Safe Self-Triggered Control Based on Precomputed Reachability Sequences

ABSTRACT

Self-triggered controllers have the potential to improve the state-of-the-art of Cyber-Physical Systems (CPSs) by enhancing the performance of the underlying closed-loop control systems. However, a major concern in deploying a self-triggered controller in a safety-critical CPS is that the stabilizing self-triggered controller may not always guarantee the satisfaction of the safety constraints. We propose a self-triggered control scheme that deals with the safe scheduling of control tasks for uncertain continuous-time linear systems. We derive a computationally efficient scheduling function that computes an upper bound on the next sampling period as a function of the current state in the presence of additive disturbance. To deal with the conservativeness of the reachability analysis, we compute a large sequence of reachable sets offline and use these precomputed sets to derive a low-complexity online scheduling function that computes sufficiently large bounds in real-time. We evaluate our algorithm on three high-dimensional benchmark control systems, where two of the examples have a twelve-dimensional joint state plus feedback input. Experimental results demonstrate that our self-triggered control algorithm guarantees the safety of the closed-loop control system through negligible online computation, establishing the feasibility of its practical implementation.

1 INTRODUCTION

Feedback controllers are the core components of any safety-critical Cyber-Physical System (CPS). A feedback controller receives the state of the plant at some discrete time instants, computes the control signal based on the state, and transmits the control signal to the actuator [6]. A significant challenge in designing a feedback control system is to decide the time instants when the control-signal updates should happen to ensure the desired behavior of the system [8].

Traditionally, the digital implementation of a feedback control system adopts either a time-triggered or an event-triggered approach. In the time-triggered approach, a fixed period is decided for the control computation based on the worst-case requirement [6, 14]. The main benefit of a time-triggered feedback control loop is its easy implementation using a timer. Moreover, a well-developed theory of discrete-time systems can be used for the implementation. The downside of the time-triggered implementation is the possible over-computation of feedback input than what is necessary. This is because the control computation and the associated sensor-to-controller and controller-to-actuator communications happen periodically. Thus, even if the system can remain in a safe state with stable dynamics without feedback updates for a long time, the feedback controller still performs unnecessary computations. This can result in under-allocation of computation time for other computation tasks, or else waste significant computation and communication resources, thus leading to performance degradation.

To deal with the over-provisioning of the time-triggered implementation, event-triggered implementation was introduced as a mechanism to perform the control computation only when necessary [18, 30, 37]. In this scheme, the state of the plant is monitored continuously, and a triggering condition is evaluated with each monitored state. Once the state satisfies the triggering condition, the state of the plant is communicated to the controller, and the control computation is activated. Unlike the time-triggered implementation, the event-triggered implementation reduces the computation and communication load significantly without any adverse effect on the control performance. However, the drawback of this implementation is that it requires monitoring the system state continuously, which requires additional hardware and also may not be a viable solution for energy-constrained CPSs.

The limitations in time-triggered and event-triggered implementation of control systems motivated the control researchers to find an alternative avenue, which led to the invention of self-triggered control [4, 5, 31]. In self-triggered control, apart from updating the control signal at the current time, the controller also computes the future time point when the feedback needs to be updated. Typically between two control computations, the actuation signal is held equal to the previously computed control signal (constant hold implementation), and the plant evolves in open-loop without observing its state. But the design of a self-triggered control scheme ensuring desired behavior of the closed-loop system is significantly challenging.

Most works on self-triggered control have aimed to ensure stability of the closed-loop system [5, 20, 21, 26, 32, 33]. However, for a safety-critical CPS, an additional requirement is that the system does not violate safety constraints on the state during its operation. Though safety issues related to periodically sampled control systems and systems with continuous (without jump) feedback trajectories have been addressed widely in the literature [10, 27, 28, 34], the same for self-triggered control systems has received limited attention.

In this paper, we tackle the problem of safe scheduling self-triggered linear systems under unknown but bounded additive disturbance input. The bounded disturbance can over-approximate modeling errors, like linearization errors or environmental effects. We derive a scheduling function that computes an upper bound on the next sampling period as a function of the current state in the presence of additive disturbance. The scheduling function has to be evaluated fast in real-time so that the system performance is not degraded; we hence derive a very low complexity scheduling function.

