Safety from Fast, In-the-Loop Reachability with Application to UAVs

Christian Llanes
The Georgia Institute of Technology
Atlanta, Georgia, USA
Christian.Llanes@GaTech.edu

Matthew Abate
The Georgia Institute of Technology
Atlanta, Georgia, USA
Matt.Abate@GaTech.edu

Samuel Coogan
The Georgia Institute of Technology
Atlanta, Georgia, USA
Sam.Coogan@GaTech.edu

ABSTRACT

We present a runtime assurance (RTA) mechanism for ensuring safety of a controlled dynamical system and an application to collision avoidance of two unmanned aerial vehicles (UAVs). We consider a dynamical system controlled by an unverified and potentially unsafe primary controller that might, *e.g.*, lead to collision. The proposed RTA mechanism computes at each time the reachable set of the system under a backup control law. We then develop a novel optimization problem based on control barrier functions that filters the primary controller when necessary in order to keep the system's reachable set within reach of a known, but conservative, safe region. The theory of mixed monotone systems is leveraged for efficient reachable set computation and to achieve a tractable optimization formulation. We demonstrate the proposed RTA mechanism on a dual multirotor UAV case study which requires a fast controller update rate as a result of the small time-scale rotational dynamics. In implementation, the algorithm computes the reachable set of an eight dimensional dynamical system in less than five milliseconds and solves the optimization problem in under one millisecond, yielding a controller update rate of 100Hz.

KEYWORDS

Controller Verification, Runtime Assurance, Monotone Systems

1 INTRODUCTION

Unmanned aerial vehicles (UAVs) have become an increasingly popular tool for agriculture, photography, search and rescue, and mapping, and are also becoming more accessible to general consumers. This increased accessibility has led to several notable accidents involving UAVs and civilians [17], emphasizing the need for verifiable safety guarantees in highly dynamic systems such as UAVs.

One attractive means of ensuring safety for controlled dynamical systems involves filtering a potentially unsafe control input at runtime; an approach referred to in literature as runtime assurance (RTA) [14]. Canonical examples of RTA mechanisms include the Simplex architecture [8, 20], which uses a decision module to switch to a backup control scheme when necessary, and control barrier functions (CBFs) [6, 7], which adjust desired control actions in a minimally invasive way to ensure forward invariance of a predetermined safe subset of the state space. A challenge with simplex architecture design is constructing a decision module that does not intervene too late [8]. With CBFs the decision for when to alter the desired control input is done smoothly and can be tuned by modifying a class-$\mathcal{K}$ function.

Of particular relevance, the works [11–13] present a CBF-type RTA mechanism where a simulation of the system dynamics is conducted prior to each evaluation of the barrier constraint. This approach avoids an explicit description of the maximal controlled invariant set of the dynamics, but requires precises knowledge of the system dynamics for real world implementation and does not allow for, *e.g.*, system models containing disturbance inputs. The primary difficulty in extending this approach to systems with disturbances is that computing reachable sets is generally computationally prohibitive for real-time implementations. A solution is introduced in [2], which considers systems with disturbances, where hyperrectangular over-approximations of reachable sets are efficiently computed in-the-loop by applying mixed monotone systems theory [10]; the resulting RTA mechanism is then formulated as a robust optimization problem, evaluated over the system's reachable set.

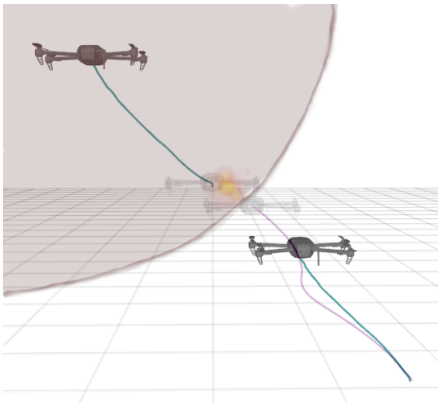


Figure 1: Simulation of two quadrotors avoiding a collision trajectory from an unsafe controller. The collision is avoided via supervision of our proposed algorithm.

In the present paper we use the framework proposed in [2] with novel computation saving techniques in order to accommodate highly dynamic systems that require fast controller update rates. We are particularly motivated by applications to multirotor UAVs, and a main contribution of this paper is the application of the proposed RTA mechanism to an eight-dimensional system modeling two multirotors in flight.

Our proposed algorithm is as follows. We first suppose existence of a backup control policy that is verified *a priori* to render a given subset of the state space robustly forward invariant. Motivated by applications where such subsets are generally conservative, our objective is to allow the system to safely evolve beyond this initial verified subset. To do so, we propose computing an over-approximation of the system's reachable set under the backup control law. By ensuring that the reachable set becomes fully contained within the verified safe subset at some point along the prediction horizon, safety is ensured and the system is allowed to evolve beyond the

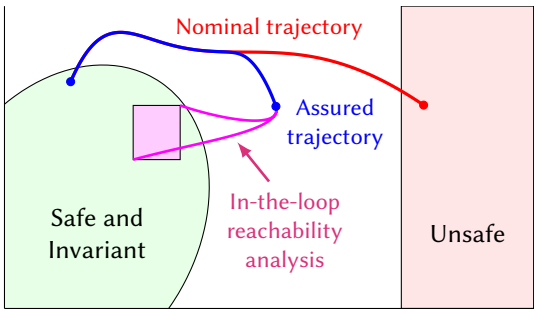


Figure 2: Depiction of run time assurance approach. At each time, the nominal control input is modified in order to ensure that the system's reachable set is contained within a known controlled forward invariant set.

verified subset. Moreover, the backup controller need not ever be actually applied to the system; rather, it acts as a certificate that continued safety is always within reach. This approach is depicted graphically in Figure 2.

We demonstrate our methodology for collision avoidance with an experimental demonstration of two quadrotor UAVs. [1] We restrict motion of each UAV to a vertical plane, and the result is an eight-dimensional system where disturbances account for unmodeled dynamics. The proposed RTA mechanism is implemented in ROS2 and runs on a Jetson TX2 single-board computer onboard each vehicle. A flight management unit (FMU) runninng PX4 autopilot is used for state estimation and control of the individual motors. The FMU also runs a low-level body-rate controller tracking desired body angular rates sent from the Jetson TX2. As demonstrated, and unsafe position controller commands the vehicles to a singular local position that leads to a collision. The proposed RTA mechanism detects the unsafe trajectory and commands each vehicle to remain a safe displacement distance.

This paper is organized as follows. We introduce preliminaries on run time assurance in Section 3.1 and mixed monotone systems theory in Section 3.2. In Section 4, we present our proposed RTA algorithm and provide implementation pseudocode. In Section 5 we present a collision avoidance case study for multirotor UAVs and in Section 6 we demonstrate our results via a flight experiment of two quadrotors supervised by our proposed algorithm and an unsafe controller that nominally would lead to a collision.

2 NOTATION

We denote vector entries via subscript, *i.e.*, x_i for $i \in \{1, \cdots, n\}$ denotes the i^{th} entry of $x \in \mathbb{R}^n$. Given $x, y \in \mathbb{R}^n$ with $x_i \leq y_i$ for all i,

$$[x,\, y] := \{z \in \mathbb{R}^n \mid x_i \leq z_i \leq y_i \text{ for all } i\}$$

denotes the hyperrectangle with endpoints x and y, and

$$\langle\!\langle x,\, y \rangle\!\rangle := \{z \in \mathbb{R}^n \mid z_i \in \{x_i,\, y_i\} \text{ for all } i\}$$

denotes the finite set of 2^n vertices of $[x, y]$. We also allow $x_i \in \mathbb{R} \cup \{-\infty\}$ and $y_i \in \mathbb{R} \cup \{\infty\}$ so that $[x, y]$ defines an *extended hyperrectangle*, that is, a hyperrectangle with possibly infinite extent in some coordinates.

Let (x, y) denote the vector concatenation of $x, y \in \mathbb{R}^n$, *i.e.*, $(x, y) := [x^T\, y^T]^T \in \mathbb{R}^{2n}$. Given $a = (x, y) \in \mathbb{R}^{2n}$ with $x_i \leq y_i$ for all i, we denote by $[\![a]\!]$ the hyperrectangle formed by the first and last n components of x, *i.e.*, $[\![a]\!] := [x, y]$, and similarly $\langle\!\langle a \rangle\!\rangle := \langle\!\langle x, y \rangle\!\rangle$.

