# Safe-by-Design Planner-Tracker Synthesis with Unmodeled Input Dynamics

Katherine S. Schweidel, Peter J. Seiler, and Murat Arcak

***Abstract*— The planner-tracker framework is a hierarchical control scheme wherein a motion plan is generated using a simplified model, and a tracking controller synthesized offline keeps the error between the true and simplified model trajectories small. This paper extends the planner-tracker framework to accommodate unmodeled input dynamics, described by integral quadratic constraints (IQCs). A sum-of-squares (SOS) program is formulated to search for the tracking controller and error bound, and the method is demonstrated on a vehicle obstacle avoidance example with input delay.**

## I. INTRODUCTION

Complex systems, such as autonomous vehicles [1] and missile guidance systems [2], use hierarchical control schemes where each control layer uses a different system model. This approach can enhance computational efficiency, as the higher-level control layer can use a simpler model to reduce computation times and to enable control strategies that may not be possible with the more complex model. Faster higher-level control can also allow the system to respond to changing environments in real-time.

Such hierarchical schemes, however, may be unsafe if the error between the models in different layers is not accounted for. A high-level motion plan generated using a simplified model may avoid obstacles that the lower-level controller is unable to. Thus, for safety, each control layer must accommodate the error arising from different models.

In the planner-tracker framework [3]–[10], a lower-fidelity "planning" model is employed for online planning and a "tracking" controller, synthesized offline, keeps the tracking error between the high-fidelity ("tracking") model and the planning model within a bounded set. System safety is then guaranteed if the planner constraints, when augmented by the tracking error bound, lie within the safety constraints.

Tube Model Predictive Control (MPC) is another approach for handling the error between an uncertain model and a nominal model used for planning [11]–[19]. Tube MPC is a robust control strategy where the error between the true model and a nominal model, free of disturbances, is bounded within a tube, and MPC is performed on the nominal model with state constraints shrunk by the size of the tube. Typically, the true and nominal models differ by only an additive disturbance in the true model. However, this may not capture the total uncertainty in the true model. Besides disturbances, another source of uncertainty is unmodeled dynamics.

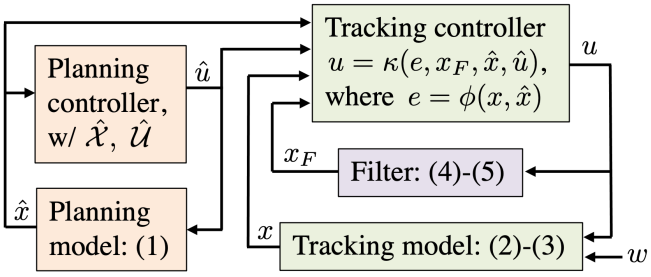


Fig. 1: Planner-tracker scheme. The planning controller uses the state $\hat{x}$ and constraints $\hat{\mathcal{X}}$ and $\hat{\mathcal{U}}$ to generate a reference input $\hat{u}$. Using a filter state $x_F$ as an additional input, the tracking controller converts this into a control $u$ which is guaranteed to keep the tracker state $x$ within state constraints $\mathcal{X}$. This is accomplished by keeping the tracking error $e$ within a set $\mathcal{O}$ whose volume is minimized.

The more sources of uncertainty we can accommodate in the higher-fidelity model, the more the models will reflect reality. In this paper, we extend the planner-tracker framework to handle unmodeled dynamics at the input of the tracker model. We characterize the uncertainty using an Integral Quadratic Constraint (IQC), which is an essential tool for ensuring robustness to unmodeled dynamics [20], [21]. In particular, we use $\alpha$-IQCs, which contain an exponential weighting factor compared to standard IQCs [22]–[24].

These $\alpha$-IQCs can be used to describe many uncertainties including unknown delays or unmodeled actuator dynamics. The Smith predictor [25] is a common compensator for systems with large delays, but the approach presented here provides safety guarantees for an *unknown* delay in a range.

The most closely related works are [23], [24], which also use $\alpha$-IQCs to incorporate unmodeled dynamics in shrinking tube MPC. However, attention is restricted to linear systems, whereas this paper is applicable to nonlinear systems. In [26], the tracking error is kept in a funnel without knowledge of the tracker model, but the control input can grow unbounded.

In Section II, we derive the error dynamics between the planner and the tracker model with unmodeled input dynamics. In Section III, we describe conditions that an error bound set must satisfy, and in Section IV we convert these conditions into an SOS program. We solve this SOS program for a vehicle obstacle avoidance example in Section V.

*Notation:* For $\xi \in \mathbb{R}^n$, $\mathbb{R}[\xi]$ represents the set of polynomials in $\xi$ with real coefficients, and $\mathbb{R}^m[\xi]$ and $\mathbb{R}^{m\times p}[\xi]$

Katherine S. Schweidel is with the Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: kschweidel@berkeley.edu).

Peter J. Seiler is with the Department of Electrical Engineering and Computer Science, University of Michigan at Ann Arbor, Ann Arbor, MI 48109 USA (e-mail: pseiler@umich.edu).

Murat Arcak is with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: arcak@berkeley.edu).

This work was supported by NSF grants CNS-2111688, ECCS-1906164, and AFOSR grant FA9550-21-1-0288.

denote all vector and matrix valued polynomial functions. The subset $\Sigma[\xi] := \{p = p_1^2 + p_2^2 + ... + p_M^2 : p_1, ..., p_M \in \mathbb{R}[\xi]\}$ of $\mathbb{R}[\xi]$ is the set of sum-of-squares polynomials in $\xi$. We say a matrix-valued polynomial $P \in \mathbb{R}^{m\times m}[\xi]$ is in $\Sigma_m[\xi]$ if $y^\top P y \in \Sigma[\xi, y]$, which implies $P$ is positive semidefinite for all $\xi$. For a function $V : \mathbb{R}^n \to \mathbb{R}$ and a scalar $\gamma \in \mathbb{R}$, we denote the $\gamma$-sublevel set of $V$ as $\Omega(V, \gamma) := \{x \in \mathbb{R}^n : V(x) \leq \gamma\}$.

