee safety for nonlinear systems by robustifying the CBF condition to measurement errors [6, 10]; however, these methods use simple sensor models or require that the entire state is invertible from one observation, precluding their use on states that must be estimated over time, e.g., velocities. In contrast,

our method only seeks to invert a *subset* of the state, which is then used in a dynamic observer to estimate the unobserved states. Other work [8] combines CLFs and CBFs to safely reach goals from observations, but focuses on simpler LiDAR sensor models.

Finally, our work relates to planning under uncertainty. Funnel-based methods buffer motion primitives with tracking tubes under perfect [16] and vision-based [27] feedback control. In contrast, we do not rely on precomputed primitives, and can plan novel trajectories. Other methods [1, 2] consider measurement error in planning but are either restricted to linear systems or simple sensor models. These methods are instances of the generally intractable belief-space planning problem; solving this problem requires simplifying assumptions [24] that may compromise safety. We do not solve the full belief-space planning problem; instead of tracking belief distributions, our set-based approach bounds the reachable states and state estimates under the worst-case error.

3 Preliminaries and Problem Statement

We consider uncertain continuous-time control-affine nonlinear systems (which include many common mechanical systems of interest [15]) with output observations

$$\dot{x}(t) = f(x(t)) + Bu(t) + B_w(t)w_x(t) \tag{1a}$$
$$y(t) = h(x(t), \theta) + B_y w_y(t) \tag{1b}$$

where $f : \mathcal{X} \to \mathcal{X}$, $\mathcal{X} \subseteq \mathbb{R}^{n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $B_w : [0, \infty) \to \mathbb{R}^{n_x \times n_{w_x}}$, $B_y \in \mathbb{R}^{n_y \times n_{w_y}}$, $\mathcal{U} \subseteq \mathbb{R}^{n_u}$, and $w_x \in \mathbb{R}^{n_{w_x}}$ is a possibly stochastic state disturbance where $\|w_x(t)\| \leq \bar{w}_x$, for all t. Without loss of generality, we assume $\|B_w(t)\| \leq 1$, for all t. Norms $\|\cdot\|$ without subscript are the (induced) 2-norm. We obtain high-dimensional observations $y \in \mathcal{Y} \subseteq \mathbb{R}^{n_y}$ (e.g., $N \times N$-pixel RGB-D images, leading to $n_y = 4N^2$), generated by a deterministic, nonlinear function $h(x, \theta) : \mathcal{X} \times \Theta \to \mathcal{Y}$ which is unknown to the robot; here, $\theta \in \mathbb{R}^{n_p}$ are external parameters (e.g., location of obstacles, map of environment, etc.). The observations may be corrupted by (possibly stochastic) sensor noise $w_y(t) \in \mathbb{R}^{n_{w_y}}$, where $\|w_y(t)\| \leq \bar{w}_y$, for all t. We note that our results also apply to time-varying $B(t)$ under some conditions on its null-space.

We assume that (1a) is locally incrementally exponentially stabilizable (IES) in domain $D_c \subseteq \mathcal{X}$, that is, there exists an α, $\lambda > 0$, and some feedback controller such that for any nominal trajectory $x^*(t) \subseteq D_c$, $\|x^*(t) - x(t)\| \leq \alpha e^{-\lambda t}\|x^*(0) - x(0)\|$ for all t. While stronger than asymptotic stability, many underactuated systems are IES [19]. We also assume that (1) is locally universally detectable [17], which ensures that any two trajectories $x_1(t)$ and $x_2(t)$ in a domain $D_e \subseteq \mathcal{X}$ that yield identical observations $y_1(t)$ and $y_2(t)$ for all t converge to each other as $t \to \infty$, i.e., $x_1(t) \to x_2(t)$. Similar assumptions are common in the estimation literature [20] to ensure estimator convergence, and do not require the full state to be observable instantaneously, e.g., as in [10].
Definitions: We assume $\mathcal{X}$ is partitioned into (un)safe ($\mathcal{X}_{\text{unsafe}}$) $\mathcal{X}_{\text{safe}}$ sets (e.g., obstacles). Let ($\mathbb{S}_n^{>0}$) $\mathbb{S}_n$ be the set of (positive definite) symmetric $n \times n$ matrices. For $Q \in \mathbb{S}_n$, denote $\bar{\lambda}(Q)$ and $\underline{\lambda}(Q)$ as its maximum and minimum eigenvalues. If $Q(x)$ is a matrix-valued function over a domain D, we denote $\bar{\lambda}_D(Q) \doteq \sup_{x \in D} \bar{\lambda}(Q(x))$ and $\underline{\lambda}_D(Q) \doteq \inf_{x \in D} \underline{\lambda}(Q(x))$. Let the Lie derivative of a matrix-valued function $Q(x) \in \mathbb{R}^{n \times n}$ along a vector $y \in \mathbb{R}^n$ be denoted as $\partial_y Q(x) \doteq \sum_{i=1}^n y^i \frac{\partial Q}{\partial x^i}$, where x^i is the ith element of vector x. For a smooth manifold $\mathcal{X}$, a Riemannian metric tensor $M : \mathcal{X} \to \mathbb{S}_{n_x}^{>0}$ provides the tangent space $T_x\mathcal{X}$ with an inner product $\delta_x^\top M(x)\delta_x$, where $\delta_x \in T_x\mathcal{X}$. The length $l(c)$ of a curve $c : [0, 1] \to \mathcal{X}$ between $c(0)$, $c(1)$ is $l(c) \doteq$

$\int_0^1 \sqrt{V(c(s), c_s(s))} ds$, where $V(c(s), c_s(s)) \doteq c_s(s)^\top M(c(s)) c_s(s)$, and $c_s(s) \doteq \partial c(s)/\partial s$. The Riemannian distance between $p, q \in \mathcal{X}$ is $d(p,q) \doteq \inf_{c \in \mathcal{C}(p,q)} l(c)$, where $\mathcal{C}(p,q)$ contains all smooth curves between p and q; a curve $\gamma(p,q)$ achieving the argmin is called a geodesic.

3.1 Problem statement

We formally state the output feedback motion planning problem (OFMP) as follows:
OFMP: Given start x_I, external parameter $\theta \in D_\theta$, goal region $\mathcal{G} \subseteq D_x$ (D_θ, D_x are defined in the next paragraph), and safe set $\mathcal{X}_{\text{safe}}$, we want to plan a state-control trajectory $x^* : [0,T] \to \mathcal{X}$, $u^* : [0,T] \to \mathcal{U}$, $x^*(0) = x_I$, under the nominal dynamics $\dot{x}(t) = f(x(t)) + Bu(t)$ such that in execution on the true system (1a), $x(t) \in \mathcal{X}_{\text{safe}}$ for all $t \in [0,T]$ and $x(T) \in \mathcal{G}$. At runtime, we do not observe $x(t)$; we are only given observations $y(t)$ generated by (1b), and must track x^* using a (dynamic) output feedback controller that we must also design. We assume f, B, B_w, and B_y are known; h is unknown; w_x, w_y are not measurable but $\bar{w}_x$ and $\bar{w}_y$ are known. If $n_r \leq n_x$ of the states can be inferred directly from y, we denote these indices as the reduced observation $y_r = C_r x \in \mathbb{R}^{n_r}$, where $C_r \in \{0,1\}^{n_r \times n_x}$ is a boolean matrix that selects the observable dimensions of x. We assume that we are given C_r. Let $x(t)$ be the executed trajectory of (1a), and let $\hat{x}(t)$ be the trajectory of the state estimate. We are given upper bounds $\bar{d}_c(0)$, $\bar{d}_e(0)$on the Riemannian distance between the true and estimated initial state $d_e(x(0), \hat{x}(0))$ and between the true/planned initial state $d_c(x^*(0), x(0))$; $d_e(\cdot,\cdot)$ and $d_e(\cdot,\cdot)$ are defined with respect to (w.r.t.) metrics M_c and M_e, defined in Sec. 3.2.

To help solve the OFMP, we are given two datasets. The first is $\mathcal{S} = \{h(x_i, \theta_i), x_i, \theta_i\}_{i=1}^{N_{\text{data}}}$, a dataset of noiseless (cf. Sec. 6 for discussion on how to relax this assumption) observation-state-parameter triplets, where $x_i \in D_p \subseteq \mathcal{X}$, $\theta_i \in D_\theta \subseteq \Theta$ are collected by any means (sampling, demonstrations, etc.). We assume D_p and D_θ (the domains where $\mathcal{S}$ is drawn from) are known, though this can be relaxed by estimating these sets as in [5,13]. We are also given a validation dataset $\mathcal{V} = \{h(x_i, \theta_i), x_i, \theta_i\}_{i=1}^{N_{\text{val}}}$ collected i.i.d. in $D_p \times D_\theta$. In the context of (1b), $h(x,\theta)$ may be a simulated image, and $B_y w_y(t)$ is the sensor noise at runtime. We also define a "trusted domain" for planning, $D = D_x \times D_\theta \subseteq \mathcal{X} \times \Theta$, where $D_x = D_r \cap D_c \cap D_e$ and D_r is defined as follows: for ease, suppose C_r selects the first n_r indices of x, then $D_r = (C_r D_p) \times \mathbb{R}^{n_x - n_r}$. D_r is defined similarly if C_r selects other indices (cf. Fig. 9). Ultimately, D_x is a set where a stabilizing controller (in D_c) and state estimator (in D_e) exist, and where the perception is valid (in D_r).