3 PRELIMINARIES

3.1 Preliminaries on Run Time Assurance

We begin with a short review of run time assurance and the online enforcement of safety constraints for controlled dynamical systems. We are interested in systems that are affine-in-control

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ is the system state and $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input. Throughout this section we denote by $\Phi(t; x, \mathbf{u})$ the state of (1) reached at time t when beginning at state $x \in \mathcal{X}$ at time 0 and evolving subject to the closed-loop feedback control policy $\mathbf{u} : \mathcal{X} \to \mathcal{U}$, and we assume that the relevant functions f and g in (1) are continuously differentiable in their inputs so that in particular $\Phi(t; x, \mathbf{u})$ is unique when it exists.

Safety of the system (1) is formalised via a set invariance constraint. A set $S \subset \mathcal{X}$ is said to be *controlled forward invariant* for (1) when there exists a control policy $\mathbf{u}$ so that $\Phi(t; x, \mathbf{u}) \in S$ for all $x \in S$ and all $t \geq 0$. Throughout this section, we assume always that subsets as in $S \subset \mathcal{X}$ are defined to be the super-zero level-set of some continuously differentiable function $h(x)$, *i.e.*,

$$S := \{x \in \mathcal{X} \mid h(x) \geq 0\}, \tag{2}$$

and the remainder of this section deals with constructing such sets.

Control barrier functions provide a means of designing controllers to render S forward invariant. Define by $L_f h(x)$ and $L_g h(x)$ the Lie derivatives of $h(x)$ along f and g, respectively, $L_f h(x) = \frac{\partial h}{\partial x}(x) f(x)$, $L_g h(x) = \frac{\partial h}{\partial x}(x) g(x)$. In the special instance where $h(x)$ has relative degree 1 on S, *i.e.*, $L_g h(x) \neq 0$ for all $x \in S$, and

$$\left(\sup_{u \in \mathcal{U}} L_f h(x) + L_g h(x) u \right) \geq -\alpha(h(x)) \tag{3}$$

holds for all $x \in S$ for some class-$\mathcal{K}$ function α, the function $h(x)$ is said to be a *control barrier function for* (1). In this instance the set S is controlled forward invariant for (1) and the argument solution to the left side of (3) provides a controller which renders S forward invariant. In particular, for all control policies $\mathbf{u}^{\text{d}} : \mathcal{X} \to \mathcal{U}$, the *assured* controller given by

$$\begin{aligned} \mathbf{u}^{\text{RTA}}(x) = \underset{u \in \mathcal{U}}{\arg\min} & |\mathbf{u}^{\text{d}}(x) - u| \\ \text{s.t. } & L_f h(x) + L_g h(x) u \geq -\alpha(h(x)) \end{aligned} \tag{4}$$

renders S forward invariant for the closed-loop dynamics

$$\dot{x} = f(x) + g(x)\mathbf{u}^{\text{RTA}}(x) \tag{5}$$

and $\mathbf{u}^{\text{RTA}}(x)$ evaluates to $\mathbf{u}^{\text{d}}(x)$ when x is far from the boundary of S.

Despite the novelty that control barrier functions provide, it may be the case that $L_g h(x) = 0$ for some $x \in S$ and in this instance the controller (4) can generally not be employed to assure the forward invariance of S. To address this issue, techniques have

[1] The code and video that accompanies this case study is publicly available through the GaTech FACTS Lab Github: github.com/gtfactslab/Llanes_ICCPS2022.

been devised for constructing barrier functions using higher-order time-derivatives of h [19, 24]. For instance, when $L_g h(x) = 0$ for all x it is sometimes possible to employ a barrier function

$$h'(x) = h(x) + cL_f h(x) \tag{6}$$

for $c \geq 0$, *i.e.*, it may be the case that $L_g h'(x) \neq 0$ for all x and in this instance $h'(x)$ can be used with (3) to provide a controller rendering S forward invariant. This approach also generalizes to higher-order time derivatives of the boundary function h.

A problem arises when $h(x)$ does not have a well defined relative degree on S, as discussed for the case of multirotor dynamics in the following example.

Example 1. Consider a multirotor UAV fixed to the Y–Z plane, *i.e.*, a planar multirotor, with dynamics

$$\begin{aligned} \ddot{y} &= \frac{\tau}{m}\sin(\phi) \\ \ddot{z} &= g - \frac{\tau}{m}\cos(\phi) \\ \dot{\phi} &= \omega_x \end{aligned} \tag{7}$$

where $y, z \in \mathbb{R}$ defines the UAV horizontal and vertical position, $\phi \in \mathbb{R}$ is the roll angle, $\tau, \omega_x \in \mathbb{R}$ are the applied thrust and angular velocity, respectively, and g is the gravitational constant. Define the system state by $x = (y, z, \dot{y}, \dot{z}, \phi) \in \mathcal{X} = \mathbb{R}^5$ and consider any set $S = \{x \in \mathbb{R}^5 \mid h(y, z) \geq 0\}$, *i.e.*, the set boundary does not depend on the UAV roll angle. Choose also any point $x \in \mathcal{X}$ where $\frac{\partial h}{\partial y}(x) \neq 0$ and $\frac{\partial h}{\partial z}(x) = 0$; for example $h^1(y) = -y$, *i.e.*, the boundary of the safe set is the $y = 0$ vertical plane and a state x is safe when $y \leq 0$. Then $\dot{h}(y, z)$ is not dependant on (τ, ω_x) and

$$\ddot{h}(x) = \frac{\partial^2 h(x)}{\partial y^2}\dot{y} - \frac{\partial h(x)}{\partial y}\tau\sin(\phi) + \frac{\partial^2 h(x)}{\partial z^2}\dot{z} + \frac{\partial h(x)}{\partial z}(\tau\cos(\phi) - g). \tag{8}$$

Now, consider any state $x = (y, z, \dot{y}, \dot{z}, \phi)$ where $\dot{y} = \dot{z} = 0$ and $\phi = 0$, *i.e.*, the UAV is hovering. Then, $\ddot{h}(x) = 0$ regardless of the input, however note that $\dddot{h}(x) \neq 0$ when instead $\phi \neq 0$. In this way $h(x)$ does not have a well defined relative degree at x. For this reason, existing methods for forming assured controllers, which deal with systems with an undefined relative degree at isolated points far from the boundary of S [23] do not apply. ■

A solution is provided in [11, 12] which explores controlled forward invariance with respect to an *implicitly* defined safe set. Given $S = \{x \in \mathcal{X} \mid h(x) \geq 0\}$, assume knowledge of a *backup control policy* $\mathbf{u}^{\mathrm{b}} : \mathcal{X} \to \mathcal{U}$ that is known to render S forward invariant. Next, construct the controller

$$\begin{aligned} \mathbf{u}^{\mathrm{RTA}}(x) = \underset{u \in \mathcal{U}}{\arg\min} & |\mathbf{u}^{\mathrm{d}} - u| \\ \text{s.t. } & \nabla h(\Phi(T; x, \mathbf{u}^{\mathrm{b}}))\frac{\partial \Phi(T; x, \mathbf{u}^{\mathrm{b}})}{\partial x}(f(x) + g(x)u) \geq \alpha(h(x)) \end{aligned} \tag{9}$$

for fixed $T \geq 0$ where $\nabla h(x)$ denotes the gradient of h evaluated at x. Now (9) renders the set

$$S' = \{x \in \mathcal{X} \mid \Phi(T; x, \mathbf{u}^{\mathrm{b}}) \in S\} \tag{10}$$

forward invariant for (1). The main idea of (9) is to consider, at all times $t \geq 0$, a T-time-unit simulation of (1) under $\mathbf{u}^{\mathrm{b}}$ and ensure that the system state at time T is contained within S. This approach avoids the relative degree issue detailed above since safety guarantees on (1) are determined via a system simulation; under mild technical conditions, it is guaranteed that the implicit barrier construction has relative degree 1 [9, Theorem 3]. Moreover, the optimization problem in (9) is always feasible, provided $x(0) \in S$, since $\mathbf{u}^{\mathrm{b}}(x) \in \mathcal{U}$ for all $x \in S'$, *i.e.*, a feasible solution to (9) always exists.

The main result of this paper is to generalise the implicit control barrier formation (9) to systems with disturbances, and we are particularly motivated by applications in UAVs for which (i) real world system behavior may deviate from the dynamic model, (ii) the system's relative degree may not exist at certain points, as demonstrated in Example 1, and (iii) actuator constraints exist. As discussed in later sections, we accommodate disturbances in the implicit control barrier function formulation via the explicit computation of reachable sets in the control loop.