While aiming at deriving a low-complexity scheduling function, we do not want the scheduling bound to be overly conservative. To find a safe upper bound on the sampling period in the presence of an additive disturbance, in principle, some reachability analysis techniques with high approximation accuracy [16, 24] can be used, as in [23], to maximize the scheduled bound. However, such accurate reachability analysis tends to have high computational complexity, which may be unsuitable for real-time computations, especially on

low-powered processors. So, in this paper, we propose a new self-triggered approach, which tries to improve both the speed of online reachability analysis and its accuracy for maximizing the safe upper bound on trigger time. We accomplish this by first pre-computing a large sequence of reachable sets offline which are encoded by scalar values. Then we use the pre-computed scalarized reachable sets to derive a low-complexity online scheduling function that determines large bounds in real time. In this regard, we state a new set-theoretic condition (Lemma 3.3) that enables the derivation of such a scheduling function.

Although previous approaches on self-triggered controller synthesis precompute offline the invariant set (or invariant functions) [9, 15, 17, 23, 35], the reachable sets in between sampling times that are also required for online scheduling are not computed offline. A significant difference in our approach is that apart from the invariant set, we also precompute offline the reachable sets in between sampling times that are also required for safe online scheduling. This improves both the speed and accuracy of online reachability analysis involved in self-triggered scheduling. In our experiments, we demonstrate a significant speedup in the computation of the safe sampling period upper bound by our approach, compared to using a highly accurate reachability analysis technique like [16].

The contributions of this paper are summarized below.

(1) We introduce a new approach of precomputing overapproximation of sets that are reachable between sampling periods and are required for scheduling, in addition to the invariant set. This allows us to derive less conservative and low complexity scheduling functions for self-triggered controllers.
(2) At a theoretical level, we derive a new set-theoretic condition (Lemma 3.3) for linear systems that can be used to derive a safe scheduling function based on precomputing reachable sets in between sampling times.
(3) We perform experiments on three high-dimensional benchmark control systems from literature, including two examples with 12 dimensional state plus feedback input, to demonstrate the applicability and scalability of our algorithm.

1.1 Notation

We denote integers by $\mathbb{Z}$, real numbers by $\mathbb{R}$ and complex numbers by $\mathbb{C}$. For $a \in \mathbb{R}$, we denote $\lceil a \rceil = \min\{i \in \mathbb{Z} \mid i \geq a\}$ (ceiling function). If $\Theta \subseteq \mathbb{R}$, then for any relation symbol $\bowtie \in \{\leq, \geq, <, >, =\}$ and $r \in \mathbb{R}$, we denote $\Theta_{\bowtie r} = \{a \in \Theta \mid a \bowtie r\}$. For $a, b \in \mathbb{R}$, we denote $[a, b] = \{c \in \mathbb{R} \mid a \leq c \leq b\}$, $[a, b) = \{c \in \mathbb{R} \mid a \leq c < b\}$, $(a, b] = \{c \in \mathbb{R} \mid a < c \leq b\}$ and $(a, b) = \{c \in \mathbb{R} \mid a < c < b\}$. For any $n, m \in \mathbb{Z}$ and a set of complex numbers $\Delta \subseteq \mathbb{C}$, the set of all n-dimensional column vectors with component elements from Δ is denoted Δ^n, while the set of all $n \times m$ dimensional matrices with component elements from Δ is denoted $\Delta^{n\times m}$. The number of rows and columns of a matrix $X \in \Delta^{n\times m}$ are denoted $rows(X) = n$ and $cols(X) = m$, respectively. The terms $|X|, |v|$ for $X \in \mathbb{C}^{n\times m}, v \in \mathbb{C}^n$ denote the matrix and the vector containing the absolute values of the elements of X, v, respectively. The infinity norm of the vector v is denoted $\|v\|_\infty = \max_{i=1}^n |v_i|$. A complex $n \times m$ matrix containing a complex number $\alpha \in \mathbb{C}$ repeated in all elements is denoted $[\alpha]_{n\times m}$. A diagonal matrix containing elements of a vector v along its diagonal is denoted $\mathtt{Diag}(v)$. For the complex matrix X, the complex number α and a set $\Psi \subseteq \mathbb{C}^n$, we denote $X\Psi = \{Xz \mid z \in \Psi\}$ and $\alpha\Psi = \{\alpha z \mid z \in \Psi\}$. The Minkowski sum of two sets $\Gamma \subseteq \mathbb{C}^n$ and $\Psi \in \mathbb{C}^n$ is $\Gamma \oplus \Psi = \{y + z \mid y \in \Gamma, z \in \Psi\}$ .. The real projection of the set of complex vectors $\Psi \subseteq \mathbb{C}^n$ is denoted $\mathfrak{Re}(\Psi) = \{u \in \mathbb{R}^n \mid \exists v \in \mathbb{R}^n,\ u + \imath v \in \Psi\}$. We denote the set of all piecewise continuous functions from $[0, \infty)$ to Ψ as $\mathtt{Trj}(\Psi)$. An *arithmetic operation* in this paper means addition, subtraction, multiplication or division between two real numbers.