3.2 Control/observer contraction metrics (CCMs/OCMs)

As our approach builds on contraction theory, we provide an overview here. Control contraction theory [18] studies incremental stabilizability by measuring the distances between trajectories w.r.t. a Riemannian metric $M_c : \mathcal{X} \to \mathbb{S}^{>0}_{n_x}$. For (1a) if $w_x \equiv 0$, a sufficient condition [22] for M_c to be called a control contraction metric (CCM) is:

$$B_\perp^\top \Big(-\partial_f W_c(x) + A(x) W_c(x) + W_c(x) A(x)^\top + 2\lambda_c W_c(x) \Big) B_\perp \preceq 0 \tag{2a}$$

$$B_\perp^\top \Big(\partial_{B^j} W_c(x) \Big) B_\perp = 0, \quad j = 1 \ldots n_u, \tag{2b}$$

for all $x \in D_c$, where $W_c(x) \doteq M_c^{-1}(x)$, $A(x) = \frac{\partial f(x)}{\partial x}$, and $B_\perp$ is a basis for the null-space of B. The CCM also defines a controller $u : \mathcal{X} \times \mathcal{X} \times \mathcal{U} \to \mathcal{U}$, which takes the current state $x(t)$ and a state/control $x^*(t)$, $u^*(t)$ on the nominal state/control trajectory being tracked $x^* : [0,T] \to \mathcal{X}$, $u^* : [0,T] \to \mathcal{U}$, and returns a u that contracts x towards x^* at rate $\lambda_c > 0$. The controller $u(x, x^*, u^*)$ can be computed directly via

$W_c(x)$ (cf. Sec. 4.2). If $w_x \equiv 0$, for any nominal $x^*(t)$, applying $u(x, x^*, u^*)$ renders the system closed-loop IES, i.e., $\|x(t) - x^*(t)\| \leq \alpha_c \|x(0) - x^*(0)\| e^{-\lambda_c t}$ for $\alpha_c > 0$. For bounded w_x, (1a) remains in a tube around $x^*(t)$; we exploit this in Sec. 4.2. Contraction also analyzes the convergence of state observers [7,17], i.e., whether a state estimate $\hat{x}(t)$ approaches the true state $x(t)$. Consider the nominal closed-loop system $\dot{x} = f(x) + Bu(\hat{x}, x^*, u^*)$ with noiseless observations $y = h(x, \theta)$ and a nominal observer

$$\dot{\hat{x}} = f(\hat{x}) + Bu(\hat{x}, x^*, u^*) + \tfrac{1}{2}\rho(\hat{x}) M_e(\hat{x}) C(\hat{x})^\top (y - h(\hat{x}, \theta)) \tag{3}$$

for the nominal system, where $C(x) = \frac{\partial h(x,\theta)}{\partial x}$, $\rho(x) \geq 0$ is a multiplier term, and $M_e : \mathcal{X} \to \mathbb{S}^{>0}_{n_x}$ is called an observer contraction metric (OCM), which should satisfy

$$\partial_{f+Bu} W_e(\hat{x}) + W_e(\hat{x}) A(\hat{x}) + A(\hat{x})^\top W_e(\hat{x}) - \rho(\hat{x}) C(\hat{x})^\top C(\hat{x}) \leq -2\lambda_e W_e(\hat{x}) \tag{4}$$

for all $\hat{x} \in D_e \subseteq \mathcal{X}, u \in \mathcal{U}$. Here, $W_e(\hat{x}) = M_e^{-1}(\hat{x})$. To show that the estimated and true trajectories $\hat{x}(t)$ and $x(t)$ converge, we can analyze a nominal "meta-level" virtual system with state q [26], which recovers the nominal $x(t)$ and $\hat{x}(t)$ when integrated from initial conditions $q(0) = x(0)$ and $q(0) = \hat{x}(0)$:

$$\dot{q} = f(q) + Bu(\hat{x}, x^*, u^*) + \tfrac{1}{2}\rho(\hat{x}) M_e(\hat{x}) C(\hat{x})^\top (y - h(q, \theta)). \tag{5}$$

By setting $q = \hat{x}$, we recover the estimator dynamics (3); if we set $q = x$, we recover $\dot{x} = f(x) + Bu(\hat{x}, x^*, u^*)$. We can then analyze the convergence of $\hat{x}$ to x via (5), and [26] shows that if (4) holds, then $\hat{x}(t)$ contracts at some rate $\gamma \in (0, \lambda_e]$ towards $x(t)$. If $M_e(x)$ and $C(x)$ are constant, one can show that this holds for $\gamma = \lambda_e$. In particular, $\|x(t) - \hat{x}(t)\| \leq \alpha_e \|x(0) - \hat{x}(0)\| e^{-\lambda_e t}$ for $\alpha_e > 0$, and $\hat{x}(t)$ remains in a tube around $x(t)$ if (3) is perturbed. For polynomial systems of moderate dimension ($n_x \lesssim 12$) with polynomial observation maps, CCMs and OCMs can be found via convex Sum of Squares (SoS) programs [22]. CCMs/OCMs can also be found for high-dimensional non-polynomial systems via learning-based methods (e.g., [5,23]).

4 Method

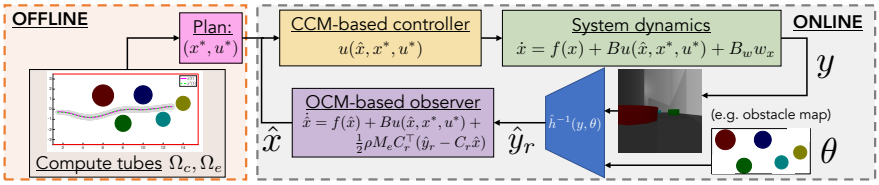

Fig. 2. Our method. **Offline**: After learning a perception system $\hat{h}^{-1}$ (Sec. 4.1), we bound its error to derive tracking tubes under imperfect perception (Sec. 4.2). We use these tubes to find safely-trackable plans (Sec. 4.4). **Online**: We design a CCM/OCM-based controller/observer (Sec. 4.3) to track the plan/perform state estimation at runtime, using $\hat{h}^{-1}$ to process rich observations y.

We describe our solution to the OFMP (cf. Fig. 2). Using dataset $\mathcal{S}$, we first train a perception system that returns a reduced-order observation that simplifies the search for the contraction metrics (Sec. 4.1). Second, we bound the error of the learned perception module, and propagate this perception error bound through the system to derive bounds on the tracking and estimation error when using a CCM-/OCM-based controller/estimator (Sec. 4.2). Third, we obtain a CCM and OCM which optimizes this bound via SoS programming (Sec. 4.3). Finally, we use these bounds to constrain a planner to return trajectories that enable safe runtime tracking and robust goal reachability from observations (Sec. 4.4). For space, all proofs for the theoretical results are in App. C.

4.1 Learning a perception module for contraction-based estimation

Let us reconsider the observer (3), which updates its estimate directly using $y - h(\hat{x}, \theta)$ in the rich observation space. To implement (3), one can use $\mathcal{S}$ to train a deep approximation of h, denoted $\hat{h}$, design an OCM satisfying (4) for $C(\hat{x}) = \frac{\partial \hat{h}(\hat{x},\theta)}{\partial x}$, and plug $\hat{h}$ and the OCM into (3). This naïve solution is flawed: 1) as n_y is large, learning an accurate $\hat{h}$ can be difficult; 2) the $C(\hat{x})$ in (4) becomes the Jacobian of a (non-polynomial) deep network, complicating OCM synthesis by precluding the use of SoS programming.

We can take a more structured approach if we know which states can be directly inferred from y; this is reasonable if the states have semantic meaning (e.g., poses, velocities). Recall C_r (Sec. 3.1) defines this reduced observation as $y_r = C_r x \in \mathbb{R}^{n_r}$. We can then learn an approximate inverse $\hat{h}^{-1}(y, \theta) : \mathbb{R}^{n_y} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_r}$ which maps a y and θ to the reduced observation. Note that if each unique y corresponds to a unique y_r, this inverse is well-defined and does not require the full state to be invertible from a single y. Concretely, consider a car with position, orientation, and velocity states $[p_x, p_y, \phi, v]$ and RGB-D data from an onboard camera (Fig. 1.A) driving in several obstacle fields. In this case, $y_r = [p_x, p_y, \phi]^\top$ and θ could be the obstacle locations. We model $\hat{h}^{-1}$ as a neural network and train it via the mean squared error between $\hat{h}^{-1}(y_i, \theta_i)$ and $C_r x_i$ for all $i \in 1, \ldots, N_{\text{data}}$. Note that as the nominal reduced observations are roughly linear, i.e., $\hat{h}^{-1}(h(x, \theta), \theta) \approx C_r x$, this simplifies the nominal observer (3) to $\dot{\hat{x}} = f(\hat{x}) + Bu(\hat{x}, x^*, u^*) + \frac{1}{2}\rho(\hat{x})M_e(\hat{x})C_r^\top C_r(x - \hat{x})$, and simplifies OCM synthesis: as $C_r^\top C_r$ is constant, (4) is SoS-representable, despite $\hat{h}^{-1}$ being non-polynomial. Compared to the nominal reduced observer, the true observer we use,

$$\dot{\hat{x}} = f(\hat{x}) + Bu(\hat{x}, x^*, u^*) + \tfrac{1}{2}\rho(\hat{x})M_e(\hat{x})C_r^\top(\hat{h}^{-1}(h(x, \theta) + B_y w_y, \theta) - C_r\hat{x}), \quad (6)$$

experiences disturbance from model error $B_w w_x$, sensor noise $B_y w_y$, and learning error $\|\hat{h}^{-1}(h(x, \theta), \theta) - C_r x\|$. Quantifying these errors for our vision-based observer (6) is one of our core contributions and is key in deriving tracking bounds useful for planning.