3.2 Preliminaries on Mixed Monotone Systems

To accommodate uncertainty in the dynamics, and facilitate the online computation of reachable sets, we propose using mixed monotone systems theory [10]. A system is mixed monotone when there exists a related decomposition function that separates the system dynamics into increasing and decreasing components, and this decomposition function is useful for, *e.g.*, efficiently computing reachable sets for the initial mixed monotone system. In this section, we provide a brief overview of mixed monotone systems theory.

We consider a system with disturbances given by

$$\dot{x} = F(x, w) \tag{11}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state and $w \in \mathcal{W} \subset \mathbb{R}^p$ is the disturbance input, and we assume throughout this section that the state space $\mathcal{X}$ is an extended hyperrectangle and the disturbance space $\mathcal{W} := [\underline{w}, \overline{w}]$ is a hyperrectangle.

Definition 1. [10] Given a locally Lipschitz continuous function $d : \mathcal{X} \times \mathcal{W} \times \mathcal{X} \times \mathcal{W} \to \mathbb{R}^n$, the system (11) is *mixed monotone with respect to d* if all of the following hold:

- For all $x \in \mathcal{X}$ and all $w \in \mathcal{W}$, $d(x, w, x, w) = F(x, w)$.
- For all $i, j \in \{1, \cdots, n\}$ with $i \neq j$, $\frac{\partial d_i}{\partial x_j}(x, w, \widehat{x}, \widehat{w}) \geq 0$ for all $x, \widehat{x} \in \mathcal{X}$ and all $w, \widehat{w} \in \mathcal{W}$ whenever the derivative exists.
- For all $i, j \in \{1, \cdots, n\}$, $\frac{\partial d_i}{\partial \widehat{x}_j}(x, w, \widehat{x}, \widehat{w}) \leq 0$ for all $x, \widehat{x} \in \mathcal{X}$ and all $w, \widehat{w} \in \mathcal{W}$ whenever the derivative exists.
- For all $i \in \{1, \cdots, n\}$ and all $k \in \{1, \cdots, m\}$, $\frac{\partial d_i}{\partial w_k}(x, w, \widehat{x}, \widehat{w}) \geq 0$ and $\frac{\partial d_i}{\partial \widehat{w}_k}(x, w, \widehat{x}, \widehat{w}) \leq 0$ for all $x, \widehat{x} \in \mathcal{X}$ and all $w, \widehat{w} \in \mathcal{W}$ whenever the derivative exists. ■

When (11) is mixed monotone with respect to d, then d is a *decomposition function* for (11), and when d is clear from context we simply say that (11) is mixed monotone. Given d,

$$\begin{bmatrix} \dot{x} \\ \dot{\widehat{x}} \end{bmatrix} = e(x, \widehat{x}) := \begin{bmatrix} d(x, \underline{w}, \widehat{x}, \overline{w}) \\ d(\widehat{x}, \overline{w}, x, \underline{w}) \end{bmatrix} \tag{12}$$

is the *embedding system relative to d* and e is the *embedding function relative to d*. Note that the embedding system contains no disturbances as d is evaluated only at the extrema of $\mathcal{W} = [\underline{w}, \overline{w}]$.

We next show how finite-time reachable sets for nondeterministic systems as in (11) are efficiently over-approximated via a single simulation of the related embedding system (12). We denote by $\Phi(T;\, x, \mathbf{w})$ the state of (11) reached at time $T \geq 0$ when starting from state x at time 0 and when evolving subject to the disturbance signal $\mathbf{w} : [0,\, T] \to \mathcal{W}$. We assume always that disturbance signals as in $\mathbf{w}(t)$ are piecewise continuous in t and that the vector field F is Lipschitz continuous so that, in particular, $\Phi(T;\, x, \mathbf{w})$ is unique when it exists. Additionally, we denote by

$$R(T;\, x) := \{\Phi^F(T;\, x,\, \mathbf{w}) \in \mathcal{X} \mid \mathbf{w} : [0,\, T] \to \mathcal{W}\} \tag{13}$$

the time-T reachable set of (11) from initial state $x \subseteq \mathcal{X}$.

We denote by $\Phi^e(T; a)$ the unique state of (12) reached at time $T \geq 0$ when starting from state $a \in \mathcal{X} \times \mathcal{X}$ at time 0, and we abuse notation slightly so that $\Phi^e_T(x) := \Phi^e(T; (x, x))$ for $x \in \mathcal{X}$.

Proposition 1. *[1] Let* (11) *be mixed monotone with respect to d and denote by e the embedding function relative to d. If $\Phi^e_t(x) \in \mathcal{X} \times \mathcal{X}$ for all $0 \leq t \leq T$ then $R(T; x) \subseteq [\![\Phi^e_T(x)]\!]$.* ∎

Proposition 1 now implies that the reachable set of (11) is efficiently over-approximated using a single simulation of the deterministic embedding system (12): a simulation of the embedding system for time horizon t, starting from state (x, x), identifies a hyperrectangular over-approximation of $R(t; x)$ where the largest and smallest points in the rectangular approximation are taken to be the first n and last n coordinates of the simulation endpoint $\Phi^E_t(x)$, respectively.

4 ALGORITHMIC IMPLEMENTATION

In this section, we present the proposed RTA mechanism that leverages prior work [2], which was demonstrated on an academic example of three platooned vehicles. We especially highlight important novelties to enable implementation on a small low-cost offboard computer with limited computing power at fast controller update rates suitable for RTA of UAVs.

4.1 Algorithm Overview

We next turn our attention to affine-in-control and disturbance nondeterministic dynamical systems

$$\dot{x} = f(x) + g_1(x)u + g_2(x)w \tag{14}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ is the state, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input and $w \in \mathcal{W} \subset \mathbb{R}^p$ is the disturbance. Additionally, we pair the system (14) with unsafe set of system states $\mathcal{X}_u \subset \mathcal{X}$ which is to be avoided.

For the purposes of this exposition, we assume given a backup feedback controller $\mathbf{u}^{\mathrm{b}} : \mathcal{X} \to \mathcal{U}$ as defined previously in Section 3.1 and constructed later in Section 5.2. The controller $\mathbf{u}^{\mathrm{b}}$ renders robustly forward invariant a backup safe subregion $S_{\mathrm{b}} = \{x \in \mathcal{X} \mid h(x) \geq 0\} \subset \mathcal{X} \setminus \mathcal{X}_{\mathrm{u}}$ where $h : \mathbb{R}^n \to \mathbb{R}$ is assumed to be a continuously differentiable and concave function. The last assumption in our approach is that the closed-loop backup dynamics

$$\dot{x} = F^{\mathrm{b}}(x, w) := f(x) + g_1(x)\mathbf{u}^{\mathrm{b}}(x) + g_2(x)w \tag{15}$$

are mixed monotone with respect to some known decomposition function d. Constructing d is the subject of Section 5.3.

For a time horizon T, define

$$\gamma^{\mathrm{ideal}}(T;\, x) := \inf_{z \in [\![\Phi^e_T(x)]\!]} h(z) = \min_{z \in \langle\!\langle \Phi^e_T(x) \rangle\!\rangle} h(z), \tag{16}$$

where the second equality comes from the concavity on h. Given a fixed backup horizon T_{b}, further define

$$\Psi^{\mathrm{ideal}}(x) := \sup_{0 \leq \tau \leq T_{\mathrm{b}}} \gamma^{\mathrm{ideal}}(\tau;\, x). \tag{17}$$

We aim to find a control input $\mathbf{u}^{\mathrm{RTA}}(x)$ that filters a desired controller $\mathbf{u}^{\mathrm{d}}(x)$ by assessing the backup trajectory. Defining by $R_{\mathrm{b}}(T;\, x)$ the time-T reachable set of (15) as in (13), if $R_{\mathrm{b}}(T;\, x) \subseteq S^{\mathrm{b}}$ for some $T \leq T_{\mathrm{b}}$, then there exists a time in the backup horizon where all possible disturbances lead to trajectories in the safe subregion. We then adjust the desired controller $\mathbf{u}^{\mathrm{d}}(x)$ depending on how close $R_{\mathrm{b}}(T;\, x)$ is to the boundary of the safe subregion and how sensitive $R_{\mathrm{b}}(T;\, x)$ to perturbations on the state x. CBF algorithms are well suited for this task, but require the utilized functions to be differentiable, whereas γ^{ideal} and Ψ^{ideal} are generally not differentiable due to the minimum operator in (16). Therefore, we use a numerically stable continuously differentiable soft-min function known as the Log-Sum-Exponential (LSE) that approximates $\min \mathcal{S}$. For some fixed parameter $p > 0$ that controls the tightness of the approximation, define

$$\mathrm{LSE}(\mathcal{S}) = -\frac{1}{p} \log \sum_{s \in \mathcal{S}} \exp(-p \cdot s). \tag{18}$$

