Robust Collision Avoidance via Sliding Control

Robust Collision Avoidance via Sliding Control

Brett T. Lopez, Jean-Jacques Slotine, and Jonathan P. How

***Abstract*— Recent advances in perception and planning algorithms have enabled robots to navigate autonomously through unknown, cluttered environments at high-speeds. A key component of these systems is the ability to identify, select, and execute a safe trajectory around obstacles. Many of these systems, however, lack performance guarantees because model uncertainty and external disturbances are ignored when a trajectory is selected for execution. This work leverages results from nonlinear control theory to establish a bound on tracking performance that can be used to select a provably safe trajectory. The Composite Adaptive Sliding Controller (CASC) provides robustness to disturbances and reduces model uncertainty through high-rate parameter estimation. CASC is demonstrated in simulation and hardware to significantly improve the performance of a quadrotor navigating through unknown environments with external disturbances and unknown model parameters.**

I. INTRODUCTION

High-speed flight in unknown environments requires a tight integration of perception, path planning, and control to ensure that fast decisions are made and executed when new observations of the environment are available. Recent advances in perception and path planning algorithms have enabled robots to navigate quickly in unknown environments, but many of these systems rely on PID or feedback linearization for the trajectory tracking component because of the simplicity in their implementation. However, real-world scenarios that require high-precision control are likely to cause these techniques to fail because of their sensitivity to model uncertainty and external disturbances. This article presents a nonlinear controller whose performance guarantees can be used to establish *provably safe* (i.e. robust) trajectory following even with model uncertainty and external disturbances.

Robust path planning accounts for model uncertainty and external disturbances by synthesizing a feedback control law that ensures the robot remains in a "tube" [1] around the desired path. Reachable sets, Lyapunov functions, and model predictive control (MPC) have all been used to synthesize tubes for a variety of dynamical systems. However, many of the existing techniques (a) cannot be used on systems with nonlinear dynamics, (b) require expensive online/offline optimization, or (c) are highly dependent on the type of trajectory planner used. Additionally, since the desired level of tracking precision is specified *a priori*, controllers must be synthesized for multiple different operating conditions. Our approach explicitly accounts for nonlinear dynamics, can be used with any trajectory planner, requires no online/offline optimization, and can dynamically change its trajectory tracking precision by incorporating information from a perception or trajectory planning system.

Lopez and How are with the Aerospace Controls Laboratory, Dept. Aeronautics and Astronautics, MIT, USA. Slotine is with the Nonlinear Systems Laboratory, Dept. Mechanical Engineering, MIT, USA. btlopez, jjs, jhow@mit.edu

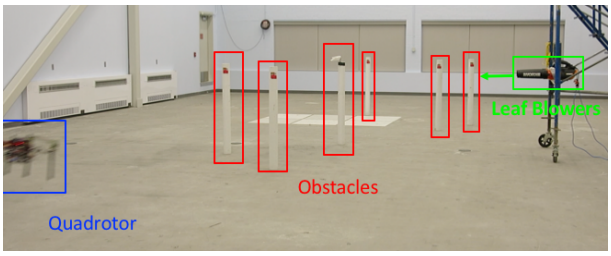


Fig. 1. Quadrotor safely navigating in unknown environment using a Composite Adaptive Sliding Controller (CASC) to estimate model parameters and reject external disturbance (generated by two leaf blowers).

The main contribution of this work is the analysis and experimental verification of a Composite Adaptive Sliding Controller (CASC) [2] for robust trajectory following. CASC leverages theoretical results from sliding control to obtain bounds on tracking error in the presence of disturbances. High-rate online parameter estimation via composite (or combined) adaptation is employed to reduce model uncertainty and maximize disturbance rejection. In addition, CASC uses perception data to dynamically change its tracking precision to match the operating environment of the robot – leading to a unique contribution of the work in that the controller is tuned online using perception information. CASC, used as an ancillary controller for a low-latency receding horizon trajectory planner [3], is demonstrated in simulation and hardware to enable robust path planning for a quadrotor navigating through an unknown environment with unknown model parameters and bounded disturbances. The results show that CASC significantly increases the likelihood of successfully navigating through cluttered environments with model uncertainty and external disturbances when compared to trajectory tracking with PID controller – a direct consequence of the robustness properties of CASC.

A second contribution of this work is the release of an open source flight stack for Qualcomm's Snapdragon Flight platform that includes a nonlinear observer, attitude controller, and motor controller interface. The code can be found at https://bitbucket.org/brettlopez/snap

II. RELATED WORK

Robust planning can be classified as a subset of robust MPC (RMPC). Standard MPC generates a sequence of control actions through an online optimization by assuming the system's dynamics are fully known and the complete absence of disturbances. However, the model uncertainty and disturbances that exist in real physical systems can reduce the overall performance of MPC. Reduced performance is

particularly problematic for path and motion planning (i.e. causing collisions with environment). RMPC attempts to address these shortcomings by directly considering model/state uncertainty and external disturbances.

The two RMPC strategies are: 1) a min-max formulation and 2) tube MPC [4]. In the min-max framework, *open-loop* MPC considers the worst-case disturbance in the optimization. However, this assumption leads to overly conservative solutions and can make the optimization infeasible. Alternatively, *feedback* MPC produces less conservative solutions by optimizing over control policies instead of control actions at the expense of computation complexity; often making this approach impractical for real system. Tube MPC [1], [5], [6] addresses the aforementioned issues by designing an ancillary controller that keeps the system's trajectory inside an invariant tube around the nominal trajectory. Thus, RMPC is decomposed into solving the standard MPC problem and designing the ancillary controller. This decomposition reduces computational complexity without sacrificing performance making it practical for real-time systems.

Path planning for nonlinear systems using tube MPC has begun to emerge in the literature with advancements in nonlinear control synthesis methods. Some of the first attempts to incorporate tube MPC ideas into path planning involved using linear reachability theory to construct tubes for nonlinear systems [7]. This approach was used to verify safe trajectories online for ground [8] and aerial vehicles [9]. Although [8], [9] were the first to successfully demonstrate the approach for nonlinear systems online, linearizing the dynamics and treating nonlinearities as bounded disturbances lead to overly conservative tubes. [10] explicitly considered nonlinear dynamics when constructing ancillary controllers via sum-of-squares (SOS) optimization that minimized the funnel size (akin to a tube). Simulation and experimental results for a variety of systems demonstrated the versatility of the approach. However, the method required a pre-specified trajectory library making it highly dependent on the choice of path/motion planner. Further, the overall performance of the system can be significantly reduced if the trajectories are not carefully selected. Lastly, the offline computation phase was extremely time consuming (e.g. 20-25 minutes per trajectory). [11] presented a novel approach that generated controllers offline via control contraction metrics [12] (based on Contraction Theory [13]). While this approach is completely independent of the planning strategy, the differential nature of the controller requires an online optimization to generate a geodesic connecting the current and desired state. Performing this optimization might not be feasible for computationally constrained systems.