## II. PROBLEM FORMULATION

### A. Planner-Tracker Framework

We now describe the planner-tracker framework, extended to include unmodeled input dynamics (Figure 1). The low-fidelity model, called the *planning* model $P$, has the form:

$$\dot{\hat{x}} = \hat{f}(\hat{x}, \hat{u}), \tag{1}$$

where $\hat{x} \in \hat{\mathcal{X}} \subseteq \mathbb{R}^{\hat{n}_x}$ and $\hat{u} \in \hat{\mathcal{U}} \subseteq \mathbb{R}^{\hat{n}_u}$ are the state and control input, respectively, of the planner system.

A higher-fidelity model, referred to as the *tracking* model, is used to track the trajectory generated for the planning model above. The tracking model is of the form:

$$\dot{x} = f(x, w) + g(x, w)(u + l), \tag{2}$$
$$l = \Delta(u), \tag{3}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the tracker state, $w \in \mathcal{W} \subseteq \mathbb{R}^{n_w}$ is an external disturbance, and $u \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ is the control input which will depend on $\hat{x}$ and $\hat{u}$ with the goal of keeping the error between the planner and tracker systems small.

An uncertain block $\Delta$ at the tracker input with output $l \in \mathbb{R}^{n_l}$ is added to the tracker input in (2). This block encompasses unmodeled dynamics, e.g., delays, where the resulting signal $l$ is a function of $u$ and/or $x$. This is distinct from the exogeneous disturbance $w$ which does not depend on $x$ or $u$ and is restricted, a priori, to $\mathcal{W}$. The presence of $\Delta$ and $l$ is what distinguishes this paper from previous works in the planner-tracker framework literature.

### B. Integral Quadratic Constraint

We assume that $\Delta$ satisfies an input-output relationship called an $\alpha$ Integral Quadratic Constraint ($\alpha$-IQC) for some constant scalar $\alpha \in \mathbb{R}$. To define this relationship, the input $u$ is passed through a linear time-invariant filter $F$ with state $x_F \in \mathbb{R}^{n_F}$ and output $z \in \mathbb{R}^{n_z}$:

$$\dot{x}_F = A_F x_F + B_F u, \tag{4}$$
$$z = C_F x_F + D_F u. \tag{5}$$

Then $\Delta$ is said to satisfy the $\alpha$-IQC defined by filter $F$ if

$$\int_0^T e^{\alpha t}(z(t)^\top z(t) - l(t)^\top l(t))dt \geq 0 \tag{6}$$

for all $T \geq 0$ and for $x_F(0) = 0_{n_F}$. We write this compactly as $\Delta \in \text{IQC}(F, \alpha)$. The $\alpha$-IQC can be used to describe uncertainties such as unknown delays or unmodeled actuators. It allows us to augment the system with $F$ and analyze the augmented system with $\Delta$ removed, treating $l$ as a free input.

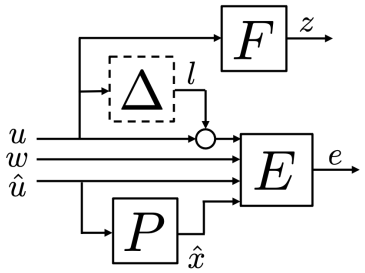


Fig. 2: Block diagram for the error system $E$. By analyzing the system augmented with the filter $F$, we can ignore $\Delta$ and treat $l$ as a free input, where $z$ and $l$ satisfy the $\alpha$-IQC (6).

There is a more general IQC framework where the filter has both $u$ and $l$ as inputs. In this paper, we restrict the filter input to be the tracker input $u$ so that there is no uncertainty affecting the filter state $x_F$. This allows us to compute $x_F$ and use it as an input to the tracking controller.

### C. Error System

We define a tracking error $e \in \mathbb{R}^{n_e}$ between the states of the planner and tracker models:

$$e = \phi(x, \hat{x}). \tag{7}$$

We assume $\phi$ is invertible in $x$ for all $\hat{x} \in \hat{\mathcal{X}}$, i.e., it admits an inverse, $\psi$, such that $e = \phi(x, \hat{x}) \Rightarrow x = \psi(e, \hat{x})$. We further assume the error dynamics can be written as

$$\dot{e} = f_e(e, \hat{x}, \hat{u}, w) + g_e(e, \hat{x}, \hat{u}, w)(u + l), \tag{8}$$
$$l = \Delta(u), \tag{9}$$

for some functions $f_e(e, \hat{x}, \hat{u}, w)$ and $g_e(e, \hat{x}, \hat{u}, w)$. Both of these requirements hold for the simple definition $\phi(x, \hat{x}) = x - \hat{x}$. They also hold for the example in Section V, where $\phi(x, \hat{x}) = r(\hat{x})(x - \hat{x})$, and $r(\hat{x}) \in \mathbb{R}^{n_x \times n_x}$ is invertible for all $\hat{x} \in \hat{\mathcal{X}}$. We denote this error system as $E$ in Figure 2.

### D. Objective

We wish to derive a tracking controller $u = \kappa(e, x_F, \hat{x}, \hat{u})$ and an associated error bound $\mathcal{O} \subseteq \mathbb{R}^{n_e}$, as small as possible, for the following closed loop error dynamics:

$$\dot{e} = f_e(e, \hat{x}, \hat{u}, w) + g_e(e, \hat{x}, \hat{u}, w)(\kappa(e, x_F, \hat{x}, \hat{u}) + l), \tag{10}$$
$$l = \Delta(\kappa(e, x_F, \hat{x}, \hat{u})). \tag{11}$$

$\mathcal{O}$ is used for shrinking the planner state constraint $\hat{\mathcal{X}}$ such that, when $e \in \mathcal{O}$, $\hat{x} \in \hat{\mathcal{X}}$ guarantees the tracker constraint $x \in \mathcal{X}$. Minimizing the volume of $\mathcal{O}$ ensures the constraint $\hat{\mathcal{X}}$ is minimally restrictive.