We further denote the LSE over a set of evaluations of a barrier function h on the corners of a hyperrectangle a by

$$\mathrm{LSE}_h(a) := \mathrm{LSE}(\, \{h(z) \mid z \in \langle\!\langle a \rangle\!\rangle\, \}). \tag{19}$$

The LSE provides a differentiable relaxation for γ^{ideal} and Ψ^{ideal} which we denote by γ and Ψ given by

$$\gamma(t;\, x) := \mathrm{LSE}_h(\Phi^e_t(x)) \tag{20}$$

$$\Psi(x) := \sup_{0 \leq \tau \leq T_{\mathrm{b}}} \gamma(\tau;\, x). \tag{21}$$

Differentiating $\Psi(x)$ with respect to x, we have

$$\frac{\partial \Psi}{\partial x}(x) = \frac{\partial \gamma}{\partial x}(t^*(x), x) = \frac{\partial \mathrm{LSE}_h}{\partial a}(\Phi^e_{t^*}(x)) \frac{\partial \Phi^e_{t^*}}{\partial x}(x) \tag{22}$$

where $t^*(x)$ is the time achieving the maximum in the right-hand side of (21) and the first equality holds by [15, Theorem 1]. The derivative $\frac{\partial \mathrm{LSE}_h}{\partial a}(\cdot)$ is computed from (18) and (19) and is evaluated at the embedding state $\Phi^e_{t^*}(x)$ at time $t^*(x) \leq T_{\mathrm{b}}$. The derivative $\frac{\partial \Phi^e_{t^*}}{\partial x}(x)$ is computed efficiently via a sensitivity matrix $S(t) \in \mathbb{R}^{2nxn}$ as explained in, *e.g.*, [21]. In particular, $S(t) = \frac{\partial \Phi^e_t}{\partial x}(x)$ where $S(t)$ is computed from the matrix differential equation

$$\frac{dS}{dt}(t) = \frac{\partial e}{\partial (x, \widehat{x})}(\Phi^e_t(x)) S(t) \tag{23}$$

with $S(t_0) = \begin{bmatrix} I & I \end{bmatrix}^T$ where $\frac{\partial e}{\partial (x, \widehat{x})}(\Phi^e_t(x))$ is the Jacobian matrix of the embedding system with respect to the embedding state.

Algorithm 1 Runtime Assurance for Nondeterministic Control Systems

input : Current State $x \in \mathcal{X}$.
: Desired control policy $\mathbf{u}^{\mathrm{d}} : \mathcal{X} \to \mathbb{R}^m$.
: Previous maximizer time $t^*_{k-1} \in \mathbb{R}$.
output : Assured control input $\mathbf{u}^{\mathrm{RTA}} \in \mathcal{U}$.
predefined : Class-$\mathcal{K}$ function $\alpha : \mathbb{R} \to \mathbb{R}$.

1: **function** $\mathbf{u}^{\mathrm{RTA}}$ =RTA($\mathbf{u}^{\mathrm{d}}$, x, t^*_{k-1})
2: **if** t^*_{k-1} initialized **then**
`timevec` $\leftarrow$ `Point_Dist`(t^*_{k-1})
3: **else** `timevec` $\leftarrow$ `linspace`$(0, T_{\mathrm{b}})$
4: **for all** $t \in$ `timevec` **compute** Φ^e_t from (12)
5: $\Gamma \leftarrow \varnothing$
6: **for all** $t \in$ `timevec` **append** $\gamma(t; x)$ from (20) **to** Γ
7: $[\Psi(x), \texttt{max\_psi\_indx}] \leftarrow \texttt{max}(\Gamma)$
8: $t^* \leftarrow$ `timevec[max_psi_indx]`
9: **compute:** $\frac{\partial \Psi}{\partial x}(x)$ as in (22)–(23)
10: **compute:** $u^* = \arg\min_{u \in [\underline{u}, \overline{u}]} ||u - u^{\mathrm{d}}||_2^2$
s.t. $\frac{\partial \Psi}{\partial x}(x)(f(x) + g_1(x)u + g_2(x)w) \geq -\alpha(\Psi(x))$
$\forall w \in \langle\!\langle \underline{w}, \overline{w} \rangle\!\rangle$
11: **if** Program feasible **then return** u^*
12: **else return** $\mathbf{u}^{\mathrm{b}}(x)$
13: **end function**

4.2 Pseudocode Implementation

We provide a pseduocode implementation of the proposed RTA mechanism in Algorithm 1.

In Line 2 of Algorithm 1, we propose a method for significantly reducing the number of computations required to identify the maximizing time t^* by leveraging the fact that t^* varies continuously along embedding system trajectories. That is, given t^*_k, the maximizer of (21) at the k^{th} iteration of Algorithm 1, then the next maximizer t^*_{k+1} will be *close* to t^*_k provided the controller update rate is small. For this reason, we reduce the search space of Lines 4 and 6 by considering a subset of `linspace`$(0, T_{\mathrm{b}})$, *i.e.*, replacing `linspace`$(0, T_{\mathrm{b}})$ by `Point_Dist`(t^*_{k-1}) if t^*_{k-1} is initialized returns a set of points clustered about t^*_{k-1}. In the experimental demonstration appearing in the next section, we employ this technique using a point distribution algorithm that provides a set of logarithmically distributed samples as outlined in Algorithm 2, centered at t^*_{k-1}. The logarithm-based point distribution algorithm requires defining the number of points desired N_{b} on the prediction horizon T_{b}, the timestep used for the discrete integration of the embedding system Δt, and a power factor η and constant multiplier μ that control the desired distribution spacing. In the algorithm the variable n_{r} and n_{l} store the number of points to the right and left of t^*_{k-1} respectively. The variables ζ_{r} and ζ_{l} are used in the logarithm increment calculation for neighboring points.

Algorithm 2 Efficient logarithm-based point distribution algorithm

input : Argument maximizer from (21) $t^*_{k-1} \in [0, T_b]$.
output: Point distribution array $\Gamma \in \mathbb{R}^{N_b}_{\geq 0}$
1: **function** Γ = `Point_Dist`(t^*_{k-1})
2: $n_{\mathrm{r}} \leftarrow \texttt{round}\Big(\frac{1}{2}(N_{\mathrm{b}} - 1)\Big(1 - \Big(\frac{2t - T_{\mathrm{b}}}{T_{\mathrm{b}}}\Big)^{\eta}\Big)\Big)$
3: $n_{\mathrm{l}} \leftarrow N_{\mathrm{b}} - n_{\mathrm{r}}$
4: $\zeta_{\mathrm{l}} = \frac{1}{n_{\mathrm{l}-1}}(\exp(\mu t^*_{k-1}) - 1)$
5: **for** $i = 0, ..., n_l - 1$ **do**
6: Γ.`insert`(`round`$(\frac{log(1 + i\zeta_{\mathrm{l}})}{\mu})\Delta t)$
7: **if** $n_{\mathrm{r}} \neq 0$ **then**
8: $\zeta_{\mathrm{r}} = \frac{1}{n_{\mathrm{r}}}(\exp(\mu(T_{\mathrm{b}} - t^*_{k-1}) - 1)$
9: **for** $j = n_r - 1, n_r - 2, ..., 0$ **do**
10: Γ.`insert`$(T_{\mathrm{b}} -$ `round`$(\frac{log(1 + j\zeta_{\mathrm{r}})}{\mu})\Delta t)$
11: **end function**

5 PROBLEM STATEMENT FOR DUAL MULTIROTOR CASE STUDY

Controlling multiple UAVs operating in close proximity to each other with environmental disturbances is challenging. The proposed solution in this work is a RTA mechanism that filters a potentially unsafe control input at runtime in order to guarantee safety. In this section, we first define the equations of motion for a *dual planar multirotor* system that serves as the primary case study of the paper. Then, we propose a backup control policy for this system that passively attracts the multirotors to a predefined safe displacement distance based on a saturated dual-mass-spring-dampener model and we provide a decomposition function for the dual planar multirotor system under the dual-mass-spring-dampener backup control policy. Finally, we demonstrate the effectiveness of the proposed Algorithm 1 with a collision avoidance flight demonstration of two quadrotor UAVs. The main limitation is from scaling the state space where the largest time complexity is a result of computing the barrier function over all corners of the hyperrectangle, which is of order $O(2^n)$. The reachability analysis is very efficient and only requires solving an ordinary differential equation as opposed to solving for the minimizer of the barrier function over a hyperrectangle. Therefore, each additional agent may run a version of the algorithm as long as the state space is of appropriate size for fast enough computations.