This work proposes using sliding control techniques to synthesize a nonlinear ancillary controller. Other works have proposed a similar strategy (see for instance [14]) but not in the context of robust path planning. The advantage of using sliding control is that the controller can be synthesized independent of the planner and requires no online/offline optimization. Contrary to sliding mode control, sliding control achieves robustness though continuous control thereby eliminating the undesirable chattering phenomenon synonymous with sliding mode control.

III. Problem Formulation

This article is concerned with the underactuated and control affine nonlinear dynamical system [15] whose dynamics can be expressed as

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + b(\boldsymbol{x})\,\boldsymbol{u} + B_d\boldsymbol{d}, \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u} \in \mathbb{R}^m$ is the control input, and $\boldsymbol{d} \in \mathbb{R}^{n_d}$ is the disturbance. Note the time argument is omitted for clarity.

Assumption 1: Assume $f(\cdot)$ is unknown but can be decomposed into $f = \bar{f} + \Delta f$ where $\bar{f}$ is the nominal dynamics and Δf is unknown but bounded (i.e. $|\Delta f| < F$). Similarly, $b(\cdot)$ is of known sign and bounded (i.e. $0 < b_{\min} < b < b_{\max}$).

Assumption 2: Assume disturbance $\boldsymbol{d}$ has the properties:

1) $\boldsymbol{d} \in \mathcal{D} := \{\boldsymbol{d} \in \mathbb{R}^{n_d} : \ ||\boldsymbol{d}|| \leq D \ \forall t\}$
2) $B_d\boldsymbol{d} \in Span\{b(\boldsymbol{x})\}$

The second property is the "matching condition" that ensures the controller can compensate for the disturbance.

State and control input constraints, arising from obstacle locations and physical limitation of the robot, are imposed by requiring $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{u} \in \mathcal{U}$ for all time. The path planning problem is defined as finding a control policy $\boldsymbol{\pi} : \mathcal{X} \times \mathbb{R} \to \mathcal{U}$ such that $\boldsymbol{x}$ is driven to a goal configuration $\mathcal{X}_{goal} \subset \mathcal{X}$. The planning problem can be formulated as the following optimization problem:

P1: Planning Optimization Problem –

$$\begin{aligned} \underset{\boldsymbol{\pi} \in \Pi(\mathcal{X},\mathcal{U})}{\text{minimize}} \quad & J_T(\boldsymbol{x}, \boldsymbol{\pi}(\boldsymbol{x})) = \int_0^T \ell(\boldsymbol{x}, \boldsymbol{\pi}(\boldsymbol{x}))\, dt \\ \text{subject to} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}) + b(\boldsymbol{x})\,\boldsymbol{u} + B_d\boldsymbol{d} \\ & \boldsymbol{x} \in \mathcal{X}, \quad \boldsymbol{\pi} \in \mathcal{U}, \quad \boldsymbol{d} \in \mathcal{D} \\ & \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad \boldsymbol{x} \in \mathcal{X}_{goal}, \end{aligned}$$

where $\boldsymbol{x}_0$ is the initial state, $\ell(\cdot,\cdot)$ is a quadratic cost function, T is the time for the state to reach the goal configuration $\mathcal{X}_{goal}$, and $\Pi(\mathcal{X},\mathcal{U})$ is the set of all feasible control policies.

P1 is an optimization over all possible control policies and is thus difficult to solve. The standard approach taken in tube MPC [4] is to decompose **P1** into 1) a deterministic MPC optimization problem and 2) a feedback control synthesis problem. The unperturbed dynamics are used in generating an open-loop control input $\bar{\boldsymbol{u}}$ and nominal state trajectory $\bar{\boldsymbol{x}}$ and a control policy that keeps the state $\boldsymbol{x}$ close to the nominal trajectory $\bar{\boldsymbol{x}}$ is synthesized. To ensure tractability, the following assumption is made:

Assumption 3: Assume the control policy $\boldsymbol{u}$ takes the form $\boldsymbol{u} = \bar{\boldsymbol{u}} + \kappa(\boldsymbol{x}, \bar{\boldsymbol{x}})$, where $\bar{\boldsymbol{u}}$ is the open-loop control input and $\bar{\boldsymbol{x}}$ is the nominal state trajectory.

In the literature $\kappa(\cdot,\cdot)$ is known as the *ancillary* controller and is designed such that the state $\boldsymbol{x}$ remains in a robust control invariant tube around the nominal trajectory $\bar{\boldsymbol{x}}$.

Definition 1: First define $\tilde{\boldsymbol{x}} := \boldsymbol{x} - \bar{\boldsymbol{x}}$. The set $\Omega \subset \mathcal{X} \subset \mathbb{R}^n$ is a *robust control invariant (RCI) tube* if there exists an

ancillary controller $\kappa(\boldsymbol{x}, \bar{\boldsymbol{x}})$ such that if $\tilde{\boldsymbol{x}}(t_0) \in \Omega$ then for all realizations of the disturbance $\boldsymbol{d}$, $\tilde{\boldsymbol{x}} \in \Omega$ for all $t \geq t_0$.

Fundamentally, RCI tubes are a mathematical object that describes how "close" the system remains to the desired trajectory for all realizations of the disturbance. Assuming $\kappa(\cdot,\cdot)$, Ω, and $\mathcal{X}_{goal}$ are known, the following MPC optimization problem at time step k is solved:

P2: MPC Optimization Problem –

$$\begin{aligned} \underset{\bar{\boldsymbol{u}} \in \bar{\mathcal{U}}}{\text{minimize}} \quad & J_T(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}(\boldsymbol{x})) \\ \text{subject to} \quad & \dot{\bar{\boldsymbol{x}}} = f(\bar{\boldsymbol{x}}) + b(\bar{\boldsymbol{x}})\,\bar{\boldsymbol{u}} \\ & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}}, \quad \bar{\boldsymbol{u}} \in \bar{\mathcal{U}} \\ & \bar{\boldsymbol{x}} = \bar{\boldsymbol{x}}_0, \quad \bar{\boldsymbol{x}} \in \mathcal{X}_{goal}, \end{aligned}$$

where $\bar{\mathcal{X}} := \mathcal{X} \sim \Omega$ is the tightened state constraint and $\bar{\mathcal{U}} := \{\bar{\boldsymbol{u}} \in \mathcal{U} : \bar{\boldsymbol{u}} + \kappa(\boldsymbol{x}, \bar{\boldsymbol{x}}) \in \mathcal{U}\}$.