2 SELF-TRIGGERED LINEAR CONTROL SYSTEM

In a self-triggered control system, a processor computes the feedback input as well as the next sampling time instant. In this paper, we consider the problem of computing in real time an upper bound on the next sampling time period at each sampling time instant, such that the system state lies within a safe set by respecting the threshold upper bound on sampling time. Note that the self-triggered controller can however update the feedback input much before the scheduled upper bound depending on additional requirements of the system. Since this computation happens in real time, the computation complexity of evaluating the threshold time should be very low. We tackle this problem for linear control systems with possible additive disturbance inputs bounded inside a box. So, we define a self-triggered linear control system as specified by a tuple

$$\mathbb{L} = (A, B, C, K, \delta, \epsilon)$$

where $A \in \mathbb{R}^{n\times n}$ is called *state action matrix*, $B \in \mathbb{R}^{n\times m}$ is called *input action matrix*, $C \in \mathbb{R}^{n\times k}$ is called *disturbance action matrix*, $K \in \mathbb{R}^{m\times n}$ is called *feedback matrix*, $\delta : \mathbb{R}^n \times \mathbb{R}^m \to [\epsilon, \infty)$ is called sampling time threshold function and $\epsilon > 0$ is the minimum possible sampling time. A *state trajectory* of the system is a continuous function $\mathbf{x} : [0, \infty) \to \mathbb{R}^n$ such that there exist piecewise continuous functions $\mathbf{u} \in \mathtt{Trj}(\mathbb{R}^m)$ and $\mathbf{v} \in \mathtt{Trj}\left([-1, 1]^k\right)$ (see Section 1.1), called *input trajectory* and *disturbance trajectory*, respectively, and a sequence of sampling times $(t_i)_{i=0}^\infty$ satisfying the following equations for all $i \in \mathbb{Z}_{\geq 0}$ and $t \in (t_i, t_{i+1})$.

$$t_i \in \mathbb{R}_{\geq 0},\ \epsilon \leq t_{i+1} - t_i \leq \delta(\mathbf{x}(t_i), \mathbf{u}(t_i)) \tag{1a}$$

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + C\mathbf{v}(t),\ \ \mathbf{v}(t) \in [-1, 1]^k \tag{1b}$$

$$\dot{\mathbf{u}}(t) = 0, \quad \mathbf{u}(t_i) = K\mathbf{x}(t_i), \tag{1c}$$

Note that in (1c), there is a discontinuous change in the feedback input trajectory $\mathbf{u}$ at time t_i. So, the above system is a hybrid dynamical system, and more specifically a linear impulsive system [3]. We call the joint, $\begin{bmatrix}\mathbf{x}\\ \mathbf{u}\end{bmatrix}$, as the *joint trajectory* of the system. We extend the notation for piecewise continuous functions in Section 1.1 to denote the set of all possible joint trajectories of $\mathbb{L}$ as $\mathtt{Trj}(\mathbb{L})$. Then the reachable set of the system at time t originating from a set $I \subseteq \mathbb{R}^n \times \mathbb{R}^m$ is