5.1 Dynamics of Dual Planar Multirotor System

We consider two multirotors of mass m_1 and m_2 each fixed to the Y-Z plane, which we denote as the *dual planar multirotor* system and consists of two copies of the planar multirotor UAV model from Example 1 in Section 3.1. The state, control, and constant variables are indexed by 1 and 2 for each respective multirotor, and we replace the four positional states y_1, y_2, z_1, and z_2 with two relative displacement states $\delta_y = y_2 - y_1$ and $\delta_z = z_2 - z_1$. We denote the velocity in y and z by v_{y} and v_z. We also denote the roll angle of each multirotor by ϕ_1, ϕ_2 and the standard local gravity g. The right-handed axes conventions used in this paper are North-East-Down (NED) for the inertial reference frame and Forward-Right-Down (FRD) for the vehicle body-fixed frame. We

make an observation that a low-level body-rate controller tracks the body-rates and collective thrust at timescales much faster than the rest of the states and we therefore make the assumption that we have instant control of the body-rates and thrust of each multirotor. This leads to the chosen control inputs of collective thrust $\tau \in \mathbb{R}$ and a desired roll rate $\omega_{\mathrm{x}} \in \mathbb{R}$.

Further, we introduce a six-dimensional disturbance input to the linear acceleration and roll rate components to capture unmodeled dynamics and deviations between the dual planar multirotor model and a full six degree of freedom model. We assume the disturbance is bounded so that the disturbance space is $\mathcal{W} := [\underline{w}, \overline{w}]$ for $\underline{w}, \overline{w} \in \mathbb{R}^6$ and $\underline{w}_i \leq \overline{w}_i$ for all i. The resulting eight-dimensional dual planar multirotor dynamics are given by

$$\begin{bmatrix} \dot{v}_{\mathrm{y1}} \\ \dot{v}_{\mathrm{y2}} \\ \dot{v}_{\mathrm{z1}} \\ \dot{v}_{\mathrm{z2}} \\ \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\delta}_{\mathrm{y}} \\ \dot{\delta}_{\mathrm{z}} \end{bmatrix} = \begin{bmatrix} m_1^{-1}(\tau_1 \sin(\phi_1)) + w_1 \\ m_2^{-1}(\tau_2 \sin(\phi_2)) + w_2 \\ m_1^{-1}(m_1 g - \tau_1 \cos(\phi_1)) + w_3 \\ m_2^{-1}(m_2 g - \tau_2 \cos(\phi_2)) + w_4 \\ \omega_{\mathrm{x1}} + w_5 \\ \omega_{\mathrm{x2}} + w_6 \\ v_{\mathrm{y2}} - v_{\mathrm{y1}} \\ v_{\mathrm{z2}} - v_{\mathrm{z1}} \end{bmatrix}. \tag{24}$$

Note that the dual planar multirotor dynamics (24) suffer from the same relative degree issue explored in Example 1 for the case of a single multirotor, and for this reason standard CBF-based controllers can generally not be employed with (24) to assure system safety. The RTA approach taken in this work employs instead online reachability analysis under a verified safe backup controller—see, *e.g.*, (9)—in order to avoid the relative degree issue detailed above.

5.2 Backup Controller for Dual Planar Multirotors

In the context of the dual planar multirotor system (24), our objective is to ensure a safe displacement between the multirotors at all times. Safety of the system (24) is defined in terms of a constraint on the relative displacements δ_y, δ_z: we consider the system safe if (δ_y, δ_z) remains within a ball of a nominal safe displacement that does not contain $(\delta_y, \delta_z) = (0, 0)$. Note that a safe set defined with respect only to displacement, as above, cannot generally be used directly to obtain a standard CBF-based safe controller due to the system having a nonexistent relative degree at particular points as demonstrated in Example 1 of Section 3.1. As discussed in the Introduction and detailed in Section 4, the RTA mechanism proposed in this paper requires a nominal backup control strategy that guarantees safety in a (generally conservative) subset of the state space. For the backup controller, we propose desired forces in the inertial y and z directions for each multirotor that imitate a nonlinear dual mass spring and friction system with spring constant K_{s} for controlling the maximum force acted on by the springs and saturation σ for controlling the distance at which the springs saturate. The formulation for the desired forces on each multirotor to imitate the spring and friction dynamics are

$$\begin{aligned} f_{\mathrm{y1}} &= K_{\mathrm{s}} \tanh[\sigma(\delta_y - D_{\mathrm{y}})] - b v_{\mathrm{y1}} \\ f_{\mathrm{y2}} &= -K_{\mathrm{s}} \tanh[\sigma(\delta_y - D_{\mathrm{y}})] - b v_{\mathrm{y2}} \\ f_{\mathrm{z1}} &= K_{\mathrm{s}} \tanh[\sigma(\delta_z - D_{\mathrm{z}})] - m_1 g - b v_{\mathrm{z1}} \\ f_{\mathrm{z2}} &= -K_{\mathrm{s}} \tanh[\sigma(\delta_z - D_{\mathrm{z}})] - m_2 g - b v_{\mathrm{z2}} \end{aligned} \tag{25}$$

with D_{y} and D_{z} as the nominal safe backup distance in y and z respectively. The total thrust desired for each multirotor is computed by constructing a desired force vector and projecting it onto the multirotor body-fixed z-axis as formulated by

$$\begin{aligned} \tau_1 &= f_{\mathrm{y1}} \sin(\phi_1) - f_{\mathrm{z1}} \cos(\phi_1) \\ \tau_2 &= f_{\mathrm{y2}} \sin(\phi_2) - f_{\mathrm{z2}} \cos(\phi_2). \end{aligned} \tag{26}$$

The desired roll rate is formulated so that the thrust vector of the vehicle tracks the desired force vector,

$$\begin{aligned} w_{\mathrm{x1}} &= K_r \left(\frac{f_{\mathrm{z1}}}{\sqrt{f_{\mathrm{z1}}^2 + f_{\mathrm{y1}}^2}} \sin(\phi_1) + \frac{f_{\mathrm{y1}}}{\sqrt{f_{\mathrm{z1}}^2 + f_{\mathrm{y1}}^2}} \cos(\phi_1) \right) \\ w_{\mathrm{x2}} &= K_r \left(\frac{f_{\mathrm{z2}}}{\sqrt{f_{\mathrm{z2}}^2 + f_{\mathrm{y2}}^2}} \sin(\phi_2) + \frac{f_{\mathrm{y2}}}{\sqrt{f_{\mathrm{z2}}^2 + f_{\mathrm{y2}}^2}} \cos(\phi_2) \right). \end{aligned} \tag{27}$$

Hereafter, we denote the closed-loop dual planar multirotor dynamics (24) under the backup control policy (25)–(27) as in (15), where we now use the shorthand notation

$$x = (v_{\mathrm{y1}}, v_{\mathrm{y2}}, v_{\mathrm{z1}}, v_{\mathrm{z2}}, \phi_1, \phi_2, \delta_y, \delta_z) \tag{28}$$

$$u = (\tau_1, \tau_2, \omega_{\mathrm{x1}}, \omega_{\mathrm{x2}}) \tag{29}$$

to describe the system state and control input respectively.

We construct a verified-but-small backup subregion by considering a local linearization of (15). The linearized system is asymptotically stable to $x^* = (0, 0, 0, 0, 0, 0, D_{\mathrm{y}}, D_{\mathrm{z}})$, and we obtain a quadratic Lyapunov function $V(\tilde{x}) = \tilde{x}^T P \tilde{x}$ for the linearized system where $\tilde{x} = x - x^*$ for P obtained from the linearization. Therefore,

$$S_{\mathrm{b}} = \{\tilde{x} \in \mathbb{R}^8 \mid V(\tilde{x}) \leq \epsilon\} \tag{30}$$

is a robustly forward invariant safe set (defined formally below) for the nonlinear backup dynamics (15) for some $\epsilon > 0$ determined numerically.