The objective of this paper is to show that CASC can be used as an ancillary controller $\kappa(\cdot,\cdot)$, with an associated RCI tube Ω, to enable robust, high-performance collision avoidance.

IV. Controller Design

A. Overview

This section presents the Composite Adaptive Sliding Controller (CASC) and provides analysis supporting its use as an ancillary controller in the tube MPC framework. The primary goal of synthesizing an ancillary controller is to minimize the associated RCI tube for bounded modeling error and disturbances. Sliding mode control (SMC) *completely cancels* any bounded modeling error or external disturbance - essentially reducing the RCI tube to zero. However, complete cancellation comes at the cost of high-frequency discontinuous control (aka. chattering) making it impractical for many real systems. The remainder of this section discusses how CASC achieves robustness through continuous control and how model uncertainty is reduced through high-rate parameter estimation. It is important to note that CASC relies heavily on past works. However, this work provides a new interpretation of past results in the context of robust path planning and online controller design using sensor information (Section V).

For brevity, the following analysis will only consider nonlinear second-order single input systems but the analysis can be easily extended to higher-order MIMO systems. Hence, (1) can be rewritten as

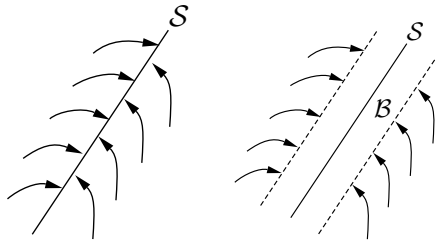
$$\ddot{x} = f(\boldsymbol{x}) + b(\boldsymbol{x})\,u + d \tag{2}$$

where $\boldsymbol{x} = [x\ \dot{x}]^T$ is the state vector, u is the control input, d is a bounded disturbance. The ancillary controller ensures the tracking error $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x_d}$, where $\boldsymbol{x_d} = [x_d\ \dot{x}_d]^T$, is bounded for all time given uncertainty in the dynamics (i.e. $f(\cdot)$, $b(\cdot)$) and the environment (i.e. d).

B. Sliding Mode Control

Let the manifold $\mathcal{S}$ be defined by the equation $s = 0$ where

$$s = \dot{\tilde{x}} + \lambda \tilde{x}, \tag{3}$$



(a) Sliding surface $\mathcal{S}$. (b) Boundary layer $\mathcal{B}$.

Fig. 2. Sliding manifolds. (a): Sliding surface for traditional sliding mode control. $\mathcal{S}$ is invariant to uncertainty and disturbances but requires discontinuous control. (b): A region $\mathcal{B}$ around $\mathcal{S}$ can be constructed such that $\mathcal{B}$ is invariant through continuous control.

and $\lambda > 0$. SMC aims to maintain $s = 0,\ \forall t > 0$ given bounded model uncertainty and external disturbances. This condition guarantees the tracking error converges to zero exponentially (from (3)).

It can be shown [16] that adding a term that is discontinuous across $\mathcal{S}$ ensures finite time convergence to $\mathcal{S}$ and that the state remains on $\mathcal{S}$ indefinitely (pictorially shown in Fig. 2a). The well known SMC law takes the form:

$$u = \bar{b}(\boldsymbol{x})^{-1}\left[\ddot{x}_d - \lambda\dot{\tilde{x}} - \bar{f}(\boldsymbol{x}) - K\,\mathrm{sgn}(s)\right], \tag{4}$$

where K is determined by the magnitude of the model uncertainty and disturbance.

The manifold $\mathcal{S}$, known as the sliding surface, is *invariant* to model uncertainty and disturbances by construction. Thus, the RCI tube associated with SMC decays exponentially to zero (through (3)) once the state reaches $\mathcal{S}$. This is an amazing property given that zero tracking error is guaranteed so long as K is selected appropriately. However, complete cancellation of model uncertainty and disturbances requires discontinuous control across $\mathcal{S}$ which, among other things, can excite unmodeled high-frequency dynamics and shorten the life span of the actuator thereby making SMC impractical for many systems.

C. Composite Adaptive Sliding Controller

One of the earliest techniques [17], [18] to eliminate discontinuous control was to introduce a thin *boundary layer* around the sliding surface $\mathcal{S}$. Let the boundary layer be defined as $\mathcal{B} := \{s : |s| \leq \Phi\}$ where Φ is the boundary layer thickness (shown in Fig. 2b). The control law that makes $\mathcal{B}$ attractive and invariant [16] is given by:

$$u = \bar{b}(\boldsymbol{x})^{-1}\left[\ddot{x}_d - \lambda\dot{\tilde{x}} - \bar{f}(\boldsymbol{x}) - K\,\mathrm{sat}(s/\Phi)\right], \tag{5}$$

where $\mathrm{sat}(\cdot)$ is the saturation function. Since $|s| \leq \Phi$ for all time, (3) implies $|\tilde{x}| \leq \Phi/\lambda$. Therefore, by definition, the boundary layer $\mathcal{B}$ is an RCI tube. Interestingly, the size of the RCI tube for boundary layer SMC can be interpreted as a design parameter since it is governed by Φ and λ – both of which are parameters of the controller. The size of the RCI tube can then be selected *a priori*. However, the tube size cannot be made arbitrarily small since the necessary bandwidth of the controller is inversely related to tube size.

It is instructive to analyze the dynamics of s inside the boundary layer. For simplicity assume $b(\boldsymbol{x}) = \bar{b}(\boldsymbol{x})$.