4.2 Bounding tracking error and state estimation error for planning

To begin, assume we have a CCM M_c and an OCM M_e that are valid in $D_c \subseteq \mathcal{X}$ and $D_e \subseteq \mathcal{X}$ and which contract at rate λ_c and λ_e, respectively. We discuss CCM/OCM synthesis in Sec. 4.3. Define the nominal closed-loop state and virtual dynamics as:

$$\dot{x}(t) = f(x(t)) + Bu(x(t), x^*(t), u^*(t)) \quad (7a)$$

$$\dot{q}(t) = f(q(t)) + Bu(\hat{x}(t), x^*(t), u^*(t)) + \tfrac{1}{2}\rho(q(t))M_e(q(t))C_r^\top C_r(x(t) - q(t)) \quad (7b)$$

Factor the CCM/dual OCM as $M_c(x) = R_c(x)^\top R_c(x)$ and $W_e(x) = R_e(x)^\top R_e(x)$. Let $\gamma_c^t(s)$, $s \in [0, 1]$ be the geodesic between $x^*(t)$ and $x(t)$ w.r.t. M_c, and $\gamma_e^t(s)$, $s \in [0, 1]$ be the geodesic between $\hat{x}(t)$ and $x(t)$ w.r.t. W_e. [17] shows if $\gamma_c^t(s) \subseteq D_c$ for all t, s and (7a) is perturbed by $w_c(t)$, i.e., $\dot{x} = f(x) + Bu(x, x^*, u^*) + w_c$, the Riemannian distance w.r.t. M_c between the true and nominal state, $d_c(t) = d_c(x^*(t), x(t))$, satisfies:

$$\dot{d_c}(t) \leq -\lambda_c d_c(t) + \textstyle\int_0^1 \|R_c(\gamma_c^t(s))w_c(t)\|ds. \quad (8)$$

If $\gamma_e^t(s) \subseteq D_e$ for all t, s and (7b) is perturbed by additive $w_q(t)$ [26], the Riemannian distance w.r.t. W_e between the true and estimated state, $d_e(t) = d_e(x(t), \hat{x}(t))$, satisfies

$$\dot{d_e}(t) \leq -\lambda_e d_e(t) + \textstyle\int_0^1 \|R_e(\gamma_e^t(s))w_q(t)\|ds. \quad (9)$$

We will use (8) and (9) to obtain upper bounds on the tracking/estimation Riemannian distances, denoted as $\bar{d}_c(t)$ and $\bar{d}_e(t)$, respectively. These upper bounds define tracking and state estimation tubes, i.e., a bound on where x and $\hat{x}$ can be, which we denote as $\Omega_c(t) = \{x \mid d_c(x^*(t), x) \leq \bar{d}_c(t)\}$ and $\Omega_e(t) = \{\hat{x} \mid d_e(x(t), \hat{x}) \leq \bar{d}_e(t)\}$, respectively. These tubes are crucial in informing where the planner can safely visit, since tracking any Ω_c-buffered candidate trajectory within D_x which remains in $\mathcal{X}_{\text{safe}}$ is guaranteed to remain safe. However, for these tubes to be usable in a planner, we need explicit bounds on the integral terms in (8) and (9). In this section, we first present the final derived bounds on the integrals (Lemmas 1 and 2), describe the ideas behind the derivations, and postpone the full mathematical details to App. B.

Lemma 1 ($\dot{d}_c(t)$)**.** *The integral term in* (8) *can be bounded as*

$$\int_0^1 \|R_c(\gamma_c^t(s))w_c(t)\|ds \leq \sqrt{\bar{\lambda}_{D_c}(M_c)}\bar{w}_x + L_{\Delta k}d_e. \tag{10}$$

In the second term, $L_{\Delta k}$ is the Lipschitz constant of the controller error (to be described later) which, together with state estimate error d_e, bounds the destabilizing effect of using incorrect state estimates in feedback control. This term, which can be explicitly estimated and thus concretely informs tube size in planning, is the key novelty of Lemma 1. Overall, (10) states that tracking degrades with larger dynamics and estimation error.

Lemma 2 ($\dot{d}_e(t)$)**.** *Let $\bar{\sigma}(B_y)$ denote the maximum singular value of B_y. For constant ρ and M_e, the integral in* (9) *simplifies to $\|R_e w_q(t)\|$ and can be bounded as:*

$$\|R_e w_q(t)\| \leq \sqrt{\bar{\lambda}(W_e)}\bar{w}_x + \tfrac{1}{2}\rho\bar{\lambda}(M_e)^{1/2}\big(L_{\hat{h}^{-1}}\sqrt{\bar{\sigma}(B_y)}\bar{w}_y + \bar{\epsilon}_{\{1,2,3\}}(x^*, \theta)\big) \tag{11}$$

We write Lemma 2 for constant ρ and M_e, as this is the representation used in Sec. 5. Here, $L_{\hat{h}^{-1}}$ is the local Lipschitz constant of $\hat{h}^{-1}$, and $\bar{\epsilon}_{\{1,2,3\}}(x^*, \theta)$ are (spatially-varying) bounds on its error $\|\hat{h}^{-1}(h(x, \theta), \theta) - C_r x\|$, each with different strengths/weaknesses (cf. Fig. 4 for a visual overview). Relative to prior work, Lemma 2 is novel as it bounds high-dimensional measurement error and learned perception module error. Overall, (11) states that estimation accuracy degrades with larger dynamics error, measurement error, and learned perception module error.

Bounding tracking error: We explain more details behind Lemma 1. As Lemma 1 relies on a bound for $w_c(t)$, we first break down the components that make up $w_c(t)$. Relative to the nominal closed-loop dynamics (7a), our true closed-loop system

$$\dot{x}(t) = f(x(t)) + Bu(\hat{x}(t), x^*(t), u^*(t)) + B_w(t)w_x(t) \tag{12}$$

is subject to two disturbances. The first is the dynamics error $B_w(t)w_x(t)$. The second is imperfect state feedback: we apply $u(\hat{x}, x^*, u^*)$ instead of $u(x, x^*, u^*)$, which unlike the latter, may not stabilize (7a) at rate λ_c. Naïvely, one can bound this error by rewriting (12) as $\dot{x} = f(x) + Bu(\hat{x}, x^*, u^*) - Bu(x, x^*, u^*) + Bu(x, x^*, u^*) + B_w w_x$, where the difference between output/perfect state feedback is in red. While $\|Bu(\hat{x}, x^*, u^*) - Bu(x, x^*, u^*)\|$ is a valid disturbance bound, we can obtain a tighter bound by exploiting the structure of $u(x, x^*, u^*)$. In general, many $u_{\text{fb}} \in \mathcal{U}$ can make $\dot{x} = f(x) + B(u^* + u_{\text{fb}})$ contract at rate λ_c towards x^*, w.r.t. M_c. Define $\dot{x}^* = f(x^*) + Bu^*$; then, the contracting u_{fb} [22] are defined by a linear inequality constraint,

$$\mathcal{U}_{\text{feas}}(x, x^*, u^*) = \{u_{\text{fb}} \mid \gamma_{c,s}^\top(1)M_c(x)\dot{x} - \gamma_{c,s}^\top(0)M_c(x^*)\dot{x}^* \leq -\lambda_c d_c(x^*, x)^2\}, \tag{13}$$

where $\gamma_{c,s}(\cdot) = \frac{\partial \gamma_c(\cdot)}{\partial s}$. As in [22], we select the minimum-norm feasible control to be $u(\hat{x}, x^*, u^*)$, i.e., $u(\hat{x}, x^*, u^*) = \arg\min_{u \in \mathcal{U}_{\text{feas}}(\hat{x}, x^*, u^*)} \|u\|$. Then, using $\mathcal{U}_{\text{feas}}$, we can rewrite (12) as $\dot{x} = f(x) + B(u(\hat{x}, x^*, u^*) - u_{\text{closest}} + u_{\text{closest}}) + B_w w_x$, where $u_{\text{closest}}(\hat{x}, x, x^*, u^*) \doteq \arg\min_{u \in \mathcal{U}_{\text{feas}}(x, x^*, u^*)} \|u - u(\hat{x}, x^*, u^*)\|$ is the closest control input to $u(\hat{x}, x^*, u^*)$ that contracts the nominal dynamics at x. Bounding the imperfect state feedback as $\|Bu(\hat{x}, x^*, u^*) - Bu_{\text{closest}}\|$ instead of $\|Bu(\hat{x}, x^*, u^*) - Bu(x, x^*, u^*)\|$ can be far tighter, as $u(\hat{x}, x^*, u^*)$ may still contract the system at rate λ_c (Fig. 3: $\hat{x}_2$ case), or there can be a contracting u closer to $u(\hat{x}, x^*, u^*)$ than $u(x, x^*, u^*)$ (Fig. 3: $\hat{x}_1$ case). Combining with the dynamics error, we can write w_c:

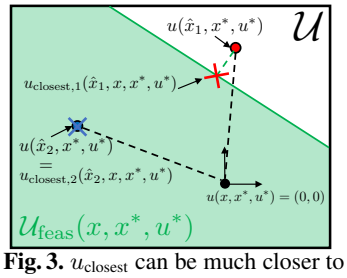

Fig. 3. u_{closest} can be much closer to $u(\hat{x}, x^*, u^*)$ than $u(x, x^*, u^*)$: we show this for two different state estimates $\hat{x}_1$ and $\hat{x}_2$.

$$w_c(t) \doteq Bu(\hat{x}(t), x^*(t), u^*(t)) - Bu_{\text{closest}}(t) + B_w(t) w_x(t) \tag{14}$$

As (14) still depends on x and $\hat{x}$, which are unknown at planning time, extra steps must be taken to obtain a useful bound that is independent of x and $\hat{x}$; we achieve this by bounding the first two terms of (14) via a Lipschitz constant. Define $\Delta k(\hat{x}, x, x^*, u^*) = \max_{s \in [0,1]} \|R_c(\gamma_c^t(s)) B(u(\hat{x}, x^*, u^*) - u_{\text{closest}})\|$, and $L_{\Delta k}$ as its local Lipschitz constant in the first argument, i.e., for all $x^* \in D$, $u^* \in \mathcal{U}$, $\{x \mid d_c(x^*, x) \le \bar{c}\}$, and $\{\hat{x} \mid d_e(x, \hat{x}) \le \bar{e}\}$ for predetermined $\bar{c}, \bar{e} > 0$ (adjustable based on the expected error),

$$|\Delta k(\hat{x}_1, x, x^*, u^*) - \Delta k(\hat{x}_2, x, x^*, u^*))| \le L_{\Delta k} d_e(\hat{x}_1, \hat{x}_2). \tag{15}$$

See Rem. 1 for details on estimating $L_{\Delta k}$. In estimating $L_{\Delta k}$, we measure input distances w.r.t. W_e; this reduces conservativeness due to the form of our estimation error bound. Combining (14)-(15) yields Lemma 1; see App. C for the detailed proof.

Bounding estimation error: Now, we provide more details behind Lemma 2. To bound $\int_0^1 \|R_e(\gamma_e^t(s)) w_q(t)\| ds$, we first note that $\|w_q\|$ is bounded by the sum of the disturbance magnitudes when $q = x$ and when $q = \hat{x}$ [26]. If $q = x$, (7b) becomes $\dot{x} = f(x) + Bu(\hat{x}, x^*, u^*)$; relative to the true closed-loop dynamics (12), the disturbance is $B_w(t) w_x(t)$. If instead $q = \hat{x}$, (7b) becomes $\dot{\hat{x}} = f(\hat{x}) + Bu(\hat{x}, x^*, u^*) + \frac{1}{2}\rho(\hat{x}) M_e(\hat{x}) C_r^\top C_r(x - \hat{x})$; relative to the true observer (6), the disturbance is

$$w_e(t) \doteq \tfrac{1}{2}\rho(\hat{x}(t)) M_e(\hat{x}(t)) C_r^\top \big(\hat{h}^{-1}\big(h(x(t), \theta) + B_y w_y(t), \theta\big) - C_r x(t)\big). \tag{16}$$

Two errors drive $w_e(t)$: the perception error $\hat{h}^{-1}(h(x, \theta), \theta) - C_r x$, and the runtime observation noise $B_y w_y$. Combining with the dynamics error gives $w_q(t) \doteq B_w(t) w_x(t) + w_e(t)$. $B_w(t) w_x(t)$ can be bounded as in Lemma 1, but $w_e(t)$ is harder to bound. Let $y_p = h(x, \theta)$ and $y = h(x, \theta) + B_y w_y$. We rewrite the norm of the red term in (16) as:

$$\begin{aligned} \|\hat{h}^{-1}(y, \theta) - C_r x\| &= \|\hat{h}^{-1}(y, \theta) - \hat{h}^{-1}(y_p, \theta) + \hat{h}^{-1}(y_p, \theta) - C_r x\| \\ &\le \underbrace{L_{\hat{h}^{-1}} \|B_y w_y\|}_{\text{from measurement noise}} + \underbrace{\|\hat{h}^{-1}(y_p, \theta) - C_r x\|}_{\text{from learning error} \doteq \epsilon(x, \theta)}. \end{aligned} \tag{17}$$

Here, $L_{\hat{h}^{-1}}$ is the local Lipschitz constant of the learned inverse function in y, i.e.,

$$\|\hat{h}^{-1}(\tilde{y}, \theta) - \hat{h}^{-1}(\check{y}, \theta)\| \le L_{\hat{h}^{-1}} \|\tilde{y} - \check{y}\|, \quad \forall \tilde{y}, \check{y} \in D_y \oplus \mathcal{Y}_d, \quad \forall \theta \in D_\theta, \tag{18}$$

where $D_y = h(D_r, D_\theta)$ is the image of the training data domains, $\oplus$ is the Minkowski sum, and $\mathcal{Y}_d = \{B_y w_y \mid \|w_y\| \le \bar{w}_y\}$ is the set of feasible measurement noise. The first braced term in (17) bounds the effect of measurement error on the reduced observation and is valid for all $(x, \theta) \in D_r \times D_\theta$ and observation noise satisfying $\|w_y\| \le \bar{w}_y$.

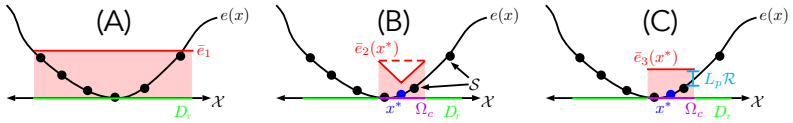

Fig. 4. Our perception error bounds. (A) $\bar{\epsilon}_1$ is simple but conservative. B) $\bar{\epsilon}_2(x^*)$ is tighter, as it only seeks to be valid over the tube Ω_c. However, it scales linearly with the size of Ω_c. C) $\bar{\epsilon}_3(x^*)$ can be tighter for larger Ω_c by adding a Lipschitz-based buffer to the largest training error in Ω_c.

Now, consider the second braced term in (17). How can we bound the learned perception module error $\epsilon(x, \theta) \doteq \|\hat{h}^{-1}\big(h(x, \theta), \theta\big) - C_r x\|$ over $D_r \times D_\theta$? We describe three options (Fig. 4) at a high level, highlight their strengths/drawbacks, and provide the details in App. B. The first bound, denoted $\bar{\epsilon}_1$, is a constant bound on $\epsilon(x, \theta)$ globally over $D_r \times D_\theta$ (Fig. 4.A). This works well if the error is consistent, but is loose if there are any error spikes. The second bound (Fig. 4.B), denoted $\bar{\epsilon}_2(x^*, \theta)$, bounds the error only in the tube Ω_c around a nominal x^*, using the Lipschitz constant of $\epsilon(x, \theta)$ (denoted L_p). Due to its locality, $\bar{\epsilon}_2(x^*, \theta)$ can be tighter than $\bar{\epsilon}_1$; however, it scales linearly with the size of Ω_c, even if $\epsilon(x, \theta)$ remains constant. The third bound, $\bar{\epsilon}_3(x^*, \theta)$ (Fig. 4.C), also bounds the error in the tube but avoids the linear scaling by taking the worst training error in Ω_c and buffering it with a *constant* value, which depends on L_p and the dataset dispersion $\mathcal{R}$. Each of these bounds $\bar{\epsilon}_{\{1,2,3\}}$ on $\epsilon(x, \theta)$ can be plugged into Lemma 2 to upper bound $\epsilon(x, \theta)$; see App. B for details.