## III. ERROR BOUND

Finding the error bound $\mathcal{O}$ involves two conditions. First, in Section III-A, we present a condition on the control law $\kappa$ and an associated storage function $V$ such that the $\gamma$-sublevel set of $V$ for some scalar $\gamma$, denoted $\Omega(V, \gamma)$, is positively invariant under the closed loop error dynamics (10)-(11) for all $\Delta \in \text{IQC}(F, \alpha)$, $\hat{x} \in \hat{\mathcal{X}}$, $\hat{u} \in \hat{\mathcal{U}}$, and $w \in \mathcal{W}$. That is,

$$V(e(0), x_F(0)) \leq \gamma \Rightarrow V(e(t), x_F(t)) \leq \gamma \ \forall t \geq 0. \tag{12}$$

Next, in Section III-B, we project $\Omega(V,\gamma)$ into the space of error variables. $\mathcal{O}$ is a bound on this projection, obtained using a "shape function" with an adjustable parameter that we use to minimize the volume. We accommodate an initial condition constraint and an input constraint in Section III-C.

### A. Invariance Condition

The following condition ensures that the $\gamma$-sublevel set of a storage function $V$ is a positively invariant set for the closed loop error dynamics (10)-(11) with tracking control law $\kappa$. It is motivated by a related result on robust control barrier functions [22, Lemma 1].

**Theorem 1.** *Given the error dynamics (8)-(9), filter dynamics (4)-(5), and IQC $\Delta \in$ IQC$(F,\alpha)$, if there exist $\mathcal{C}^1$ functions $V : \mathbb{R}^{n_e} \times \mathbb{R}^{n_F} \to \mathbb{R}$ and $\kappa : \mathbb{R}^{n_e} \times \mathbb{R}^{n_F} \times \mathbb{R}^{\hat{n}} \times \mathbb{R}^{\hat{m}} \to \mathbb{R}$, and a scalar $\gamma > 0$ such that*

$$\begin{aligned}\nabla_e V(e,x_F)^\top &(f_e(e,\hat{x},\hat{u},w) \\ &+ g_e(e,\hat{x},\hat{u},w)(\kappa(e,x_F,\hat{x},\hat{u})+l)) \\ + \nabla_{x_F} V(e,x_F)^\top &(A_F x_F + B_F \cdot \kappa(e,x_F,\hat{x},\hat{u})) \\ + (C_F x_F + D_F \kappa(e,x_F,\hat{x},\hat{u}))^\top &(C_F x_F + D_F \kappa(e,x_F,\hat{x},\hat{u})) \\ &- l^\top l < -\alpha(V(e,x_F)-\gamma)\end{aligned} \tag{13}$$

$$\forall \hat{x} \in \hat{\mathcal{X}}, \hat{u} \in \hat{\mathcal{U}}, w \in \mathcal{W}, l \in \mathbb{R}^{n_l}, (e,x_F) \text{ s.t. } V(e,x_F) \le \gamma,$$

*then $\Omega(V,\gamma)$ is a positively invariant set for the closed loop error dynamics with controller $\kappa$.*

**Proof.** The dynamics of the augmented closed-loop system with state $\xi := [e; x_F]$, inputs $(l,\hat{x},\hat{u},w)$, and output $z$ are:

$$\begin{aligned}\dot{\xi} &= \begin{bmatrix} f_e(e,\hat{x},\hat{u},w) + g_e(e,\hat{x},\hat{u},w)(\kappa(e,x_F,\hat{x},\hat{u})+l) \\ A_F x_F + B_F \cdot \kappa(e,x_F,\hat{x},\hat{u}) \end{bmatrix} \\ &=: F(\xi,l,\hat{x},\hat{u},w),\end{aligned} \tag{14}$$

$$z = C_F x_F + D_F \kappa(e,x_F,\hat{x},\hat{u}) =: H(\xi,\hat{x},\hat{u}). \tag{15}$$

Note that condition (13) is equivalent to

$$\nabla V(\xi)^\top F + H^\top H - l^\top l < -\alpha(V(\xi)-\gamma). \tag{16}$$

For simplicity of notation, define $\bar{F}(t) := F(\xi(t),l(t),\hat{x}(t),\hat{u}(t),w(t))$, and similarly for $\bar{H}$. Define $\bar{V}(t) := V(\xi(t)) - \gamma$ so that $\bar{V}(t) = 0$ when $V(\xi(t)) = \gamma$.

Now the theorem is proved by contradiction. Assume there exist a time $t_3 > 0$ and signals $\xi(\cdot)$, $\hat{x}(\cdot)$, $\hat{u}(\cdot)$, $l(\cdot)$, and $w(\cdot)$ such that $\bar{V}(0) \le 0$ but $\bar{V}(t_3) > 0$. Then, by continuity of $\bar{V}$, there exists $t_1 \in [0,t_3)$ such that $\bar{V}(t_1) = 0$, $\bar{V}(t) \le 0$ for $t \le t_1$, and $\bar{V}(t) > 0$ for $t \in (t_1, t_1+\epsilon_1]$ for some $\epsilon_1 > 0$. Then the inequality (13) holds for $t \le t_1$. Furthermore, since (13) is a *strict* inequality, by continuity of $\bar{V}$ there exists $\epsilon_2 > 0$ such that for $t \in [0, t_1+\epsilon_2]$, the relaxed, non-strict version of (13) holds:

$$\nabla V(\xi(t))^\top \bar{F}(t) + \bar{H}(t)^\top \bar{H}(t) - l(t)^\top l(t) \le -\alpha \bar{V}(t). \tag{17}$$

Defining $t_2 = t_1 + \min\{\epsilon_1, \epsilon_2\}$, we see that (17) holds on $[0,t_2]$ and $\bar{V}(t_2) > 0$. Next, note that

$$\begin{aligned}\frac{d}{dt}\{e^{\alpha t}\bar{V}(t)\} &= e^{\alpha t}\{\alpha \bar{V}(t) + \nabla V(\xi(t))^\top \bar{F}(t)\} \\ &\le -e^{\alpha t}(\bar{H}(t)^\top \bar{H}(t) - l(t)^\top l(t)).\end{aligned} \tag{18}$$