Substituting (5) into (3), yields

$$\dot{s} = -\frac{K}{\Phi} s + \left(\Delta f\left(\boldsymbol{x}\right) + d\right), \tag{6}$$

which can be interpreted as a low-pass filter where the input is the dynamics uncertainty $\Delta f\left(\cdot\right)$ and external disturbance d. The key term in (6) is the ratio K/Φ and how it dictates the response of s (and ultimately the control input) to uncertainty. In particular, (6) displays the tradeoff between robustness/RCI tube size and control activity: large K or small RCI tube increases the the magnitude of the high-frequency component of the s dynamics leading to a high-frequency control signal. This is undesirable for numerous reason (i.e. shortened actuator life) but is especially problematic for interconnected feedback systems where controller-controller interaction can cause instability [19].

The previous discussion highlighted the importance of keeping the ratio K/Φ small to reduce the influence of uncertainty. Ideally, K should only be as large as the disturbance magnitude D to keep $\mathcal{B}$ invariant. Thus, we propose to use online parameter estimation to reduce model uncertainty and minimize the ratio K/Φ. Before proceeding the following assumption about the structure of $f\left(\cdot\right)$ is made:

Assumption 4: Assume $f\left(\cdot\right)$ contains only parametric uncertainty and those parameters can be linearly parameterized. Specifically, $f\left(\cdot\right)$ can be written as

$$f\left(\boldsymbol{x}\right) = a_1 g\left(\dot{x}\right) + a_2 h\left(x\right), \tag{7}$$

where $g\left(\cdot\right)$ and $h\left(\cdot\right)$ are known functions, but a_1 and a_2 are unknown (possibly slowly time varying) coefficients. Further, $b\left(\cdot\right)$ is an unknown (possibly slowly time-varying) coefficients independent of the state but of known sign.

Estimating a_1, a_2, and b is done by finding an adaptation law for a given Lyapunov function such that the system is stable. The Lyapunov function and stability proof was first proposed in [20] and is presented here for completeness. First define the new composite variable s_Δ as,

$$s_\Delta = s - \Phi \text{sat}\left(s/\Phi\right), \tag{8}$$

which represents the distance from the boundary layer. We wish to derive an adaptation law that is convergent to the boundary layer. Consider the Lyapunov candidate function:

$$V = \frac{1}{2}s_\Delta^2 + \frac{1}{2}b\left(\gamma_1^{-1}\left(\hat{l}_1 - a_1/b\right)^2 + \gamma_2^{-1}\left(\hat{l}_2 - a_2/b\right)^2 + \gamma_b^{-1}\left(\hat{b}^{-1} - b^{-1}\right)^2\right) \tag{9}$$

where $\hat{l}_i$ is the estimate of a_i/b and γ_j is the adaptation rate for parameter j. One can show (see [20]) that selecting the following adaptation laws leads to asymptotic convergence:

$$\dot{\hat{l}}_1 = -\gamma_1 g\left(\dot{x}\right) s_\Delta \tag{10}$$

$$\dot{\hat{l}}_2 = -\gamma_2 h\left(x\right) s_\Delta \tag{11}$$

$$\dot{\hat{b}}^{-1} = -\gamma_b\left[-\ddot{x}_d + \lambda\dot{\tilde{x}} + K\text{sat}\left(s/\Phi\right)\right] s_\Delta. \tag{12}$$

As pointed out in [20], the above adaptation laws provide a natural dead zone for adaptation (i.e. when s is outside the boundary layer) which has been show to improve the performance of adaptive controllers [16].

It is important to acknowledge that using adaptation to reduce model uncertainty removes the invariance property of the boundary layer (since s can now leave $\mathcal{B}$). From a practicality standpoint, however, so long as s converges back to $\mathcal{B}$ sufficiently fast then the tracking error bound essentially remains unchanged. High-rate estimation without undesirable transients can be achieved through composite (or combined) adaptation, as first proposed in [2]. Composite adaptation fuses tracking error and model prediction error to update the unknown parameters. For systems of the form (7), the error in the actual and prediction acceleration is used for adaption. For brevity the predictor-error adaptation scheme is not derived here but a more detailed discussion can be found in [2], [16].

The CASC thus takes the following form:

$$\kappa_{\text{CASC}} = -\hat{l}_1 g\left(\dot{x}\right) - \hat{l}_2 h\left(x\right) + \hat{b}^{-1}\left[\ddot{x}_d - \lambda\dot{x} - \bar{f}\left(\boldsymbol{x}\right) - K\text{sat}\left(s/\Phi\right)\right], \tag{13}$$

with

$$\dot{\hat{l}}_1 = -\left(\gamma_1 s_\Delta + \gamma_{p_1} e_p\right) g\left(\dot{x}\right) \tag{14}$$

$$\dot{\hat{l}}_2 = -\left(\gamma_2 s_\Delta + \gamma_{p_2} e_p\right) h\left(x\right) \tag{15}$$

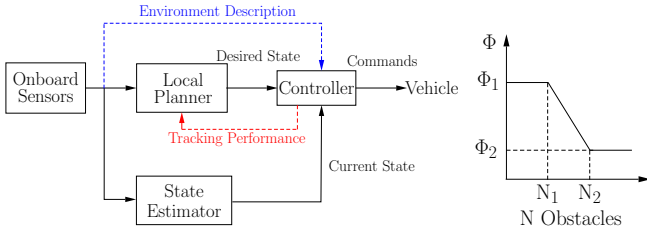
$$\dot{\hat{b}}^{-1} = -\gamma_b\left[-\ddot{x}_d + \lambda\dot{\tilde{x}} + K\text{sat}\left(s/\Phi\right)\right] s_\Delta - \gamma_{p_b} e_p \ddot{x}_p, \tag{16}$$

where $e_p = \ddot{x} - \ddot{x}_p$, $\ddot{x}_p = \hat{b}\kappa_{\text{CASC}}$, and γ_{p_i} is the predictor-error adaptation rate.

V. Robust Control in Path Planning

A general system architecture for autonomous navigation in unknown environments is shown in Fig. 3a. On-board sensors are used for global navigation, local collision avoidance, and localization. The local planner is responsible for generating a collision free trajectory that is tracked by a low-level controller. The distinguishing functionality of the local planner is to determine the likelihood of collision for a desired trajectory. Many local planners assume the low-level controller can track the desired trajectory with nearly zero error – making collisions imminent when model uncertainty or disturbances are present. Incorporating the theoretical guarantees of CASC into the local planner can be interpreted as adding a feedback loop between the planner and controller, the red dashed line in Fig. 3a. This feedback loop contains information about the controller's current and predicted performance that enables the planner to select a collision-free path with high confidence.