Integrating the differential inequalities: Now that we can bound the RHSs of the differential inequalities (8) and (9) via Lemmas 1 and 2, we show how these bounds on $\dot{d}_c$ and $\dot{d}_e$ bound the values of d_c and d_e, thereby providing the desired tubes. By grouping terms in (8)-(9), we have the following affine vector-valued differential inequality,

$$\begin{bmatrix} \dot{d}_c \\ \dot{d}_e \end{bmatrix} \le \begin{bmatrix} -\lambda_c & L_{\Delta k} \\ (*) & -\lambda_e \end{bmatrix} \begin{bmatrix} d_c \\ d_e \end{bmatrix} + \begin{bmatrix} \sqrt{\bar{\lambda}_{D_c}(M_c)}\bar{w}_x \\ \sqrt{\bar{\lambda}(W_e)}\bar{w}_x + \frac{\rho}{2}\bar{\lambda}(M_e)^{1/2}\big(L_{\hat{h}^{-1}}\bar{w}_y + \bar{\epsilon}_{\{1,\tilde{2},3\}}(x^*, \theta)\big) \end{bmatrix}, \quad (19)$$

where we regroup the terms for $\bar{\epsilon}_2(x^*, \theta)$ as $\bar{\epsilon}_{\tilde{2}}(x^*, \theta) \doteq \bar{\epsilon}_2(x^*, \theta) - L_p d_c / \sqrt{\underline{\lambda}_{D_c}(M_c)}$, and $(*) = 0.5 L_p \rho \sqrt{\bar{\lambda}(M_e)/\underline{\lambda}_{D_c}(M_c)}$ if using $\bar{\epsilon}_2$ and 0 else. Then, we have this result:

Theorem 1 (From derivative to value). *Let RHS denote the right hand side of* (19). *Given bounds on the Riemannian distances at $t = 0$: $d_c(0) \le \bar{d}_c(0)$ and $d_e(0) \le \bar{d}_e(0)$, upper bounds $\bar{d}_c(t) \ge d_c(t)$ and $\bar{d}_e(t) \ge d_e(t)$ for all $t \in [0, T]$ can be written as*

$$\begin{bmatrix} d_c(t) \\ d_e(t) \end{bmatrix} \le \int_{\tau=0}^{t} RHS\Big(\tau, \begin{bmatrix} d_c \\ d_e \end{bmatrix}\Big) d\tau \doteq \begin{bmatrix} \bar{d}_c(t) \\ \bar{d}_e(t) \end{bmatrix}, \quad d_c(0) = \bar{d}_c(0),\ d_e(0) = \bar{d}_e(0). \quad (20)$$

Evaluating the integral in (20) is efficient as RHS is affine, so $\bar{d}_c$ and $\bar{d}_e$ can be readily used in planning (cf. Sec. 4.4). However, note that these tubes are only *locally valid*, e.g., evaluating the tubes outside of D_x will give incorrect values. We detail a set of validity conditions in Sec. 4.4, prove their sufficiency in Thm. 2, use them in our planner, and show in Sec. 5 that a baseline that ignores these conditions is unsafe. Finally, we close with a remark on how we estimate the constants in the bounds.

Remark 1 (Estimating constants from data). The derived bounds depend on several constants that are unknown *a priori*, such as $L_{\Delta k}$ and $L_{\hat{h}^{-1}}$, and if $\bar{\epsilon}_1$, $\bar{\epsilon}_2$, or $\bar{\epsilon}_3$ is being used, $\bar{\epsilon}_1$, L_p, and $\{L_p, \mathcal{R}\}$ also need to be estimated, respectively. As overapproximating each constant also yields valid (and looser) bounds, we use the i.i.d. validation set $\mathcal{V}$ to overestimate each constant via a sampling-based approach based on extreme value theory [5]. This returns a value which overestimates the true constant with a user-desired probability δ, where δ holds in the limit of infinite samples. See [5, 13, 28] for details.

4.3 Optimizing CCMs and OCMs for output feedback

We briefly discuss how we obtain the CCM/OCM that define the controller/observer; for space, we detail our method in App. D. We write two SoS programs to independently synthesize the CCM/OCM, which are approximately optimized to minimize their tube sizes. We search over polynomial CCMs and constant OCMs. For polynomial dual CCMs $W_c(x)$, we also find a constant metric $\bar{W}_c \succeq W_c(x)$, for all x, in order to simplify constraint checking in Sec. 4.4. For linear systems, these SoS programs simplify to a standard semidefinite program (SDP), which scale to higher-dimensional systems.

4.4 Solving the OFMP

Algorithm 1: **C**ontraction-based **O**utput feedback **RRT** (CORRT)

Input: $x_I, \mathcal{G}, \theta, \mathcal{S}$, training error $\{e_i\}_{i=1}^{N_{\text{data}}}$, estimated constants, $\bar{d}_c(0), \bar{d}_e(0), \bar{c}, \bar{e}$

$\mathcal{T} \leftarrow \{(x_I, \bar{d}_c(0), \bar{d}_e(0), 0)\}$ `// node: state, CCM/OCM Riem. dist. bound, time`
$\mathcal{P} \leftarrow \{(\emptyset, \emptyset)\}$ `// parent: previous control/dwell time`
while True **do**
  $(x_{\text{n}}, \bar{d}_c^n, \bar{d}_e^n, t_{\text{n}}) \leftarrow$ `SampleNode(`$\mathcal{T}$`)` `// sample node from tree`
  $(u_{\text{p}}, t_{\text{p}}) \leftarrow$ `SampleProposedControl()` `// sample ctrl/dwell time`
  $(x_{\text{p}}^*(t), u_{\text{p}}^*(t)), t \in [t_n, t_n + t_p) \leftarrow$ `IntegrateDyn(`$x_{\text{n}}, u_{\text{p}}, t_{\text{p}}$`)` `// get extension`
  $(\bar{d}_c^n(t), \bar{d}_e^n(t)), t \in [t_n, t_n + t_p) \leftarrow$ `ErrBnd(`$\bar{d}_c^n, \bar{d}_e^n, x_{\text{p}}^*(t), u_{\text{p}}^*(t), \mathcal{S}, \{e_i\}, \theta$`)` `// new tube`
  $(b_L^c, b_L^e) \leftarrow (\bar{d}_c^n(t) \le \bar{c}, \bar{d}_e^n(t) \le \bar{e}), \forall t \in [t_n, t_n + t_p)$ `// check upper bound`
  $b_c \leftarrow \Omega_c^n(t) \subseteq (D_c \cap D_r \cap \mathcal{X}_{\text{safe}}), \forall t \in [t_{\text{n}}, t_{\text{n}} + t_{\text{p}})$ `// check tracking tube`
  $b_e \leftarrow \Omega_c^n(t) \oplus (\Omega_e^n(t) \ominus \{x(t)\}) \subseteq (D_e \cap D_c), \forall t \in [t_{\text{n}}, t_{\text{n}} + t_{\text{p}})$ `// chk. estimator tube`
  if $b_L^c \wedge b_L^e \wedge b_c \wedge b_e$ **then** $\mathcal{T} \leftarrow \mathcal{T} \cup \{(x_c^*(t_{\text{n}} + t_{\text{p}}), \bar{d}_c^n(t_{\text{n}} + t_{\text{p}}), \bar{d}_e^n(t_{\text{n}} + t_{\text{p}}), t_{\text{p}})\}$;
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(u_{\text{p}}, t_{\text{n}} + t_{\text{p}})\}$
  else continue `// add extension if all checks pass`
  if $\exists t, \Omega_c^n(t) \subseteq \mathcal{G}$ **then** break; return plan `// return if in` $\mathcal{G}$

Given the CCM, OCM, and the ability to compute tracking tubes, we can now solve the OFMP. Our solution builds upon a kinodynamic RRT [15], though we note that the tubes derived in Sec. 4.2 are planner-agnostic. We grow a search tree $\mathcal{T}$ by integrating sampled controls held for sampled dwell-times until $\mathcal{G}$ is reached. To ensure we stay in $\mathcal{X}_{\text{safe}}$ at runtime, we impose extra constraints on each candidate transition, which are informed by the tubes; this translates to a restriction on where $\mathcal{T}$ can grow (cf. Fig. 5).

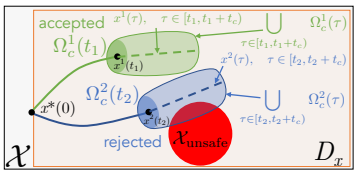

Fig. 5. Visualization of Alg. 1.

To use the Riemannian distance bounds $\bar{d}_c(t)$ and $\bar{d}_e(t)$ from (20) in planning, recall that these bounds define sets centered around $x^*(t)$ and $x(t)$, $\Omega_c(t)$ and $\Omega_e(t)$, which x and $\hat{x}$ are guaranteed to remain within. We can use these sets for collision and constraint checking. If the metric defining $\Omega(t)$ is constant, each $\Omega(t)$ defines an ellipsoid, i.e., $\Omega_c(t) = \{x(t) \mid (x(t) - x^*(t))^\top M_c(x(t) - x^*(t)) \le \bar{d}_c(t)^2\}$ and $\Omega_e(t) = \{\hat{x}(t) \mid (\hat{x}(t) - x(t))^\top W_e(\hat{x}(t) - x(t)) \le \bar{d}_e(t)^2\}$. If the metric is state-dependent (as is the case for some CCMs we use), we can use $\bar{W}_c$ (see Sec. 4.3) to obtain an ellipsoidal outer approximation of $\Omega_c(t)$: $\Omega_c(t) \subseteq \{x(t) \mid (x(t) - x^*(t))^\top (\bar{W}_c)^{-1}(x(t) - x^*(t)) \le$

$\bar{d}_c(t)^2\} \doteq \tilde{\Omega}_c(t)$ that can ease constraint checking. Thus, we can guarantee *at planning time* that *in execution*, $x(t) \in \tilde{\Omega}_c(t)$, and $\hat{x}(t) \in \tilde{\Omega}_c(t) \oplus (\Omega_e(t) \ominus \{x(t)\})$, where $A \ominus B \doteq \{x - y \mid x \in A, y \in B\}$. As (20) defines Ω for *any* nominal trajectory, we can quickly compute tubes along all edges in $\mathcal{T}$. For instance, suppose we wish to extend from some node in $\mathcal{T}$, $x_n^*(t_n)$, which satisfies $d_c^n(t_n) \leq \bar{d}_c^n(t_n)$ and $d_e^n(t_n) \leq \bar{d}_e^n(t_n)$, to a candidate state $x_n^*(t_n + t_p)$ by applying control u over $[t_n, t_n + t_p)$. Then, using (20), we can obtain $\bar{d}_c^n(t)$ and $\bar{d}_e^n(t)$, for all $t \in [t_n, t_n + t_p)$, and to remain collision-free in execution, we require the induced $\tilde{\Omega}_c(t) \subseteq \mathcal{X}_{\text{safe}}$; we check this in line 9 of our planner, Alg. 1. Here, we assume obstacles are inflated to account for robot geometry.