Integrating (18) from $t = 0$ to $t = t_2$, we have

$$\begin{aligned}&e^{\alpha t_2}\bar{V}(t_2) - \bar{V}(0) \\ &\le -\int_0^{t_2} e^{\alpha t}(\bar{H}(t)^\top \bar{H}(t) - l(t)^\top l(t))dt \le 0,\end{aligned} \tag{19}$$

where the last inequality used (6). Rearranging, we have

$$\bar{V}(t_2) \le e^{-\alpha t_2}\bar{V}(0) \le 0, \tag{20}$$

which is a contradiction. □

### B. Projection Condition

If we find $V$ and $\kappa$ satisfying (13), then $\Omega(V,\gamma) \subseteq \mathbb{R}^{n_e} \times \mathbb{R}^{n_F}$ is a positively invariant set. Next we obtain an error bound $\mathcal{O} \subseteq \mathbb{R}^{n_e}$ by bounding the projection of $\Omega(V,\gamma)$ from the $(e,x_F)$-space into the $e$-space $\mathbb{R}^{n_e}$. $\mathcal{O}$ will be used to shrink planning state constraints $\hat{\mathcal{X}}$ such that $\hat{x} \in \hat{\mathcal{X}}$ implies $x \in \mathcal{X}$. Then safety of the tracker system is guaranteed as long as the planner constraints are satisfied. We would like to obtain an error bound $\mathcal{O}$ that is as small as possible so that the constraints on the planner are minimally restrictive.

We introduce a "shape function" $P : \mathbb{R}^{n_e} \to \mathbb{R}$, whose sublevel sets are regular objects like balls or hyper-rectangles that can be conveniently used to shrink state constraints. $P$ need not depend on all error variables (e.g., if the only constraint in $\mathcal{X}$ is for obstacle avoidance, $P$ may just depend on the position errors). Then we enforce the constraint

$$\text{proj}_{\mathbb{R}^{n_e}}(\Omega(V,\gamma)) \subseteq \Omega(P,c) =: \mathcal{O}, \tag{21}$$

i.e., the projection onto $\mathbb{R}^{n_e}$ of the $\gamma$-sublevel set of $V$ is contained within the $c$-sublevel set of $P$. By minimizing $c$, we can shrink the error bound $\mathcal{O}$ as much as possible.

### C. Additional Conditions

There are two additional conditions that we may want the storage function $V$ and the tracking controller $\kappa$ to satisfy. First, the initial error $e(0)$ may be known to lie in a set $\mathcal{E}_0 \subseteq \mathbb{R}^{n_e}$. To ensure this set is included in the invariant set, we can enforce the constraint

$$\mathcal{E}_0 \times \{0_{n_F}\} \subseteq \Omega(V,\gamma), \tag{22}$$

where we used the fact that the filter initial condition is $x_F(0) = 0_{n_F}$. Secondly, we can enforce input constraints $\mathcal{U}$ with the following constraint:

$$\begin{aligned}&\kappa(e,x_F,\hat{x},\hat{u}) \in \mathcal{U} \quad \forall \hat{x} \in \hat{\mathcal{X}},\ \hat{u} \in \hat{\mathcal{U}}, \\ &\forall (e,x_F) \in \mathbb{R}^{n_e} \times \mathbb{R}^{n_F} \text{ s.t. } V(e,x_F) \le \gamma.\end{aligned} \tag{23}$$

## IV. SOS OPTIMIZATION

We now formulate an SOS optimization that searches for a tracking controller $\kappa$ and a storage function $V$ satisfying (13) and (21)-(23) while minimizing the volume of the error bound $\mathcal{O}$.

Finding generic functions $V$ and $\kappa$ that satisfy (13) is a difficult problem. Below we show how SOS programming can be used to search for these functions by restricting to polynomial candidates $V \in \mathbb{R}[e,x_F]$ and $\kappa \in \mathbb{R}^{n_u}[e,x_F,\hat{x},\hat{u}]$. Besides this restriction, we make the following assumption:

**Assumption 1.** *The mappings $f_e \in \mathbb{R}^{n_x}[(e,\hat{x},\hat{u},w)]$ and $g_e \in \mathbb{R}^{n_x \times n_u}[(e,\hat{x},w)]$ in (8) are polynomials. Sets $\mathcal{E}_0$, $\hat{\mathcal{X}}$, $\hat{\mathcal{U}}$, and $\mathcal{W}$ are semi-algebraic sets, i.e., there exists $p_0 \in \mathbb{R}[e]$ such that $\mathcal{E}_0 = \{e \in \mathbb{R}^{n_x} : p_0(e) \leq 0\}$; with similar definitions for $\hat{\mathcal{X}}$, $\hat{\mathcal{U}}$, and $\mathcal{W}$ with polynomials $p_{\hat{x}} \in \mathbb{R}[\hat{x}]$, $p_{\hat{u}} \in \mathbb{R}[\hat{u}]$, and $p_w \in \mathbb{R}[w]$. The control constraint set $\mathcal{U}$ is a hypercube $\mathcal{U} = \{u \in \mathbb{R}^{n_u} : \underline{u} \leq u \leq \overline{u}\}$, where $\underline{u}, \overline{u} \in \mathbb{R}^{n_u}$.*

The invariance condition (13) involves the unbounded variable $l$ and contains a term that is quadratic in the decision variable $\kappa$. The following lemma gives a sufficient condition for (13) that is suitable for use in an SOS program.