A byproduct of the CASC's design is the ability to dynamically change the boundary layer thickness based on the *current environment*. This can be accomplished by using perception data as feedforward into the controller (blue dashed line in Fig. 3a) and enables CASC to respond appropriately to the surrounding environment. For instance, the boundary layer thickness law in Fig. 3b, where N is the number of obstacles, ensures tight trajectory tracking in cluttered environments (e.g. a forest) and looser trajectory tracking in low clutter environments (e.g. an open field). This



(a) General system architecture. (b) Boundary layer thickness law.

Fig. 3. General architecture and boundary layer thickness law. (a): Robust local planning requires an additional feedback loop with the controller to make accurate predictions about the vehicle's response to model uncertainty and external disturbances. (b): The boundary layer thickness (and hence tracking precision) can be modified by the number of observed obstacles. Dynamically changing tracking precision permits more/less aggressive maneuvers in dense/open environments.

is an ideal property since precise tracking typically leads to aggressive maneuvers that require a lot of energy, which are problematic for energy-limited systems. Further, modifying the boundary layer thickness is much simpler than gain scheduling given only a single parameter needs to change. Generating a specialized response for different environments is crucial for the evolution of high-performance autonomous system – a capability naturally embedded in CASC.

VI. Quadrotor Model and Control

A. Model

Let the position of a quadrotor with mass m and inertia tensor $\boldsymbol{J}$ be described in an inertial frame I by vector $\boldsymbol{r}^I$. The vehicle's orientation with respect to the inertial frame and body angular rates are denoted as the quaternion $\boldsymbol{q_a}$ and vector $\boldsymbol{\omega}^B$.

The quaternion representation of the Newton-Euler equations of motion are

$$\ddot{\boldsymbol{r}}^I = \frac{1}{m}\mathcal{L}_{q_a}\boldsymbol{F}^B - \boldsymbol{c}\cdot|\dot{\boldsymbol{r}}|\dot{\boldsymbol{r}} - \boldsymbol{g} + \boldsymbol{d}, \tag{17}$$

$$\mathcal{L}_{q_a}\boldsymbol{F}^B = \boldsymbol{q_a}\otimes\begin{bmatrix}0\\ \boldsymbol{F}^B\end{bmatrix}\otimes\boldsymbol{q_a^*}, \tag{18}$$

$$\dot{\boldsymbol{\omega}}^B = \boldsymbol{J^{-1}}\left[\boldsymbol{M}^B - \boldsymbol{\omega}^B\times\boldsymbol{J}\boldsymbol{\omega}^B\right], \tag{19}$$

$$\dot{\boldsymbol{q}}_{\boldsymbol{a}} = \frac{1}{2}\boldsymbol{q_a}\otimes\begin{bmatrix}0\\ \boldsymbol{\omega}^B\end{bmatrix}, \tag{20}$$

where $\boldsymbol{c}$ is the body drag coefficient vector, $\boldsymbol{g} = [0\ 0\ g]^T$ is the gravity vector, $\boldsymbol{d}$ is an unknown but bounded disturbance vector, $\boldsymbol{F}^B = [0\ 0\ F_{total}]^T$ is the body-frame force vector, and $\boldsymbol{M}^B$ is the body-frame moment vector. Note (17)–(20) neglect higher order aerodynamic effects (i.e. blade flap).

Let F_i be the force produced by motor i. In practice, motor forces are generated by sending a throttle command $\delta_T \in [0\ 1]$ to an Electronic Speed Control (ESC) that governs the rotation rate of the propeller. Thus, there exists a mapping $b(\cdot)$ that can be approximately inverted to give the necessary throttle for a desired force F_d:

$$\delta_{T_{i,d}} = \hat{b}^{-1}\left(F_{i,d}\right), \tag{21}$$

where $b(\cdot)$ is identical to that presented in Section IV

Three sources of uncertainty exist in (17)-(21): the drag coefficient vector $\boldsymbol{c}$, inertia tensor $\boldsymbol{J}$, and thrust gain b^{-1}. The inertia tensor can be easily approximated if the mass of the individual components on the vehicle are known. Typically, extensive experimental testing is required to obtain the drag coefficient and thrust gain. However, (17) is of the same form as the dynamics presented in Section IV-C so the drag coefficient and thrust gain can be estimated online by CASC.

B. Control Architecture

The attitude dynamics in (19) typically evolve much faster in time than the position dynamics in (17) for quadrotors. This property permits successive loop closure [19] where the output of one controller serves as the reference for another. In this work, a desired trajectory is used as the reference for position and velocity control. Feedforward acceleration and jerk supplement position and velocity feedback to generate a desired attitude and angular rate. Feedback is performed on the desired attitude and angular rate generated by the outer-loop controller. CASC, used as the outer-loop controller, provides robust position and velocity control. A quaternion-based sliding controller is used for robust attitude control. See [21] for how position and velocity error is mapped to desired attitude and angular rates.

It is worth noting that successive loop closure requires sufficient frequency separation between the two loops. This places a constraint on how small the boundary layer thickness can be in CASC since, from (6), the smaller the boundary layer thickness the higher-frequency content contained in the control signal. Thus, the boundary layer thickness, and hence the size of the RCI tube, must be carefully selected to prevent coupling of the two loops.

VII. Test Environment

A. Simulation

CASC was tested in a simulation environment capable of providing real-time perception data and dynamics modeling for a quadrotor. The simulator's dynamics engine is a custom C++ implementation that has been experimentally verified. Gazebo [22] and the hector_quadrotor package [23] were used to simulate perception data (in the form of a point cloud) as the quadrotor navigated custom worlds. 3-D depth perception was simulated using the Asus model with a 60deg FOV, 6m maximum range, and 60Hz frame rate. The Relaxed-constraint Triple Integrator Planner (R-TIP) [3] was used for low-latency collision avoidance.

World 1, shown in Fig. 4, consisted of 150 obstacles randomly placed in a 60x20m grid. The vehicle starts at the origin (●) and attempts to navigate to the goal (●) at a speed of $5^{\mathrm{m}}/_{\mathrm{s}}$. Two crosswind d_y profiles, a sinusoid (Fig. 6a) and a series of steps (Fig. 6b), were applied to test the controller's ability to reject disturbances and the planner's ability to utilize the predicted tracking performance. In addition, the thrust gain b^{-1} and drag coefficient vector $\boldsymbol{c}$ were assumed to be unknown. A PID controller was also tested to provide a performance baseline.