$$\mathcal{R}(\mathbb{L}, I, t) = \left\{ \begin{bmatrix}\mathbf{x}(t)\\ \mathbf{u}(t)\end{bmatrix} \mid \begin{bmatrix}\mathbf{x}\\ \mathbf{u}\end{bmatrix} \in \mathtt{Trj}(\mathbb{L}),\ \begin{bmatrix}\mathbf{x}(0)\\ \mathbf{u}(0)\end{bmatrix} \in I \right\}.$$

The problem of safe scheduling of self-triggered controller in real time is defined as follows.

Problem 2.1 (Finding safe sampling time threshold function). *We are given matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{n\times k}$,*

$K \in \mathbb{R}^{m\times n}$, $T \in \mathbb{R}^{p\times n}$, *a positive real* $\epsilon > 0$ *and an integer. We are given a linearly constrained set* $S = \left\{z \in \mathbb{R}^{n+m} \mid \|Tz\|_\infty \leq 1\right\}$, *called* safe set. *We want to find a function* $\delta : \mathbb{R}^{n+m} \to [\epsilon, \infty)$ *that aims to maximize the value of* $\delta(z)$ *for any* $z \in \mathbb{R}^{n+m}$ *such that the following are true.*

(1) Let $\mathbb{L} = (A, B, C, K, \delta, \epsilon)$. *Then* $\forall t \in [0, \infty)$ $\mathcal{R}(\mathbb{L}, I, t) \subseteq S$.
(2) The number of arithmetic operations involved in computing $\delta(z)$ *for any* $z \in \mathbb{R}^{n+m}$ *is less that* N.

The first condition above means that the state of the system remains within a given safe set at all times. The second condition means that the schedule for the next feedback update is computed sufficiently fast in online execution. Here, the speed is measured in terms of the number of arithmetic operations, assuming that the arithmetic operations contribute majorly to the time complexity of evaluating $\delta(z)$. This real time complexity bound can ensure that very little time is spent in finding a schedule for the next feedback update, so that more time can be allocated to other incumbent tasks.

Example 2.1 (Depth control of underwater vehicle). This example is adapted from [29]. The depth of an underwater vehicle is controlled by a feedback input. The state of the underwater vehicle at any time is represented by a vector $\mathbf{x}(t) = \left[\theta(t), w(t), q(t), y(t)\right]$, where $\theta(t)$ is the pitch angle of the vehicle, $w(t)$ is the heave velocity of the vehicle, $q(t)$ is the rate of change of pitch angle, and $y(t)$ is the depth of the vehicle. The matrices of the system specified as $\mathbb{L} = (A, B, C, K, \epsilon)$ are given below. All units are S.I.

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0.0175 & -1.273 & -3.559 & 0 \\ -0.052 & 1.273 & -2.661 & 0 \\ -5 & 1 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0.085 \\ 21.79 \\ 0 \end{bmatrix},$$

$$C = 0.02B, \quad K = \begin{bmatrix} -0.7214 & 0.0445 & -0.1873 & 0.2292 \end{bmatrix}$$

The dimension of combined state and feedback input is $4 + 1 = 5$. The minimum sampling time period is $\epsilon = 0.001$ seconds. The initial set is the origin. The safe set is $S = \left\{z \in \mathbb{R}^5 \mid 0.5\,|z_4| \leq 1\right\}$, i.e., the deviation in depth of the vehicle from a reference depth should be less than 2m. We want to find a scheduling function $\delta : \mathbb{R}^5 \to [\epsilon, \infty)$ such that $\forall t \in [0, \infty)$, $\mathcal{R}(\mathbb{L}, 0, t) \subseteq S$. Also, for any $z \in \mathbb{R}^5$, the number of arithmetic operations involved in evaluating $\delta(z)$ should be less than a user-defined limit $N = 666$. Here, N can be chosen based on the time constraint on evaluating δ online and the relation between the worst case execution time for δ and the upper bound on arithmetic complexity of δ. For instance, let us consider that the time constraint on evaluating δ is 10^{-3} seconds. We have a very low power embedded processor where the worst case execution time of each arithmetic operation is $1.49 * 10^{-6}$ seconds. Then if the number of arithmetic operations in evaluation δ is less than $N = 666$, we get that the time of evaluating δ is less than $666 * 1.49 * 10^{-6} + e < 10^{-3}$ seconds, when a negligible time $e < 10^{-6}$ is spent on other operations apart from arithmetic operations.