**Lemma 1.** *A sufficient condition for (13) is*

$$-\mathcal{Q}(V,\kappa,\gamma,s_V,s_X,s_U,s_W) \in \Sigma_{2n_u+1}[e,x_F,\hat{x},\hat{u},w], \quad (24)$$

$$\text{where } \mathcal{Q} = \begin{bmatrix} \mathcal{Q}_{11} & \nabla_e V^\top g & (C_F x_F + D_F\kappa)^\top \\ g^\top \nabla_e V & -4I & 0 \\ C_F x_F + D_F \kappa & 0 & -I \end{bmatrix}, \quad (25)$$

$$\begin{aligned} \text{and } \mathcal{Q}_{11}(V,\kappa,\gamma,s_V,s_X,s_U,s_W) &= \nabla_e V^\top (f_e + g_e \cdot \kappa) \\ &+ \nabla_{x_F} V^\top (A_F x_F + B_F \kappa) + (\alpha - s_V)\cdot(V-\gamma) \\ &- s_X \cdot p_X - s_U \cdot p_U - s_W \cdot p_W + \epsilon, \qquad \epsilon > 0. \quad (26) \end{aligned}$$

**Proof.** We manipulate (13), reproduced below, into an equivalent condition without quadratic terms in decision variables:

$$\begin{aligned} &\nabla_e V^\top (f_e + g_e \cdot (k+l)) + \nabla_{x_F} V^\top (A_F x_F + B_F \kappa) \quad (27) \\ &+ (C_F x_F + D_F\kappa)^\top (C_F x_F + D_F \kappa) - l^\top l < -\alpha(V-\gamma) \\ &\forall \hat{x} \in \hat{\mathcal{X}}, \hat{u} \in \hat{\mathcal{U}}, w \in \mathcal{W}, l \in \mathbb{R}^{n_l}, (e,x_F) \text{ s.t. } V(e,x_F) \leq \gamma. \end{aligned}$$

Maximizing over the unconstrained variable $l$ gives a worst-case value of $l^* = \frac{1}{2} g_e^\top \nabla_e V$. Plugging this in yields

$$\begin{aligned} &\nabla_e V^\top (f_e + g_e \cdot \kappa) + \nabla_{x_F} V^\top (A_F x_F + B_F \kappa) \quad (28) \\ &\qquad + (C_F x_F + D_F\kappa)^\top (C_F x_F + D_F \kappa) \\ &\qquad + \frac{1}{4} \nabla_e V^\top g_e g_e^\top \nabla_e V < -\alpha(V-\gamma) \\ &\forall \hat{x} \in \hat{\mathcal{X}},\ \hat{u} \in \hat{\mathcal{U}},\ w \in \mathcal{W},\ (e,x_F) \text{ s.t. } V(e,x_F) \leq \gamma. \end{aligned}$$

While (28) only needs to hold for certain values of $(e,\hat{x},\hat{u},w)$, the S-procedure [27] is a method that gives a sufficient condition that holds for all $(e,\hat{x},\hat{u},w)$, where constraints such as $\hat{x} \in \hat{\mathcal{X}}$ are encoded via nonnegative multipliers. We now use the S-procedure to ensure that (28) holds whenever $V \leq \gamma$, $\hat{x} \in \hat{\mathcal{X}}$, $\hat{u} \in \hat{\mathcal{U}}$, and $w \in \mathcal{W}$, and add $\epsilon > 0$ to the left hand side so the inequality is non-strict:

$$\begin{aligned} &\nabla_e V^\top (f_e + g_e \cdot \kappa) + \nabla_{x_F} V^\top (A_F x_F + B_F \kappa) \quad (29) \\ &\qquad + (C_F x_F + D_F\kappa)^\top (C_F x_F + D_F \kappa) \\ &\qquad + \frac{1}{4} \nabla_e V^\top g_e g_e^\top \nabla_e V + (\alpha - s_V)(V-\gamma) \\ &\qquad - s_X \cdot p_X - s_U \cdot p_U - s_W \cdot p_W + \epsilon \leq 0, \end{aligned}$$

where $s_V$, $s_X$, $s_U$, and $s_W$ are nonnegative multipliers. Note that (29) has quadratic terms in $\nabla_e V$ and $\kappa$ which are both decision variables, so (29) is not convex in either variable. Thus, start by applying Schur complements to expand the term $\frac{1}{4} \nabla_e V^\top g_e g_e^\top \nabla_e V$ that is quadratic in $\nabla_e V$. Then (29) is equivalent to

$$\begin{bmatrix} \mathcal{Q}_{11} + (C_F x_F + D_F\kappa)^\top (C_F x_F + D_F \kappa) & \nabla_e V^\top g \\ g^\top \nabla_e V & -4I \end{bmatrix} \leq 0, \quad (30)$$

where $\mathcal{Q}_{11}$ is defined in (26). Applying Schur complements a second time to expand the term $(C_F x_F + D_F\kappa)^\top (C_F x_F + D_F\kappa)$ that is quadratic in $\kappa$, (30) is equivalent to

$$\mathcal{Q}(V,\kappa,\gamma,s_V,s_X,s_U,s_W) \leq 0, \quad (31)$$

with $\mathcal{Q}$ as in (25)-(26). Finally, (31) can be relaxed as the SOS condition (24). □

Using Lemma 1 and applying the generalized S-procedure [27] to the set containment constraints (21)-(23), we obtain the following SOS optimization problem for finding $V$, $\kappa$, and $\mathcal{O}$:

$$\begin{aligned} \min_{\substack{V,\kappa,\gamma,s_0,s_F \\ s_V^i,s_X^i,s_U^i,s_W \\ c^{-1}>0,\gamma>0}} \quad & \beta\delta - c^{-1} && \text{(32a)} \\ \text{s.t. } & s_W, s_0 \in \Sigma[e,x_F,\hat{x},\hat{u}] && \text{(32b)} \\ & s_V^i, s_X^i, s_U^i \in \Sigma[e,x_F,\hat{x},\hat{u}],\ i \in \{-n_u,...,n_u\} && \text{(32c)} \\ & \delta(Z^\top Z)I - \mathcal{Q}(V,\kappa,\gamma,s_V^0,s_X^0,s_U^0,s_W) && \text{(32d)} \\ & \qquad \in \Sigma_{2n_u+1}[e,x_F,\hat{x},\hat{u}] \\ & (V-\gamma) - (c^{-1}P - 1) \in \Sigma[e,x_F] && \text{(32e)} \\ & s_0 \cdot p_0 + s_F \cdot x_F - (V-\gamma) \in \Sigma[e,x_F] && \text{(32f)} \\ & \overline{u}_i - \kappa_i + s_V^i \cdot (V-\gamma) + s_X^i \cdot p_{\hat{x}} && \text{(32g)} \\ & \qquad + s_U^i \cdot p_{\hat{u}} \in \Sigma[(e,\hat{x},\hat{u})],\ i \in \{1,...,n_u\} \\ & \kappa_i - \underline{u}_i + s_V^{-i} \cdot (V-\gamma) + s_X^{-i} \cdot p_{\hat{x}} && \text{(32h)} \\ & \qquad + s_U^{-i} \cdot p_{\hat{u}} \in \Sigma[(e,\hat{x},\hat{u})],\ i \in \{1,...,n_u\} \end{aligned}$$