World 2, shown in Fig. 5, was designed to showcase CASC's ability to adjust to different environments online while maintaining robustness to external disturbances. The first portion of the world had 150 obstacles placed in a

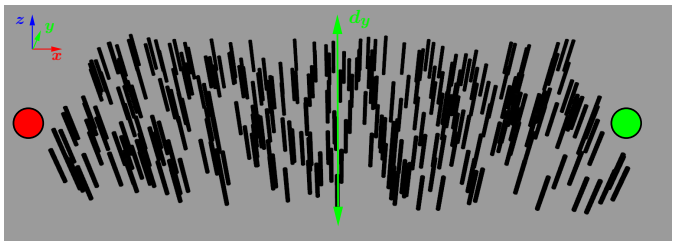


Fig. 4. Top-down view of the world 1. The vehicle starts at the origin (●) and navigates to the goal location 80m away (●) while avoiding obstacles. A sinusoid and step crosswind was applied to test CASC's robustness to disturbances.

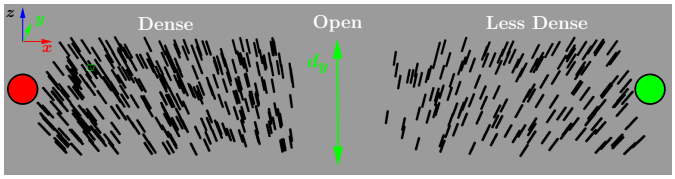
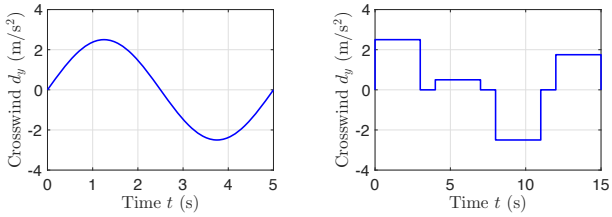


Fig. 5. Top-down view of world 2. The vehicle starts at the origin (●) and navigates to the goal location 120m away (●) while avoiding obstacles. The environment was designed to test CASC's ability to dynamically changed its parameters depending on the level of clutter in environment. A crosswind was still applied to demonstrate CASC's robustness to disturbances.



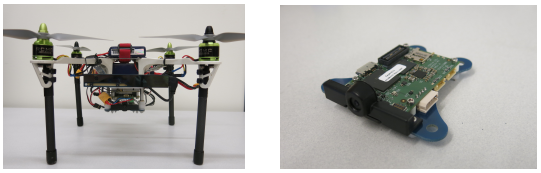
(a) Sinusoid crosswind. (b) Step crosswind.

Fig. 6. Crosswind profiles used as external disturbances. (a): Sinusoid crosswind with an amplitude of 2.5m/s^2 and period 5s. (b): Step inputs crosswind with a maximum magnitude of 2.5m/s^2.

40X20m grid, the second portion had no pole in a 40X20m grid, and the third portion had 75 obstacles in a 40x20m grid. The ideal response of the controller would be to have tight tracking in the first phase, loose tracking in the second, and moderate tracking in the third. The key to dynamically changing the controller's parameters is to have perception data fedforward to the controller and a feedback loop between the local planner and controller. This allowed the boundary layer thickness to expand when no obstacles were in the sensor's FOV and contract otherwise (similar to Fig. 3b). In addition, the planner adjusts the collision buffer around the vehicle to accommodate for large/small tracking error.

B. Hardware

The quadrotor used in this work (Fig. 7a) is equipped with a Jetson TX1 for onboard perception and planning and a Qualcomm Snapdragon Flight (Fig. 7b) for estimation and control. Fig. 8 shows the system diagram and the rates of each subsystem. Qualcomm's visual inertial SLAM (VISLAM) algorithm outputs a position and orientation at 30Hz [24]. The pose output from VISLAM updates a nonlinear observer that propagates high-rate IMU measurements. The



(a) Quadrotor platfrom. (b) Snapdragon Flight board.

Fig. 7. Experimental test platform. (a): Quadrotor with Jetson TX1 (perception and planning) and Snapdragon Flight (estimation and control). (b): Snapdragon Flight board with two onboard cameras and IMU.

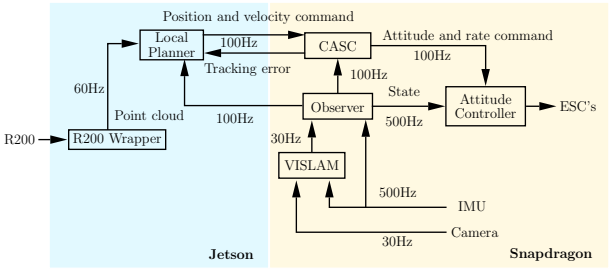
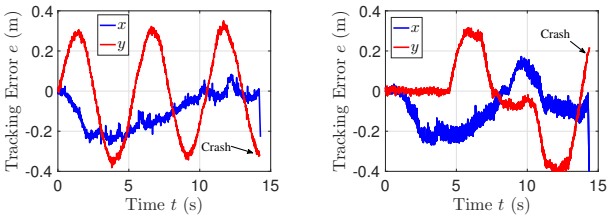


Fig. 8. System architecture with associated rates for each subsystem.



(a) Tracking error with sinusoid disturbance. (b) Tracking error with step disturbance.

Fig. 9. PID x-y tracking error. (a): Poor tracking performance, with a peak error of 0.3m, for the sinusoid disturbance caused the vehicle to crash into an obstacle. (b): Poor tracking performance with a peak error of 0.3m was also observed for the step disturbance again causing a crash.

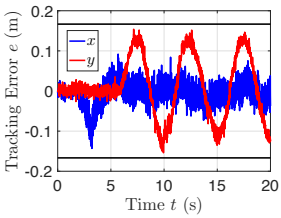
state vector includes the vehicle's position, velocity, attitude, angular rate, and IMU biases. A quaternion-based sliding controller is used for high-precision robust attitude control. Robust attitude control is crucial to prevent interactions between the inner-loop and outer-loop controllers. The flight stack is open source and can be found online.

On the Jetson TX1, an Intel Realsense R200 stereo camera generates a point cloud that is used by the local planner for collision avoidance. Additional algorithmic components (i.e. global planner, symbolic perception) will be added to the Jetson TX1 in the future. The vehicle weighs 0.9kg and hovers at 45% throttle.