To remain collision-free at runtime, we must add extra constraints on $\mathcal{T}$ to ensure the tubes are valid, as discussed in Sec. 4.2. We describe these constraints now, and prove they are sufficient in Thm. 2. At a high level, the estimated constants, CCM, and OCM must be valid for any x and $\hat{x}$ that can be reached at runtime. Thus, in line 8, we ensure $d_c(t)$ and $d_e(t)$ remain less than $\bar{c}$ and $\bar{e}$ for all time, so that $L_{\Delta k}$ (15) is valid. In line 9, we ensure that $\Omega_c(t) \subseteq D_c \cap D_r$, i.e., the system remains where the controller can contract x towards x^*, and $\bar{\epsilon}_i$ is valid. In line 10, we ensure $\hat{x}$ remains in $D_e \cap D_c$; this ensures that (6) contracts towards the true state x via (2), and that a feasible feedback control exists in (13); ensuring this at planning time (when we only know $x^*(t)$) requires a Minkowski sum of Ω_c and $\Omega_e \ominus \{x(t)\}$. Constraint-satisfying candidate extensions are added to $\mathcal{T}$ (line 11); else, they are rejected (line 12). This continues until the goal is reached (line 13). We visualize our planner (Fig. 5), **C**ontraction-based **O**utput feedback **RRT** (CORRT), detailed in Alg. 1. Finally, Thm. 2 shows our method ensures safety and goal reachability if all estimated constants are valid; as our estimates are probabilistically-valid, the overall guarantees are probabilistic (cf. Rem. 2):

Theorem 2 (CORRT correctness). *Assume that $L_{\Delta k}$, $L_{\hat{h}^{-1}}$, and the estimated constants in $\bar{\epsilon}_{\{1,2,3\}}$ are valid over their computed domains. Then Alg. 1 returns a trajectory $(x^*(t), u^*(t))$, which when tracked on the true system* (1a) *using $u(\hat{x}, x^*, u^*)$ with state estimates $\hat{x}$ generated by* (6)*, reaches $\mathcal{G}$ while satisfying $x(t) \in \mathcal{X}_{safe}$, for all $t \in [0, T]$.*

5 Results

We evaluate CORRT on a 4D car with RGB-D observations, a 6D quadrotor with RGB observations, and a 14D acceleration-controlled 7DOF arm with RGB observations. All observations are rendered in PyBullet. We compare with three baselines; two are shared across experiments, so we overview them here. To show the need to plan where the CCM/OCM are valid and the error bounds are accurate, Baseline 1 (B1) plans using the tracking tubes from (20) inside Alg. 1 but is *not constrained* to stay within D, i.e., the checks in line 8-10 of Alg. 1 are relaxed. To show the need to consider estimation error in planning, Baseline 2 (B2) assumes perfect state knowledge in computing its tubes, i.e., $d_e(t) \equiv 0$. All baselines execute with the same CCM/OCM as our method. See Table 1 for error statistics and the video http://tinyurl.com/wafr22corrt for visualizations.

	CORRT trk. err.	CORRT est. err.	B1 trk. err.	B1 est. err.	B2 trk. err.	B2 est. err.	B3 trk. err.	B3 est. err.
Car	0.175 ± 0.117	0.032 ± 0.022	17.49 ± 79.86	143.4 ± 1202	1.520 ± 6.306	3.597 ± 19.90	—	—
Quad	0.151 ± 0.187	0.029 ± 0.028	39.30 ± 142.1	52.64 ± 185.9	40.56 ± 302.1	63.53 ± 424.1	—	—
Arm	2.0e-4 ± 1.3e-5	0.053 ± 0.039	2.0e-4 ± 1.4e-5	0.145 ± 0.239	—	—	0.000 ± 0.000	0.316 ± 0.249

Table 1. Statistics on the tracking/estimation error reduction across all experimental results. "Trk. err." $= \|x^*(T) - x(T)\| / \|x^*(0) - x(0)\|$. "Est. err." $= \|\hat{x}(T) - x(T)\| / \|\hat{x}(0) - x(0)\|$. In each cell: average error ± standard deviation over all trials.

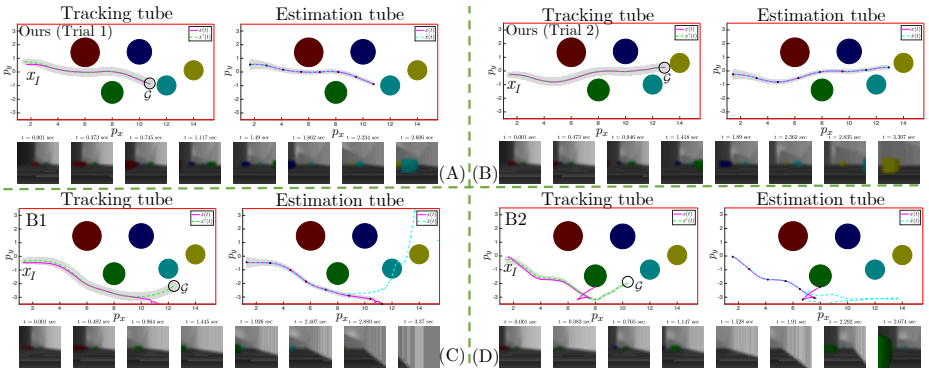

Fig. 6. 4D car. Planned, executed, and estimated trajectories, overlaid with corresponding tracking and estimation tubes $\Omega_c(t)$ and $\Omega_e(t)$. For eight timesteps corresponding to the black dots on the Ω_e plot, we also show RGB component of the observations seen at runtime (bottom). A) and B): two examples of CORRT, which safely reach the goal. C) and D): B1 and B2: both crash.

4D nonholonomic car We consider a ground vehicle in an obstacle field (Fig. 1.A), governed by (E.15). The observations are given by 48x48 RGB-D images taken from a front-facing onboard camera (Fig. 1.A, inset); this makes $y \in \mathbb{R}^{9216}$. Three states can be directly inferred from a single image: p_x, p_y, and ϕ. For this example, $\theta \in \mathbb{R}^5$ parameterizes the p_y-translation of each of the five obstacles. We are given $N_{\text{data}} = 250000$ datapoints to train the perception system $\hat{h}^{-1}$, sampled uniformly from $C_r D_p = [0, 13.5] \times [-2.5, -2.5] \times [-\pi/3, \pi/3]$ and $D_\theta = [0.5, 1.5] \times [-1.5, -0.5] \times [0.5, 1.5] \times [-1, 0] \times [0, 1]$. We model $\hat{h}^{-1}$ as a fully-connected neural network, with five hidden layers of width 1024 and softplus activations. We use the method of Sec. 4.3 to obtain a constant CCM M_c with $\bar{\lambda}(M_c) = 1$, $\underline{\lambda}(M_c) = 0.07$, and $\lambda_c = 2.5$, and a constant OCM M_e with $\bar{\lambda}(W_e) = 5.44$, $\underline{\lambda}(W_e) = 0.05$, and $\lambda_o = 0.6$, where $D_c = (-\infty, \infty)^2 \times [-\pi/3, \pi/3] \times [2, 5] = D_e$. To compute our tubes in CORRT, we use $\bar{\epsilon}_3(x^*, \theta)$, since for this example Ω_c may be large. The constants are estimated to be $L_{\Delta k} = 3.28$, $L_{\hat{h}^{-1}} = 0.05$, $L_p = 0.024$, and $\mathcal{R} = 0.69$. In computing our tubes, we assume $\|w_x\| \leq 0.05$, $\bar{d}_c(0) = 0.2$, $\bar{d}_e(0) = 0.1$, and $w_y \in \mathbb{R}^{n_y}$ satisfies $\|w_y\| \leq 0.25$. To simulate noisy depth images, B_y is set to be a diagonal $n_y \times n_y$ matrix, with 0 diagonal entries for RGB indices and 1 for the depth indices.