The variables in (32b) and (32c) are SOS multipliers. Condition (32d) is the result of applying the S-procedure to (13) (justified in Lemma 1), with an added slack variable $\delta$ multiplying $(Z^\top Z)I$, where $Z$ is the vector of monomials in $\mathcal{Q}$. Condition (32e) is the result of applying the S-procedure to (21). Condition (32f) is the result of applying the S-procedure to (22), where the multiplier $s_F$ need not be SOS. Conditions (32g) and (32h) are the result of applying the S-procedure to (23) at each vertex of $\mathcal{U}$. The cost function (32a) is a weighted difference of the slack variable $\delta$ from (32d) and the parameter $c^{-1}$ from (32e) with a weight $\beta$ that the user can tune. This cost helps to achieve the joint goals of having the invariance condition (13) hold up to numerical tolerances, and making the error bound $\mathcal{O}$ as small as possible by maximizing $c^{-1}$ (i.e., minimizing $c$). We found that in practice, formulating the problem in terms of $c^{-1}$ rather than $c$ led to a smaller error bound.

Even after removing quadratic terms in $\nabla_e V$ and $\kappa$ in Lemma 1, the SOS program (32) is nonconvex because it has terms that are bilinear in the decision variables, but it can be solved by alternately solving convex subproblems with the decision variables $(V,\gamma,\delta,c,s_0,s_F,s_X^i,s_U^i,s_W)$ and $(\kappa,\delta,c,s_0,s_F,s_V^i,s_X^i,s_U^i,s_W)$.

## V. NUMERICAL EXAMPLE

We demonstrate the method on an obstacle avoidanc example, where the planner uses a Dubin's vehicle model:

$$\dot{\hat{x}} = \begin{bmatrix} \hat{u}_2 \cos(\hat{x}_3) \\ \hat{u}_2 \sin(\hat{x}_3) \\ \hat{u}_1 \end{bmatrix}. \quad (33$$

The states are positions $(\hat{x}_1, \hat{x}_2)$ and heading angle $\hat{x}_3$, an the inputs are angular rate $\hat{u}_1$ and longitudinal velocity $\hat{u}_2$

In the tracker model, the input $u_1$ is delayed by $\tau$ seconds denoted $D_\tau(u_1)$. Using an uncertain block $\Delta$, the tracke model can be put into the form of (2)-(3):

$$\dot{x} = \begin{bmatrix} u_2 \cos(x_3) \\ u_2 \sin(x_3) \\ D_\tau(u_1) \end{bmatrix} = \begin{bmatrix} u_2 \cos(x_3) \\ u_2 \sin(x_3) \\ u_1 + l \end{bmatrix}, \quad l = \Delta(u_1). \quad (34)$$

Thus we have $\Delta(u) = D_\tau(u_1) - u_1$, so $\Delta$ is the *deviation* due to the delay. The value of the delay is unknown but is known to lie in the interval $\tau \in [0, \tau_{\max}]$. The $\alpha$-IQC associated with $\Delta$ is computed as in Section V of [22], using the MATLAB function `fitmagfrd` to obtain a filter $F$ whose Bode plot upper bounds the Bode plot of all possible values of $\Delta$.

For simplicity, there is no exogenous disturbance $w$ in this example. We define the error as

$$e = R(\hat{x}_3)^\top (x - \hat{x}), \text{ where } R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (35)$$

Rotating the error into the frame of the planner model gives:

$$\dot{e} = \begin{bmatrix} \hat{u}_1 e_2 + u_2 \cos(e_3) - \hat{u}_2 \\ -\hat{u}_1 e_1 + u_2 \sin(e_3) \\ u_1 + l - \hat{u}_1 \end{bmatrix}, \quad l = \Delta(u_1), \quad (36)$$

and the sin/cos terms can be approximated as polynomials using a second order Taylor series expansion about $e_3 = 0$.

### A. SOS Tracking Controller

We solve the SOS program (32) with $\alpha = 5$,$\tau_{\max} = 0.05$s, $P(e) = e_1^2 + e_2^2$, $\hat{\mathcal{U}} = \{\hat{u} \in \mathbb{R}^2 : \|u\|_2^2 \le 1\}$, and $\mathcal{U} = \mathbb{R}^2$. As can be seen in the SOS cost function (32a), larger $\beta$ encourages a small value of $\delta$ and smaller $\beta$ encourages a smaller value of $c$. After searching over several values, $\beta = 1e3$ was selected because it gave the smallest error bound while still yielding a $\delta$ value on the order of $1e{-}4$ which was approximate magnitude of the numerical tolerance from the SOS solver. Since the dynamics (36) don't depend on $\hat{\mathcal{X}}$, we don't need to select $\hat{\mathcal{X}}$ before solving the SOS program. We use the SOSTOOLS toolbox [28] in MATLAB with solver MOSEK [29], and after 15 iterations (28 minutes):

$$\begin{aligned} &\gamma = 1, \ \delta = 7 \times 10^{-4}, \ c = 10.63 \quad (37) \\ &V = 0.30e_1^2 + 0.75e_1x_F + 0.13e_2^2 + 0.23e_3^2 + 0.70x_F^2 \\ &k_1 = 0.61e_1e_2 - 0.07e_2\hat{u}_2 - 0.28e_2x_F - 5.76e_3 + 0.82\hat{u}_1 \\ &k_2 = -0.001e_2\hat{u}_1 - 0.67e_1 + 0.11\hat{u}_2 + 2.01x_F. \end{aligned}$$

Terms with coefficient magnitudes less than 1e-4 have been omitted. The value $c = 10.63$ indicates an error bound radius of $\sqrt{c} = 3.26$. Hence, the planner state constraints will be shrunk by the set $\mathcal{O} = \{e \in \mathbb{R}^3 : e_1^2 + e_2^2 \le 3.26^2\}$.