Proposition 2. *Algorithm 1 solves the problem statement for the dual multirotor case study.* ∎

Proof sketch of Proposition 2. The RTA receives as inputs the current system state x and the desired control input u^{d}. A linear spacing of points in $[0, T_{\mathrm{b}}]$ is obtained in Line 2, and stored in the point distribution array `timevec`. For each time t in `timevec`, the time-t reachable set of the backup dynamics (15) is computed in Line 3 using the mixed monotonicity property. For each hyperrectangular reachable set approximation, the LSE of h evaluated over the corners of the hyperrectangle is computed in Line 5 and stored in Γ. In Line 6, $\Psi(x)$ from (21) is computed as the maximum element of Γ, and the maximizing time t^* is assigned in Line 7. Line 8 computes the derivative of $\Psi(x)$ with respect to x using sensitivity analysis. Finally, an assured control input is computed in Line 9 using a control barrier function type formulation, as discussed above. If ever the optimization problem in Line 9 is infeasible, which

can only happen in the rare case that the state is such there is a significant difference between the LSE approximation of the barrier minimum and the true minimum, the backup control input is applied, guaranteeing the system returns to the safe region S^{b} and avoids the unsafe region $\mathcal{X}_{\mathrm{u}}$ along the way. □

5.3 Decomposing The Closed-Loop Multirotor Dynamics

A key assumption in our proposed RTA mechanism is that the backup dynamics (15) are mixed monotone with respect to a known decomposition function $d(x, w, \widehat{x}, \widehat{w})$. Several general methods exist for constructing decomposition functions for continuous-time dynamical systems: see [18] for decomposition functions derived from bounds on the system Jacobian matrix; see [1] for decomposition functions for systems defined by polynomial vector fields; and see [5] for a decomposition function construction defined as a pointwise-in-time optimization problem. All of these methods overapproximate the reachable set with various levels of conservatism. The pointwise-in-time optimization provides the tightest approximation of the reachable set with a hyperrrectangle and therefore is the least conservative. The conservatism of the overapproximation can be further reduced via a transformation [4] from a hyperrectangle to a parallelotope. In this work, we decompose the closed-loop dual planar multirotor dynamics (15) using the procedure detailed in [3] whereby the system model is first represented as an interconnection amongst several subsystems and then a decomposition function for (15) is formed from individual decomposition functions for the subsystems in the interconnection.

Lemma 1. The closed-loop dual planar multirotor dynamics (15) are mixed monotone. ■

We sketch the proof of Lemma 1 by constructing the first entry of decomposition function d for the closed-loop dual planar multirotor dynamics (15). The remaining entries either following similarly or from existing standard techniques.

Decomposition functions are generally constructed elementwise, as the conditions on d_i for some $i \in \{1, \cdots, n\}$ specified in Definition 1 are dependent only on the i^{th} entry of F^{b}. As such, the first entry of the decomposition function d is derived from the first entry of the backup dynamics $\dot{x} = F^{\mathrm{b}}(x, w)$, given by

$$F_1(x, w) = m_1^{-1}(\tau_1(x)\sin(\phi_1)) + w_1 \tag{31}$$

where $\tau(x)$ is given by (26). The result [3, Theorem 1] now implies that the first entry of d is given by

$$d_1(x, w, \widehat{x}, \widehat{w}) = \\ m_1^{-1} d^{u_1u_2}(\mathcal{T}_1(x, \widehat{x}), d^{\sin}(\phi_1, \widehat{\phi}_1), \mathcal{T}_1(\widehat{x}, x), d^{\sin}(\widehat{\phi}_1, \phi_1)) + w_1 \tag{32}$$

where $d^{u_1u_2}$ is a decomposition function for

$$\dot{x} = u_1u_2 \tag{33}$$

and is given by

$$d^{u_1u_2}(u, \widehat{u}) = \begin{cases} \min\{u_1u_2, \widehat{u}_1u_2, u_1\widehat{u}_2, \widehat{u}_1\widehat{u}_2\} & \text{if } u \preceq \widehat{u} \\ \max\{u_1u_2, \widehat{u}_1u_2, u_1\widehat{u}_2, \widehat{u}_1\widehat{u}_2\} & \text{if } \widehat{u} \preceq u, \end{cases} \tag{34}$$

where $d^{\sin}$ is a decomposition function for

$$\dot{x} = \sin(u) \tag{35}$$

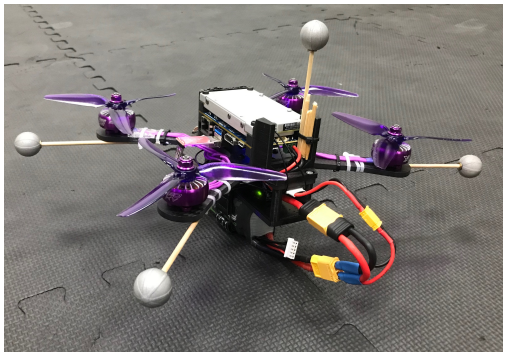


Figure 3: Photograph of a 700g quadrotor used in the flight experiment. A Jetson TX2 single-board computer running ROS2 is fixed on top of the vehicle and is connected to a flight controller under the vehicle running PX4 autopilot for estimation and motor control.

and is given by

$$d^{\sin}(u, \widehat{u}) = \sin(u) + u - \widehat{u} \tag{36}$$

and where $\mathcal{T}_1(x, \widehat{x})$ is an inclusion function for $\tau_1(x)$, as defined in [16].

6 HARDWARE DEMONSTRATION

In this section, we demonstrate the proposed algorithm in a real world flight experiment with two 250 scale 700g quadrotors shown in Figure 3, each with a Jetson TX2 on-board computer and a flight controller (FC) running PX4 autopilot with an inertial measurement unit (IMU). The on-board computer runs ROS2 and receives external position measurements from OptiTrack motion capture cameras. The external position measurements are sent to the FC and processed by the PX4 extended Kalman filter estimation module. The primary controller and our proposed RTA algorithm is run on ROS2 and sends body-rate and thrust commands to the FC. The flight controller then uses a low-level body-rate controller to output pulse-width modulation (PWM) commands to the electronic speed controller and finally to the individual motors.

For this demonstration we constrain the two quadrotors shown in Figure 4 to remain within the Y–Z plane and maintain a yaw angle of zero. This results in inter-vehicle safety for the hardware demonstration a $2D$ problem as in the dual planar multirotor system introduced in Section 5. We then use the backup controller defined in Section 5.2 with parameters $K_{\mathrm{s}} = 60$, $\sigma = 0.06$, $b = 7$, $K_{\mathrm{r}} = 6$, $D_{\mathrm{y}} = 1\mathrm{m}$, $D_{\mathrm{z}} = 1\mathrm{m}$. For the logarithm-based point distribution algorithm we use $\eta = 3$, $\mu = 0.01$, $\Delta t = 10\mathrm{ms}$, and $N_{\mathrm{b}} = 9$. The bounded disturbance in the linear acceleration is $\mathcal{W}_{1,2} = [-5 \times 10^{-7}, 5 \times 10^{-7}]\mathrm{m\,s^{-2}}$ for the Y-axis and $\mathcal{W}_{3,4} = [-1 \times 10^{-4}, 1 \times 10^{-4}]\mathrm{m\,s^{-2}}$ for the Z-axis. The disturbance in the roll rate is $\mathcal{W}_{5,6} = [-1 \times 10^{-5}, 1 \times 10^{-5}]\mathrm{s^{-1}}$ which mainly is associated with the center of gravity not coinciding with the center of thrust. For demonstration, the unsafe primary controller is constructed in a similar approach to the backup controller with the relative position terms δ_{y} and δ_{z} being replaced with absolute position, although any primary controller would be valid. The verified backup subregion is constructed from

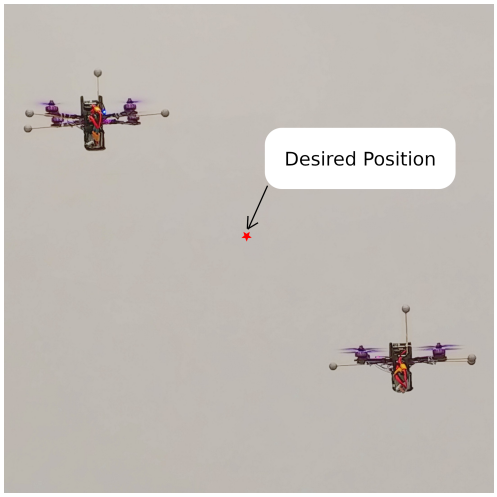
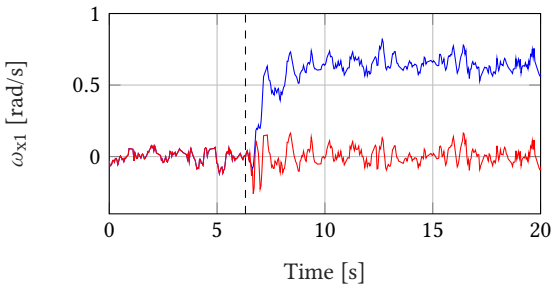


Figure 4: Photograph of the two quadrotors during the flight experiment. The vehicles are both commanded by an unsafe controller with a desired position that nominally results in a collision. The proposed RTA algorithm supervises the vehicles and filters the unsafe controller resulting in a safe displacement while the vehicles are centrally positioned at the desired unsafe position.