We plan for 150 start/goals in D; our unoptimized implementation takes 2.5 minutes on average. This is done offline; the tracking controller is computed at real-time rates following Sec. 4.2 and [22]. For each trial, the obstacle map θ is selected uniformly within D_θ. See Table 1 for error statistics. Over all trials, our method ensures $x(t)$ and $x^*(t)$ always remain within the CORRT-computed $\Omega_c(t)$ and $\Omega_e(t)$, respectively, and reduces the initial tracking/estimation error by a factor of > 5 and 30, respectively. In contrast, B1 violates its $\Omega_c(t)$ and $\Omega_e(t)$ in 90/150 and 101/150 trials, respectively, fails to reduce tracking/estimation error, and can crash. For instance, in Fig. 6.C, the plan leaves D_r, causing observation error to increase (here, $\hat{h}$ is inaccurate, since it is not trained outside of D_r), destabilizing $\hat{x}$ (Fig. 6.C, right), in turn destabilizing x, leading to the crash. Similarly, B2 violates its computed Ω_c in 60/150 trials (no $\Omega_e(t)$ is computed for B2, as it assumes perfect state information), fails to shrink tracking/estimation errors, leading to crashes (see Fig. 6). As in B1, this crash also arises from observation error. Overall, this experiment suggests that CORRT ensures safe goal-reaching for non-

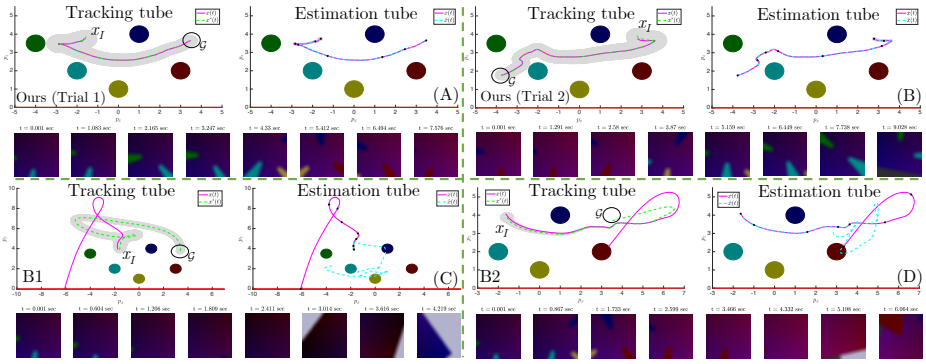

Fig. 7. 6D quadrotor. Planned, executed, and estimated trajectories, overlaid with $\Omega_c(t)$ and $\Omega_e(t)$. Snapshots of the runtime observations are shown (bottom). A) and B): two examples of CORRT, which safely reach the goal. C) and D): B1 and B2: both crash.

holonomic systems using RGB-D data, and that it generalizes to different environments (i.e., obstacle layouts), while baselines are unsafe.

6D quadrotor We consider a planar quadrotor in an obstacle field (Fig. 1.B), governed by (E.16). The observations are given by 48x48 RGB images taken from a front-facing onboard camera (Fig. 1.B, inset); this makes $y \in \mathbb{R}^{6912}$. Three states can be directly inferred from an image: p_x, p_z, and ϕ. Here, we consider a single set of map configurations, i.e., θ is a singleton. We are given $N_{\text{data}} = 140000$ datapoints to train $\hat{h}^{-1}$, sampled uniformly from $C_r D_p = [-4.5, 4.5] \times [0.5, 4.5] \times [-\pi/4, \pi/4]$. We model $\hat{h}^{-1}$ as a fully-connected neural network, with five hidden layers of width 1024 and ReLU activations. Using the method of Sec. 4.3, we obtain a polynomial CCM M_c with $\bar{\lambda}_{D_c}(M_c) = 6.55$, $\underline{\lambda}_{D_c}(M_c) = 0.22$, and $\lambda_c = 0.8$, and a constant OCM M_e with $\bar{\lambda}(W_e) = 8.13$, $\underline{\lambda}(W_e) = 0.1$, and $\lambda_e = 0.7$, where $D_c = (-\infty, \infty)^2 \times [-\pi/3, \pi/3] \times [-4.5, 4.5] \times [-1, 1] \times [-2, 2]$ and $D_e = (-\infty, \infty)^2 \times [-\pi/4, \pi/4] \times [-5, 5] \times [-2.5, 2.5] \times [-2.5, 2.5]$. To update our tracking tubes in CORRT, we found it sufficient to use the first error bound $\bar{\epsilon}_1$, which we estimate to be 0.008, and $L_{\Delta k} = 3.6$. In computing our tubes, we assume $\|w_x\| \leq 0.0125$, $\bar{d}_c(0) = 0.15$, $\bar{d}_e(0) = 0.1$, and noiseless images $\|w_y\| = 0$.

We plan for 150 start/goals in D, taking 6 minutes on average (see Table 1 for statistics). Across all trials, CORRT ensures $x(t)$ and $\hat{x}(t)$ stay inside the CORRT-computed tubes $\Omega_c(t)$ and $\Omega_e(t)$, respectively, and reduces the initial tracking/estimation error by a factor of > 6 and 34. In contrast, B1 violates its computed $\Omega_c(t)$ and $\Omega_e(t)$ in 61/150 and 76/150 trials, respectively, fails to reduce error, and can be unsafe (see Fig. 7). Similarly, B2 violates its Ω_c in 142/150 trials. We show concrete examples of this in Fig. 7.C-.D; the plans in both cases exit D_r, moving to p_x and p_z values outside of the $[-4.5, 4.5] \times [0.5, 4.5]$ training range, leading to high $\hat{h}^{-1}$ error. The plans also take overly-aggressive turns that bring the velocities outside of D_e and D_c; this further destabilizes the system, causing crashes in both cases. Overall, this experiment suggests the need to ensure that $\hat{h}^{-1}$, the CCM, and the OCM are correct, and that CORRT ensures this to guarantee safety for underactuated systems via RGB observations.

17D manipulation task We consider an acceleration-controlled 7DOF Kuka arm, where each joint follows double integrator dynamics (E.18), which is grasping an object (a rubber duck) with an unknown orientation relative to the end effector. We assume

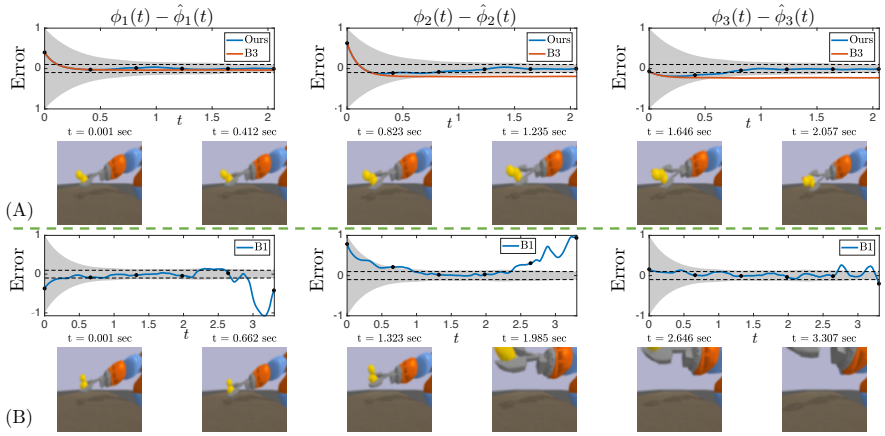

Fig. 8. 7DOF arm. State estimate error, overlaid with $\Omega_e(t)$ (in gray). Runtime observations are shown (bottom). A): when using CORRT, the state estimate error remains in $\Omega_e(t)$ and achieves $|\hat{\phi}_i(T) - \phi_i(T)| \leq 0.1$. B3 fails to meet this requirement. B) B1 also fails the 0.1 requirement.

slight noise in the dynamics (E.18), $\bar{w}_x = 0.0125$, due to the weight of the object. Our goal is to estimate the unknown orientation, represented as three Euler angles $\{\phi_i\}_{i=1}^3$, using our observer (6), given 80x80 RGB images (Fig. 1.C) of the arm and grasped object (see Fig. 1.C, inset); this makes $y \in \mathbb{R}^{19200}$. We may also plan motions for the arm to improve the quality of the observations/state estimates, though in doing so, we also need to counteract the dynamics error. Our overall goal is to guarantee our final estimate of the relative orientation satisfies $|\phi_i(T) - \hat{\phi}_i(T)| \leq 0.1$, $i = 1, 2, 3$.

We assume that the joint angles and velocities can be perfectly estimated (i.e., directly measured), given the accuracy of the Kuka joint encoders, focusing instead on estimating the unknown $\{\phi_i\}_{i=1}^3$ and controlling j and $\dot{j}$ (the joint angles and velocities) using our method. We assume the object is rigidly attached to the gripper, such that its relative orientation is constant over time. Combining $\{\phi_i\}_{i=1}^3$ and the 14D model, the full state of the system is 17D (E.17), i.e., $x = [\phi_1, \phi_2, \phi_3, j_1, \ldots, j_7, \dot{j}_1, \ldots \dot{j}_7]^\top$. To train $\hat{h}^{-1}$, we note that $\{\phi_i\}_{i=1}^3$ can all be estimated from the image. For this example, since j is known and affects the generated y, we design $\hat{h}^{-1}$ to take as input $y \in \mathbb{R}^{19200}$ and $j \in \mathbb{R}^7$ (i.e., j plays the role of θ) and to output $\{\phi_i\}_{i=1}^3$. We are given $N_{\text{data}} = 62500$ datapoints to train $\hat{h}^{-1}$, where $\{\phi_i\}_{i=1}^3$ are sampled uniformly from $[-\pi/3, \pi/3]^3$ and j is sampled uniformly from $[-0.05, 0] \times [0, 0.05] \times [0.15, 0.32] \times [-1.83, -1.69] \times [-0.05, 0.05]^2 \times [-\pi/3, \pi/3]$. We model $\hat{h}^{-1}$ as a fully-connected neural network, with five hidden layers of width 1024 and softplus activations. We compute a constant CCM for the 14D subsystem: CCM synthesis for the full 17D system fails, as the $\{\phi_i\}_{i=1}^3$ are not controllable due to the rigid attachment. Since the arm dynamics are linear, the CCM optimization simplifies to a standard semidefinite program that can be quickly solved. We compute a constant OCM for the full 17D system, to enable estimation of $\{\phi_i\}_{i=1}^3$. Using the method of Sec. 4.3, we obtain a CCM M_c with $\bar{\lambda}(M_c) = 100$, $\underline{\lambda}(M_c) = 2.81$, and $\lambda_c = 2.89$, and a constant OCM M_e with $\bar{\lambda}(W_e) = \underline{\lambda}(W_e) = 0.1$, and $\lambda_e = 9.5$. As the dynamics are linear, a constant CCM/OCM holds globally, i.e., $D_e = D_c = \mathcal{X}$. To update the tubes in CORRT, we use $\bar{\epsilon}_2(x^*, j)$, where we estimate $L_p = 2.45$. Since j and $\dot{j}$ are known, no error arises