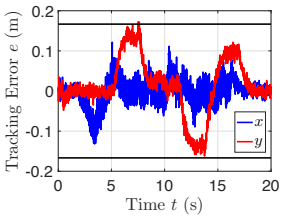
VIII. Results

A. Simulation

A PID controller was used as the performance baseline for world 1. Fig. 9a and Fig. 9b show the tracking error for the sinusoid and step crosswind disturbance, respectively. The large tracking error of nearly 0.4m, and the inability to predict it, caused the vehicle to crash in both test cases. The inability to robustly track a desired trajectory is further motivation for employing robust controllers (such as CASC) into path planning.

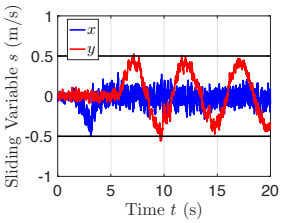


(a) Tracking error with sinusoid disturbance.

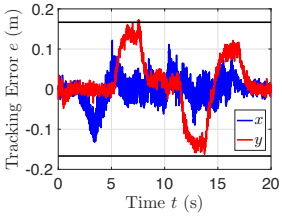


(b) Tracking error with step disturbance.

Fig. 10. CASC x-y tracking error. (a): Tracking error for sinusoid disturbance is 2.5x smaller than that of PID and is strictly less than the theoretical bound (black) as desired. (b): Tracking error for step disturbance is 2.5x smaller than that of PID and contained within the theoretical bounds (black) except at one time step where the measurement noise pushed it outside the boundary.



(a) Sliding variable with sinusoid disturbance.

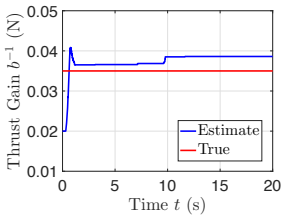


(b) Sliding variable with step disturbance.

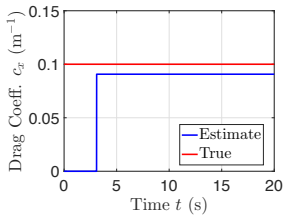
Fig. 11. CASC x-y sliding variable. (a): Sliding variable for sinusoid disturbance breifly exits boundary layer (black) but is contained for almost the entire flight. (b): Sliding variable for step disturbance briefly leaves the boundary layer (black) but converges quickly back to the interior and remains there for the rest of the flight.

The tracking error for CASC in world 1 with a sinusoid and step crosswind disturbance is shown in Fig. 10a and Fig. 10b, respectively. The tracking error for both cases was reduced by a factor of 2.5 when compared to that of PID and is less than the predicted tracking error bound (shown as the black line). In both cases the vehicle was able to safely reach the goal – a direct result of the robustness of CASC. The sliding variable s is also shown in Fig. 11a for the sinusoid and in Fig. 11b for the step disturbance where the boundary layer (black line) is added for reference. As stated in Section IV-C, online estimation reduces model uncertainty at the expense of maintaining the the invariance property of the boundary layer. However, s rapidly converges back into the boundary layer because of the high-rate estimation provided by combined adaptation. Therefore, while the theoretical guarantees are technically lost through adaptation, Fig. 11 clearly shows in practice that s will remain in or extremely close to the boundary layer.

The thrust gain and drag coefficient vector were unknown so they were estimated online. Fig. 12a and Fig. 12b show the estimate of the thrust gain and x drag coefficient, respectively, for the sinusoid disturbance case. The true values are shown for reference. The estimated thrust gain varied slightly throughout the flight but remains within 10% of the true value. Additionally, the drag coefficient converged quickly to with 10% of the true value and remained steady throughout the flight. Similar results were obtained for the step disturbance case. Fig. 12 demonstrates CASC's ability to accurately estimate parameters online.

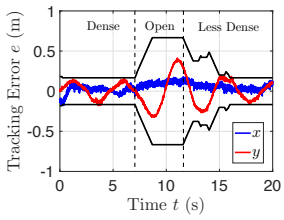


(a) Thrust gain.

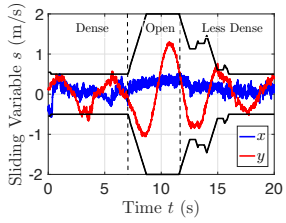


(b) x drag coeff.

Fig. 12. Estimate and true value of model parameters. (a): The thrust gain converges quickly during takeoff with little overshoot. The estimate is within 10% of the true value for the entirety of the flight. (b): The x drag coefficient quickly converges to within 10% of the true value just as the vehicle reaches top speed.



(a) Tracking error.



(b) Sliding variable.

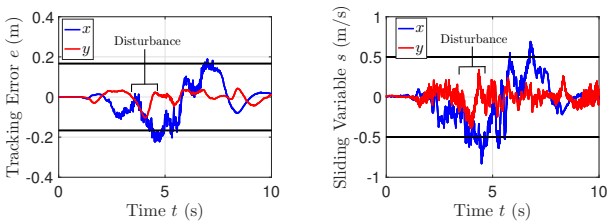
Fig. 13. CASC x-y tracking error and sliding variable for dynamic boundary layer thickness with sinusoid disturbance. (a): The tracking error bound (black) is small when the vehicle is in the dense and less dense environment but increases in open space. (b): The boundary layer thickness (black) exhibits identical behavior. The sliding variable converges quickly back to the boundary because of high-rate parameter estimation.

The proposed boundary layer thickness modification law (based on obstacle density) was tested in world 2 where the quadrotor had to navigate in an environment with varying obstacle densities. Fig. 13a and Fig. 13b show the tracking error and sliding variable, respectively, for the dense, open, and less dense portions of the environment. The tracking error bound in Fig. 13a and boundary layer thickness in Fig. 13b (both shown in black) are small for the dense obstacle phase, expand in the open space phase, and contract in the less dense obstacle phase. Expansion in open space minimizes expended energy by preventing unnecessary large control action to reject disturbances. Fig. 13 displays CASC's ability to respond to changes in the environment.