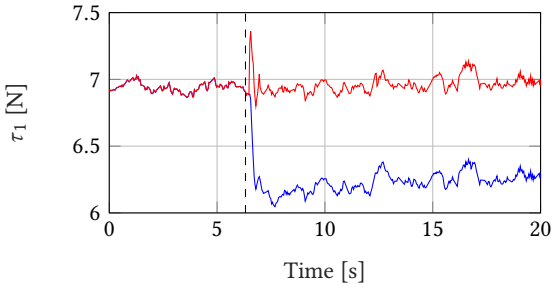
a robustly forward invariant safe set S_b as previously mentioned in Section 5.2.

The flight demonstration is performed by initially taking off and settling to the defined safe displacement distance D_y and D_z. Then, our proposed algorithm is enabled to supervise the unsafe primary position controller. Finally, an unsafe position command is sent to the primary controller that would nominally lead to a collision at $(y, z) = (0, -2)$m. Choosing when to intervene is dependent on the Class-$\mathcal{K}$ function α in Algorithm 1 and for this experiment we use $\alpha(\Psi(x)) = 100\Psi(x)^3$. To solve the optimization problem in Algorithm 1 we use an operator splitting quadratic program solver [22] that achieves convergence in approximately 500µs. We also use a logarithm-based point distribution algorithm to unevenly space nine points along a backup horizon of $T_b = 1$s within 7µs so that the majority of points are near t^*_{k-1} as stated in Section 4.2 for the purposes of reducing computation time. The execution time for Algorithm 1 for the hardware demonstration is plotted as a histogram in Figure 8. The execution time with the greatest frequency of occurence is between 2.5ms–3.0ms for the dual planar multirotor dynamics with the chosen barrier function and distribution algorithm parameters described above. This enables achieving control update rates with the RTA mechanism of up to 250Hz, although we run the control loop at 100Hz for the flight demonstration and observe satisfactory results.

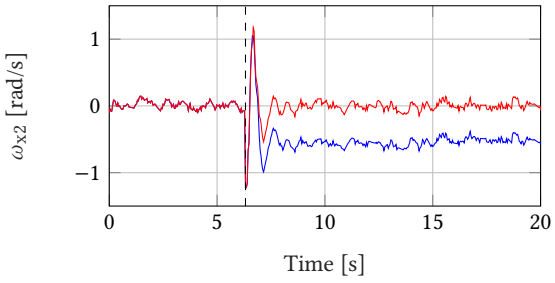
As previously mentioned, the proposed RTA algorithm filters the desired control inputs when necessary to avoid collision, as shown in Figure 5. In particular, upon activating the unsafe position control commands at time $t = 6.1$s, the proposed RTA algorithm allows the desired control inputs to be applied for the safe position control commands and shortly after activating the unsafe position commands until the $\dot{\Psi}$ lower bound from the CBF constraint in



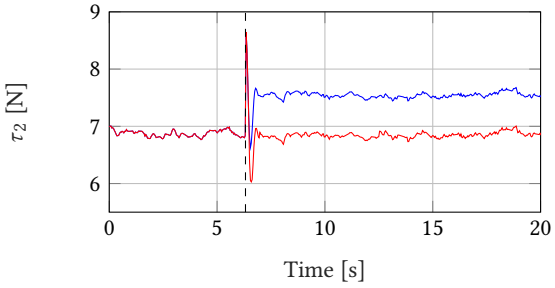
(a) Quadrotor 1 roll rate command vs. experiment time.



(b) Quadrotor 1 thrust command vs. experiment time.

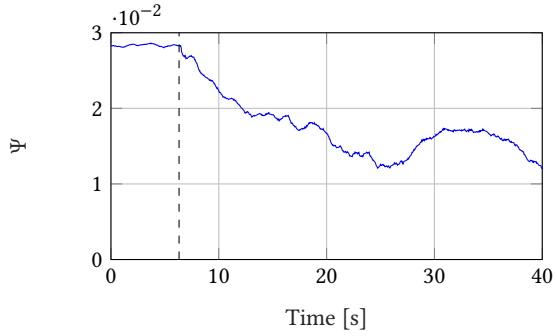


(c) Quadrotor 2 roll rate command vs. experiment time.

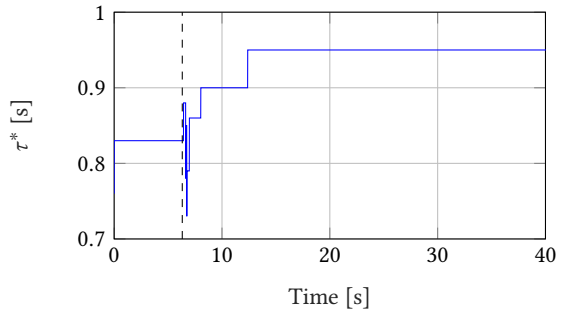


(d) Quadrotor 2 thrust command vs. experiment time.

Figure 5: Case Study: applied and desired roll rate and thrust command plots for the dual quadrotor experiment. The desired input from the primary controller is plotted in blue and applied input of the RTA algorithm is in red. The black dashed line represents the activation of the unsafe controller.



(a) Barrier evaluation $\Psi(x)$ vs. experiment time.



(b) Argument solution t^* for (21) vs. experiment time. At time t^* the systems reachable set under the backup control policy in contained within S_b

Figure 6: Case Study: barrier evaluation $\Psi(x)$ and corresponding maximizing time t^* for the dual quadrotor experiment. The maximum barrier evaluation is $\Psi = 0.03$ and the quadrotors are considered safe if $\Psi \geq 0$. For this particular example, the value of Ψ will continue converging to zero if the experiment time is extended. The backup prediction horizon is $T_b = 1$s. The black dashed line represents the activation of the unsafe position control commands.

Algorithm 1 is not satisfied by the desired control inputs. The mathematical description of the lower bound is defined in the function $\alpha(\Psi(x)) = 100\Psi(x)^3$ which is dependent on $\Psi(x)$. Notice in Figure 6a, the value of $\Psi(x)$ decreases after activating the unsafe position control commands. As $\Psi(x)$ decreases this further constrains the lower bound on $\dot{\Psi}$ and we see that $\dot{\Psi}$ is initially decreasing quickly upon activating the unsafe position control commands at time $t = 6.1$s, but the convergence decreases throughout the experiment as a result of the lower bound constraint. Ideally, the value of $\Psi(x)$ will converge to zero if the experiment is continued.

7 CONCLUSION

This work presents an efficient algorithm for runtime assurance of control systems with disturbance and demonstrates the approach with a hardware flight demonstration of two quadrotors avoiding collision. The proposed runtime assurance controller computes reachable sets under a backup control law to ensure the system remains within proximity of a safe backup subregion and does not enter an unsafe region. The experiment is implemented on embedded hardware and is demonstrated by a dual planar multirotor system case study with a controller update rate of 100Hz. We demonstrate that the multirotors remain a safe displacement distance apart with our RTA algorithm while commanded by a nominally unsafe controller.

ACKNOWLEDGEMENT

This work is supported in part by the NASA University Leadership Initiative (ULI) under grant number 80NSSC20M0161 and the Air Force Office of Scientific Research under award FA9550-19-1-0015.