from incorrect state estimates; thus, $L_{\Delta k}$ does not need to be estimated. We assume $\bar{d}_c(0) = 10^{-3}$, $\bar{d}_e(0) = 0.32$, and noiseless images $\|w_y\| = 0$.

We plan 100 trajectories in D from various initial j, $\dot{j}$, and orientation estimates, taking 45 seconds on average. We summarize the error statistics in Table 1. Across all trials, when planning with CORRT, x and $\hat{x}$ always remain within the computed tubes $\Omega_c(t)$ and $\Omega_e(t)$; the CCM keeps the tracking error very small, and the OCM shrinks the error by a factor of > 18. Crucially, if a plan is found where $\Omega_e(T)$ satisfies the estimation accuracy threshold, we can ensure our true state estimate satisfies $|\phi_i(T) - \hat{\phi}_i(T)| \leq 0.1$, $i = 1, 2, 3$. We are able to find plans that achieve this threshold for 100/100 trials. We compare with two baselines in this example: B1 (as described before), and B3, which keeps the arm stationary and runs (6) for the same duration as the plan computed using CORRT. The purpose of B3 is to show that the actions taken by the CORRT plan help to reduce estimation error. In contrast to CORRT, B1 violates its computed $\Omega_e(t)$ in 44/100 trials and can fail to achieve the required estimation accuracy, only satisfying the 0.1 threshold in 79/100 trials (see Fig. 8). One failure example is shown in Fig. 8.B: the arm moves too close to the camera (outside of D_r), causing the duck to fall out of frame. This causes a sharp increase in $\hat{h}^{-1}$ error, since ϕ_i cannot be observed; this destabilizes (6), leading to a failure to satisfy the 0.1 threshold. Note that B1 does not violate Ω_c; this is because the controller is not a function of the incorrect ϕ_i estimates. Similarly, B3 often fails to satisfy the 0.1-estimation accuracy threshold, only satisfying it in 7/100 trials (see Fig. 8.A for a failure example). This shows that passively estimating ϕ_i without moving the arm cannot achieve the needed estimation accuracy; instead, the arm must be moved towards regions with smaller perception error. Overall, this experiment suggests the applicability of our approach on high-dimensional systems, that it can design actions that improve state estimates, and that our approach can plan paths that guarantee a desired level of state estimation accuracy.

6 Discussion and Conclusion

We present a motion planning algorithm for control-affine systems that enables safe tracking at runtime using an output feedback controller with image observations as input. To achieve this, we learn a perception system and use it in an OCM and CCM-based output feedback control loop. We derive tracking tubes for the closed-loop system and use them within an RRT-based planner to compute plans that theoretically guarantee safe goal-reaching at runtime. Our results empirically validate this safety guarantee, and show that ignoring the effects of state estimation error and the local validity of the perception system/estimator/controller can lead to unsafe behavior.

Our method has some weaknesses which reveal directions for future work. While the large dataset $\mathcal{S}$ used to train $\hat{h}^{-1}$ is easy to gather in simulation, sim-to-real is then needed for $\hat{h}^{-1}$ to transfer to the real world. Thus, in future work, we will combine synthetic, domain-randomized perception data with a small real-world labeled dataset to train generalizable perception modules that have calibrated estimates of the sim-to-real error. Our method also assumes noiseless training data, to ensure L_p is finite; in the future, we wish to relax this by investigating Lipschitz constant estimation methods robust to input noise [4] . Another drawback is the conservativeness of using worst-case disturbances; to mitigate this, we will integrate stochastic contraction [11] into our method. Finally, we require θ to be known; in future work, we will aim to jointly estimate θ and x with similar convergence guarantees.

References

1. Agha-mohammadi, A., Chakravorty, S., Amato, N.: FIRM: sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. IJRR (2014)
2. Bahreinian, M., Mitjans, M., Tron, R.: Robust sample-based output-feedback path planning. In: IROS. pp. 5780–5787. IEEE (2021)
3. Bonnabel, S., Slotine, J.E.: A contraction theory-based analysis of the stability of the deterministic extended kalman filter. TAC **60**(2), 565–569 (2015)
4. Calliess, J.: Conservative decision-making&inference in uncertain dynamical systems (2014)
5. Chou, G., Ozay, N., Berenson, D.: Model error propagation via learned contraction metrics for safe feedback motion planning of unknown systems. CDC (2021)
6. Cosner, R., Singletary, A., Taylor, A., Molnár, T., Bouman, K., Ames, A.: Measurement-robust control barrier functions: Certainty in safety with uncertainty in state. In: IROS (2021)
7. Dani, A.P., Chung, S., Hutchinson, S.: Observer design for stochastic nonlinear systems via contraction-based incremental stability. TAC **60**(3), 700–714 (2015)
8. Dawson, C., Lowenkamp, B., Goff, D., Fan, C.: Learning safe, generalizable perception-based hybrid control with certificates. RA-L (2022)
9. Dean, S., Matni, N., Recht, B., Ye, V.: Robust guarantees for perception-based control. In: L4DC. vol. 120, pp. 350–360. PMLR (2020)
10. Dean, S., Taylor, A.J., Cosner, R.K., Recht, B., Ames, A.D.: Guaranteeing safety of learned perception modules via measurement-robust control barrier functions. In: CoRL (2020)
11. Kawano, Y., Hosoe, Y.: Contraction analysis of discrete-time stochastic systems (2021)
12. Kloss, A., Martius, G., Bohg, J.: How to train your differentiable filter. AURO (2021)
13. Knuth, C., Chou, G., Ozay, N., Berenson, D.: Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants. IEEE RA-L (2021)
14. Lakshiliikantham, V., Leela, S.: In: Differential and Integral Inequalities - Theory and Applications: Ordinary Differential Equations, vol. 55, pp. 3–44 (1969)
15. LaValle, S.: Planning algorithms. Cambridge university press (2006)
16. Majumdar, A., Tedrake, R.: Funnel libraries for real-time robust feedback motion planning. IJRR **36**(8), 947–982 (2017)
17. Manchester, I.R., Slotine, J.E.: Output-feedback control of nonlinear systems using control contraction metrics and convex optimization. In: Australian Control Conference (2014)
18. Manchester, I.R., Slotine, J.E.: Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design. IEEE Trans. Autom. Control. **62**(6), 3046–3053 (2017)
19. Manchester, I.R., Tang, J.Z., Slotine, J.E.: Unifying robot trajectory tracking with control contraction metrics. In: ISRR. vol. 3, pp. 403–418. Springer (2015)
20. Maybeck, P.S.: Stochastics models, estimation, and control (1979)
21. Renganathan, V., Shames, I., Summers, T.H.: Towards integrated perception and motion planning with distributionally robust risk constraints. IFAC World Congress (2020)
22. Singh, S., Landry, B., Majumdar, A., Slotine, J.E., Pavone, M.: Robust feedback motion planning via contraction theory (2019)
23. Sun, D., Jha, S., Fan, C.: Learning certified control using contraction metric. CoRL (2020)
24. Sunberg, Z.N., Kochenderfer, M.J.: Online algorithms for pomdps with continuous state, action, and observation spaces. In: ICAPS. pp. 259–263. AAAI Press (2018)
25. Tsukamoto, H., Chung, S.: Learning-based robust motion planning with guaranteed stability: A contraction theory approach. RA-L **6**(4), 6164–6171 (2021)
26. Tsukamoto, H., Chung, S.: Neural contraction metrics for robust estimation and control: A convex optimization approach. IEEE CSL **5**(1), 211–216 (2021)
27. Veer, S., Majumdar, A.: Probably approximately correct vision-based planning using motion primitives. In: CoRL. vol. 155, pp. 1001–1014. PMLR (2020)
28. Weng, T.W., Zhang, H., Chen, P.Y., Yi, J., Su, D., Gao, Y., Hsieh, C.J., Daniel, L.: Evaluating the robustness of neural networks: An extreme value theory approach. ICLR (2018)
29. Yang, H., Shi, J., Carlone, L.: TEASER: fast and certifiable point cloud registration. T-RO **37**(2), 314–333 (2021)