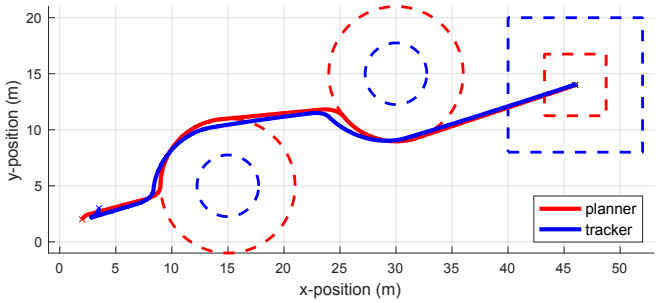


Fig. 3: Planner-tracker simulation. The planner trajectory in red avoids the bloated obstacle (dashed red circles) and reaches the shrunk target region (dashed red square). The tracker trajectory in blue tracks the planner and avoids the true obstacles (dashed blue circles) and reaches the true target region (dashed blue square).

### B. MPC Planning Controller

MPC is used as the planning controller to generate a motion plan for the vehicle that will avoid circular obstacles (bloated by the error bound) and reach a target region (shrunk by the error bound). At each time step, the following optimization is solved. We apply the input $\hat{u}_t$ at the current time step and we resolve the optimization at the next step.

$$\begin{aligned} \min_{u_t, \ldots, u_{t+T-1}} \ & \sum_{k=t}^{t+T-1} (\hat{u}_k^\top R \hat{u}_k + (\hat{x}_k - \hat{x}_{\text{des}})^\top Q (\hat{x}_k - \hat{x}_{\text{des}})) \quad (38a) \\ & + (\hat{x}_{t+T} - \hat{x}_{\text{des}})^\top Q_T (\hat{x}_{t+T} - \hat{x}_{\text{des}}) \\ \text{s.t. } & \forall k \in \{t, \ldots, t+T-1\}: \\ & \hat{x}_{k+1} = \hat{x}_k + dt \cdot \hat{f}(x_k, u_k), \quad (38b) \\ & \hat{x}_k \in \hat{\mathcal{X}} = \mathcal{X} \ominus \mathcal{O}, \quad (38c) \\ & \hat{u}_k \in \hat{\mathcal{U}}, \quad (38d) \\ & \hat{x}_{t+T} \in \hat{\mathcal{X}} = \mathcal{X} \ominus \mathcal{O}, \quad (38e) \\ & \hat{x}_t = \hat{x}(dt \cdot t) \quad (38f) \end{aligned}$$

The cost (38a) penalizes the control input and the distance from the desired final point $\hat{x}_{\text{des}}$ at the center of the target set. Constraint (38b) is the forward Euler discretized planner dynamics, and (38c) and (38e) are planner state constraints, which are tracker obstacle avoidance constraints shrunk by the error bound $\mathcal{O}$. Constraint (38d) restricts the input, and (38f) ensures that the optimization starts from the current planner state. We solve the optimization using the toolbox YALMIP [30] in MATLAB and the solver Ipopt [31], with parameters $T = 20$, $dt = 0.1$s, $R = \frac{1}{2}I_2$, $Q = 10I_3$, $Q = 100I_3$, and $\hat{x}_{\text{des}} = [46\text{m}; 14\text{m}]$. $\mathcal{X} = \{x \in \mathbb{R}^3 : (x_{1:2} - x_{\text{obs},j})^2 \ge r_{\text{obs},j}^2, j = 1, 2\}$, $x_{\text{obs},1} = [15\text{m}; 5\text{m}]$, $x_{\text{obs},2} = [30\text{m}; 15\text{m}]$, $r_{\text{obs},1} = r_{\text{obs},2} = 2.74$m.

### C. Results

We simulate the combined planning and tracking control system for the scenario shown in Figure 3. The planner trajectory in red avoids the bloated obstacles and reaches the shrunk target region. The tracker trajectory in blue tracks the planner trajectory closely; it does intersect with the bloated obstacles, but not with the true obstacles, preserving safety. It also reaches the target region. Figure 4 shows the evolution

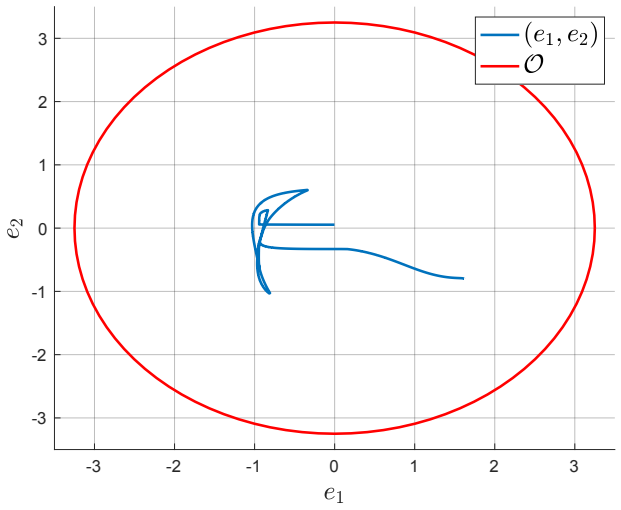


Fig. 4: Position error $(e_1, e_2)$ and the error bound $\mathcal{O}$.

of the error in $(e_1, e_2)$ space and confirms that the error never leaves the bound $\mathcal{O}$. Note that while $\Omega(V, \gamma)$ is invariant and its projection lies in $\mathcal{O}$, $\mathcal{O}$ itself is not necessarily invariant.