REFERENCES

[1] M. Abate and S. Coogan. 2020. Computing Robustly Forward Invariant Sets for Mixed-Monotone Systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*. 4553–4559. https://doi.org/10.1109/CDC42340.2020.9304461

[2] M. Abate and S. Coogan. 2020. Enforcing Safety at Runtime for Systems with Disturbances. In *2020 59th IEEE Conference on Decision and Control (CDC)*. 2038–2043. https://doi.org/10.1109/CDC42340.2020.9304203

[3] M. Abate and S. Coogan. 2021. Decomposition Functions for Interconnected Mixed Monotone Systems. In Submission.

[4] M. Abate and S. Coogan. 2021. Improving the Fidelity of Mixed-Monotone Reachable Set Approximations via State Transformations. In *2021 American Control Conference (ACC)*. 4674–4679. https://doi.org/10.23919/ACC50511.2021.9483264

[5] M. Abate, M. Dutreix, and S. Coogan. 2021. Tight Decomposition Functions for Continuous-Time Mixed-Monotone Systems With Disturbances. *IEEE Control Systems Letters* 5, 1 (2021), 139–144. https://doi.org/10.1109/LCSYS.2020.3001085

[6] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada. 2019. Control Barrier Functions: Theory and Applications. In *2019 18th European Control Conference (ECC)*. 3420–3431. https://doi.org/10.23919/ECC.2019.8796030

[7] A. D. Ames, J. W. Grizzle, and P. Tabuada. 2014. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control (CDC)*. 6271–6278. https://doi.org/10.1109/CDC.2014.7040372

[8] Stanley Bak, Taylor T. Johnson, Marco Caccamo, and Lui Sha. 2014. Real-Time Reachability for Verified Simplex Design. In *2014 IEEE Real-Time Systems Symposium*. 138–148. https://doi.org/10.1109/RTSS.2014.21

[9] Yuxiao Chen, Mrdjan Jankovic, Mario Santillo, and Aaron D Ames. 2021. Backup Control Barrier Functions: Formulation and Comparative Study. *arXiv preprint arXiv:2104.11332* (2021).

[10] S. Coogan. 2020. Mixed Monotonicity for Reachability and Safety in Dynamical Systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*. 5074–5085. https://doi.org/10.1109/CDC42340.2020.9304391

[11] T. Gurriet, M. Mote, A. D. Ames, and E. Feron. 2018. An Online Approach to Active Set Invariance. In *2018 IEEE Conference on Decision and Control (CDC)*. 3592–3599. https://doi.org/10.1109/CDC.2018.8619139

[12] T. Gurriet, M. Mote, A. Singletary, E. Feron, and A. D. Ames. 2019. A Scalable Controlled Set Invariance Framework with Practical Safety Guarantees. In *2019 IEEE 58th Conference on Decision and Control (CDC)*. 2046–2053.

[13] T. Gurriet, A. Singletary, J. Reher, L. Ciarletta, E. Feron, and A. Ames. 2018. Towards a Framework for Realizable Safety Critical Control through Active Set Invariance. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. 98–106. https://doi.org/10.1109/ICCPS.2018.00018

[14] Kerianne Hobbs, Mark Mote, Matthew Abate, Samuel Coogan, and Eric Feron. 2021. Run Time Assurance for Safety-Critical Systems: An Introduction to Safety Filtering Approaches for Complex Control Systems. arXiv:2110.03506 [cs.SY]

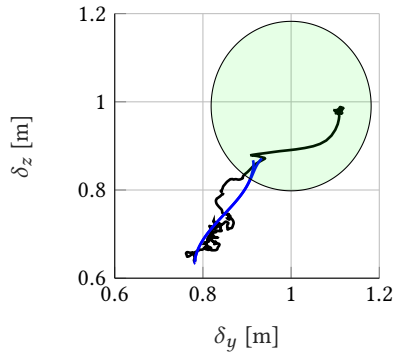
[15] William Hogan. 1973. Directional derivatives for extremal-value functions with applications to the completely convex case. *Operations Research* 21, 1 (1973), 188–209.

[16] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter. 2001. Applied interval analysis. Springer.

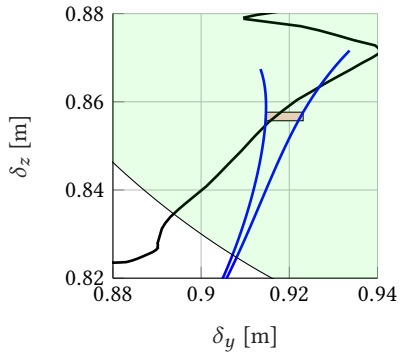
[17] Haye Kesteloo. 2017. *Drone carrying candy crashes into crowd, injuring six in Japan.* https://dronedj.com/2017/11/13/drone-carrying-candy-crashes-into-crowd-injuring-six-in-japan/

[18] Pierre-Jean Meyer, Alex Devonport, and Murat Arcak. 2019. TIRA: Toolbox for interval reachability analysis. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. 224–229.

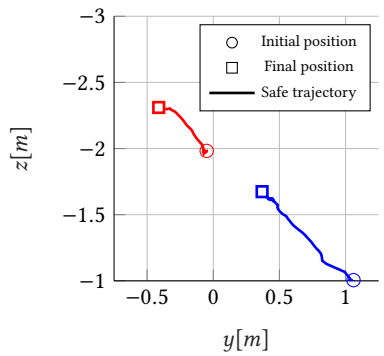
[19] Q. Nguyen and K. Sreenath. 2016. Exponential Control Barrier Functions for enforcing high relative-degree safety-critical constraints. In *2016 American Control Conference (ACC)*. 322–328. https://doi.org/10.1109/ACC.2016.7524935



(a) Trajectories of quadrotors in flight experiment in (δ_y, δ_z) coordinates.



(b) Figure 7a, magnified, as to show the reachable set approximation derived from the online application of Proposition 1.



(c) Absolute position trajectories of quadrotor 1 in red and quadrotor 2 in blue. Both vehicles are commanded to $(y, z) = (0, -2)$m.

Figure 7: Case Study: projection of the relative displacement state trajectories. Both vehicles are commanded to $(y, z) = (0, -2\text{m})$ which would nominally lead to a collision, *i.e.*, the displacement states converge to $(\delta_y, \delta_z) = (0, 0)$. The RTA mechanism applies the performance control input to the system until the reachable set over-approximation, shown in red, approaches the boundary of the backup sub-region S_b. At this point, the RTA mechanism filters the performance control input to ensure safety. The trajectory of the embedding system is shown in blue and a (δ_y, δ_z) projection of the backup sub-region S_b at time t^* is shown in green. The true state trajectory of the dual quadrotor system is illustrated in black.

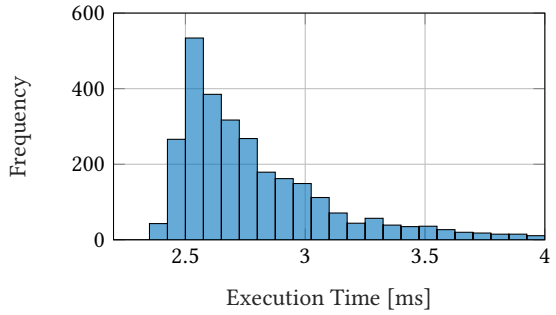


Figure 8: Case Study: Histogram of execution time for Algorithm 1 on a Jetson TX2. The greatest frequency of occurrence is within 2.5 and 3.0 ms. The highest execution time is 4.5 ms and the range of 4.0ms–4.5ms is omitted from the histogram for having a frequency of occurrence less than 10 out of about 3000 for the hardware demonstration.

[20] Jose German Rivera and Alejandro Andres Danylyszyn. 1995. *Formalizing the Uni-processor Simplex Architecture.* Technical Report. Carnegie Mellon University School of Computer Science.

[21] Hans Seywald and R.R. Kumar. 1996. Desensitized optimal trajectories. 93 (01 1996), 103–113.

[22] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. 2020. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation* 12, 4 (2020), 637–672. https://doi.org/10.1007/s12532-020-00179-2

[23] Xiao Tan, Wenceslao Shaw Cortez, and Dimos V. Dimarogonas. 2021. High-order Barrier Functions: Robustness, Safety and Performance-Critical Control. *IEEE Trans. Automat. Control* (2021), 1–1. https://doi.org/10.1109/TAC.2021.3089639

[24] W. Xiao and C. Belta. 2019. Control Barrier Functions for Systems with High Relative Degree. In *2019 IEEE 58th Conference on Decision and Control (CDC).* 474–479. https://doi.org/10.1109/CDC40024.2019.9029455