B. Hardware

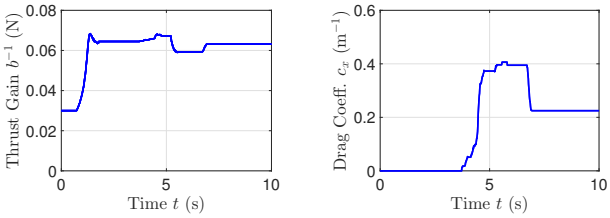
CASC was experimentally tested on the quadrotor platform (Section VII-B) using only onboard sensing to navigate through the unknown environment shown in Fig. 1. The 15m traversal required the vehicle to avoid obstacles while simultaneously rejecting an external disturbance generated by two leaf blowers.

The tracking error and sliding variable for the experiment is shown in Fig. 14a and Fig. 14b, respectively. Note that the leaf blowers are pointed along the $+y$ axis. The tracking error for x and y increases when the disturbance is applied as expected. The y tracking error remains within the predicted bounds while the x tracking error briefly exceed the predicted amount by only 4cm. This was attributed to a combination of measurement noise and the need for a more sophisticate aerodynamic model for the complicated flow-field generated by the blowers. The y sliding variable is completely contained



(a) Tracking error. (b) Sliding variable.

Fig. 14. CASC x-y tracking error and sliding variable for experiment with step disturbance. (a): Tracking error remains within or near bounds (black) predicted by CASMC. (b): Modeling errors cause the sliding variable to briefly leave the boundary layer (black) but it quickly converges back through high-rate estimation.



(a) Thrust gain. (b) x drag coeff.

Fig. 15. Parameter estimates for hardware experiments. (a): Thrust gain converges quickly during takeoff and only varies by 8.6% throughout the flight. (b): The x drag coefficient varies during the disturbance but quickly converges to a steady state value after.

in the boundary layer. The x component briefly exits because of modeling error but reenters quickly. Fig. 14 demonstrates CASC's ability to reject disturbances on a real system.

The thrust gain and x drag coefficient estimates are shown in Fig. 15a and Fig. 15b, respectively. The thrust gain converges quickly during takeoff and only varies by 8.6% during the flight - indicating the estimate converged to approximately the true value. The drag coefficient estimate exhibits more variation throughout the flight but converges just after the vehicle passes the blowers. The variation was attributed to the simplicity of aerodynamic model compared to the complicated flow-field generated by the blowers.

IX. Conclusion

This work presented an analysis and experimental verification of a nonlinear controller that is robust to disturbances and reduces model uncertainty through high-rate parameter estimation. The key properties of the Composite Adaptive Sliding Controller (CASC) are the guaranteed tracking performance when disturbances are present (which can be used to establish provably safe path planning) and the ability to change its tracking precision based on the current operating environment. CASC was shown to reject external disturbances and estimate model parameters on a quadrotor navigating in unknown environment using only onboard sensors.

X. Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1122374 and by the DARPA Fast Lightweight Autonomy (FLA) program. The author would also like to acknowledge Boeing Research & Technology for support of the indoor flight facility and hardware.

References

[1] Dimitri P Bertsekas and Ian B Rhodes. On the minimax reachability of target sets and target tubes. *Automatica*, 7(2):233–247, 1971.

[2] Jean-Jacques E Slotine and Weiping Li. Composite adaptive control of robot manipulators. *Automatica*, 25(4):509–519, 1989.

[3] Brett T. Lopez and Jonathan P. How. Aggressive 3-d collision avoidance for high-speed navigation. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017.

[4] David Q Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[5] Wilbur Langson, Ioannis Chryssochoos, SV Raković, and David Q Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, 2004.

[6] David Q Mayne, SV Raković, Rolf Findeisen, and Frank Allgöwer. Robust output feedback model predictive control of constrained linear systems. *Automatica*, 42(7):1217–1222, 2006.

[7] Matthias Althoff, Olaf Stursberg, and Martin Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 4042–4048. IEEE, 2008.

[8] Matthias Althoff and John M Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, 2014.

[9] Daniel Althoff, Matthias Althoff, and Sebastian Scherer. Online safety verification of trajectories for unmanned flight with offline computed robust invariant sets. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3470–3477. IEEE, 2015.

[10] Anirudha Majumdar and Russ Tedrake. Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research*, 36(8):947–982, 2017.

[11] Sumeet Singh, Anirudha Majumdar, Jean-Jacques Slotine, and Marco Pavone. Robust online motion planning via contraction theory and convex optimization. *ICRA submission*, 2017.

[12] Ian R Manchester and Jean-Jacques E Slotine. Control contraction metrics and universal stabilizability. *IFAC Proceedings Volumes*, 47(3):8223–8228, 2014.

[13] Winfried Lohmiller and Jean-Jacques E Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.

[14] Matteo Rubagotti, Davide Martino Raimondo, Antonella Ferrara, and Lalo Magni. Robust model predictive control with integral sliding mode in continuous-time sampled-data nonlinear systems. *IEEE Transactions on Automatic Control*, 56(3):556–570, 2011.

[15] Reza Olfati-Saber. *Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles*. PhD thesis, Massachusetts Institute of Technology, 2001.

[16] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*, volume 199. prentice-Hall Englewood Cliffs, NJ, 1991.

[17] Jean-Jacques Slotine and Shankar S Sastry. Tracking control of non-linear systems using sliding surfaces, with application to robot manipulators. *International journal of control*, 38(2):465–492, 1983.

[18] Jean-Jacques E Slotine. Sliding controller design for non-linear systems. *International Journal of control*, 40(2):421–434, 1984.

[19] Jonathan P How, Emilio Frazzoli, and Girish Vinayak Chowdhary. Linear flight control techniques for unmanned aerial vehicles. In *Handbook of Unmanned Aerial Vehicles*, pages 529–576. Springer, 2015.

[20] J-JE Slotine and JA Coetsee. Adaptive sliding controller synthesis for non-linear systems. *International Journal of Control*, 43(6):1631–1651, 1986.

[21] Brett Thomas Lopez. Low-latency trajectory planning for high-speed navigation in unknown environments. Master's thesis, Massachusetts Institute of Technology, 2016.

[22] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2149–2154. IEEE, 2004.

[23] Johannes Meyer, Alexander Sendobry, Stefan Kohlbrecher, Uwe Klingauf, and Oskar von Stryk. Comprehensive simulation of quadrotor uavs using ros and gazebo. In *3rd Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*, page to appear, 2012.

[24] Giuseppe Loianno, Chris Brunner, Gary McGrath, and Vijay Kumar. Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and imu. *IEEE Robotics and Automation Letters*, 2(2):404–411, 2017.