## VI. CONCLUSION

This paper used $\alpha$-IQCs to extend the planner-tracker framework to accommodate unmodeled dynamics at the input of the tracker model. An SOS program was formulated to search for a tracking controller $\kappa$ and an error bound $\mathcal{O}$. The method was demonstrated on a vehicle obstacle avoidance example with an MPC planner and an input delay in the tracker model. Future work includes formally accounting for the planner discretization error and the error from the polynomial approximation of trigonometric terms.

## REFERENCES


[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 187–210, 2018.

[2] N. Palumbo, R. Blauwkamp, and J. Lloyd, "Modern homing missile guidance theory and techniques," *Johns Hopkins APL Technical Digest*, vol. 29, no. 1, pp. 42–59, 2010.

[3] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[4] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "FaSTrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE Conference on Decision and Control (CDC)*, pp. 1517–1522, Dec. 2017.

[5] S. Singh, A. Majumdar, J. Slotine, and M. Pavone, "Robust online motion planning via contraction theory and convex optimization," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5883–5890, May 2017.

[6] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan, "Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots," *International Journal of Robotics Research*, vol. 39, no. 12, pp. 1419–1469, 2020.

[7] S. Singh, M. Chen, S. L. Herbert, C. J. Tomlin, and M. Pavone, "Robust tracking with model mismatch for fast and safe planning: An SOS optimization approach," in *Algorithmic Foundations of Robotics XIII* (M. Morales, L. Tapia, G. Sánchez-Ante, and S. Hutchinson, eds.), (Cham), pp. 545–564, Springer International Publishing, 2020.

[8] S. W. Smith, H. Yin, and M. Arcak, "Continuous abstraction of nonlinear systems using sum-of-squares programming," in *2019 IEEE 58th Conference on Decision and Control*, pp. 8093–8098, 2019.

[9] U. Rosolia and A. D. Ames, "Multi-rate control design leveraging control barrier functions and model predictive control policies," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 1007–1012, 2021.

[10] K. S. Schweidel, H. Yin, S. W. Smith, and M. Arcak, "Safe-by-design planner–tracker synthesis with a hierarchy of system models," *Annual Reviews in Control*, vol. 53, pp. 138–146, 2022.

[11] L. Chisci, J. A. Rossiter, and G. Zappa, "Systems with persistent disturbances: predictive control with restricted constraints," *Automatica*, vol. 37, no. 7, pp. 1019–1028, 2001.

[12] W. Langson, I. Chryssochoos, S. V. Rakovic, and D. Q. Mayne, "Robust model predictive control using tubes," *Automatica*, vol. 40, pp. 125–133, 2004.

[13] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523–533, 2006.

[14] S. V. Raković, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2746–2761, 2012.

[15] S. V. Raković and Q. Cheng, "Homothetic tube MPC for constrained linear difference inclusions," in *Chinese Control and Decision Conference (CCDC)*, pp. 754–761, IEEE, 2013.

[16] D. Muñoz-Carpintero, M. Cannon, and B. Kouvaritakis, "Recursively feasible robust MPC for linear systems with additive and multiplicative uncertainty using optimized polytopic dynamics," in *Conference on Decision and Control (CDC)*, pp. 1101–1106, IEEE, 2013.

[17] J. Fleming, B. Kouvaritakis, and M. Cannon, "Robust tube MPC for linear systems with multiplicative uncertainty," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1087–1092, 2014.

[18] S. V. Raković, W. S. Levine, and B. Açikmese, "Elastic tube model predictive control," in *American Control Conference (ACC)*, pp. 3594–3599, IEEE, 2016.

[19] M. Bujarbaruah, U. Rosolia, Y. R. Stürz, X. Zhang, and F. Borrelli, "Robust MPC for linear systems with parametric and additive uncertainty: A novel constraint tightening approach," *arXiv preprint arXiv:2007.00930*, 2020.

[20] A. Megretski and A. Rantzer, "System analysis via integral quadratic constraints," *IEEE Transactions on Automatic Control*, vol. 42, no. 6, pp. 819–830, 1997.

[21] P. Seiler, R. M. Moore, C. Meissen, M. Arcak, and A. Packard, "Finite horizon robustness analysis of LTV systems using integral quadratic constraints," *Automatica*, vol. 100, pp. 135–143, 2019.

[22] P. Seiler, M. Jankovic, and E. Hellstrom, "Control barrier functions with unmodeled input dynamics using integral quadratic constraints," *IEEE Control Systems Letters*, vol. 6, pp. 1664–1669, 2022.

[23] L. Schwenkel, J. Köhler, M. A. Müller, and F. Allgöwer, "Dynamic uncertainties in model predictive control: guaranteed stability for constrained linear systems," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 1235–1241, 2020.

[24] L. Schwenkel, J. Köhler, M. A. Müller, and F. Allgöwer, "Model predictive control for linear uncertain systems using integral quadratic constraints," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 355–368, 2023.

[25] K. Astrom, C. Hang, and B. Lim, "A new Smith predictor for controlling a process with an integrator and long dead-time," *IEEE Transactions on Automatic Control*, vol. 39, no. 2, pp. 343–345, 1994.

[26] C. K. Verginis, D. V. Dimarogonas, and L. E. Kavraki, "KDF: Kinodynamic motion planning via geometric sampling-based algorithms and funnel control," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 978–997, 2023.

[27] P. Parrilo, "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization," *PhD thesis, California Institute of Technology*, 2000.

[28] A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Prajna, P. Seiler, P. A. Parrilo, M. M. Peet, and D. Jagt, *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*. `http://arxiv.org/abs/1310.4716`, 2021.

[29] *The MOSEK optimization toolbox for MATLAB manual, 9.0.*, 2019.

[30] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in matlab," in *In Proceedings of the CACSD Conference*, 2004.

[31] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.