3 A NEW SET-BASED CONDITION FOR SAFE SCHEDULING

In this section, we derive a low complexity safe scheduling function solving Problem 2.1 based on the existence of a convex set and other mathematical quantities that express the effect the system dynamics on the convex set between two consecutive updates. . The condition is quite general in the sense that it does not assume a fixed set representation (such as, zonotopes, polytopes, ellipsoids) for the convex set and can in principle be solved using any convex set representation. Such a set and the related mathematical quantities can be computed offline and their computation does not contribute to the real-time computational complexity of evaluating δ. The derivation is as follows.

Let us consider a system $\mathbb{L} = (A, B, C, K, \delta, \epsilon)$ for which we define the following mathematical quantities.

$$A_c = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}, \quad A_r = \begin{bmatrix} \mathcal{I}_n & 0 \\ K & 0 \end{bmatrix} \tag{2a}$$

$$\Upsilon(t) = \bigcup_{\mathtt{v} \in \mathtt{Trj}([-1,1]^k)} \int_0^t \exp\left(A_c\,(t-\tau)\right) \begin{bmatrix} C \\ 0 \end{bmatrix} \mathbf{v}(\tau) d\tau \tag{2b}$$

We can get closed form expressions for the joint state of the system reached within the next sampling time period as a function of the previous state for a given disturbance input trajectory. We can also derive approximations of the reachable set expressed as a function of $\Upsilon(t)$ when the disturbance input trajectory is unknown but bounded as specified in (1b). These relations are given in the following lemma.

Lemma 3.1. *Let* $\mathbf{z}$ *be a joint trajectory of* $\mathbb{L}$ *where* $(t_i)_{i=0}^{\infty}$ *is the sequence of sampling times and* $\mathbf{v} : [0, \infty) \to [-1, 1]^k$ *is the associated disturbance trajectory. Then* $\forall i \in \mathbb{Z}_{\geq 0}$, $\forall t \in (t_i, t_{i+1})$, *we get,*

$$\begin{aligned} \mathbf{z}(t) &= \exp\left(A_c\,(t-t_i)\right)\mathbf{z}(t_i) + \int_{t_i}^{t} \exp\left(A_c\,(\tau - t_i)\right) \begin{bmatrix} C \\ 0 \end{bmatrix} \mathbf{v}(\tau) d\tau \\ &\subseteq \exp\left(A_c\,(t-t_i)\right)\mathbf{z}(t_i) \oplus \Upsilon(t - t_{i+1}). \end{aligned} \tag{3a}$$

$$\begin{aligned} \mathbf{z}(t_{i+1}) &= A_r \exp\left(A_c\,(t_{i+1} - t_i)\right)\mathbf{z}(t_i) + \\ &\qquad A_r \int_{t_i}^{t_{i+1}} \exp\left(A_c\,(\tau - t_i)\right) \begin{bmatrix} C \\ 0 \end{bmatrix} \mathbf{v}(\tau) d\tau \\ &\subseteq A_r \exp\left(A_c\,(t_{i+1} - t_i)\right)\mathbf{z}(t_i) \oplus A_r \Upsilon(t_{i+1} - t_{i+1}). \end{aligned} \tag{3b}$$

Proof. For all times $t \in (t_i, t_{i+1})$, the joint trajectory $\mathbf{z}$ evolves according to the differential equation

$$\dot{\mathbf{z}}(t) = A_c \mathbf{z}(t). \tag{4}$$

As the state of the system is a continuous function at all times, while the feedback input trajectory is a continuous function between any two sampling times because there is no reset, we get that at any $t \in (t_i, t_{i+1})$, $\mathbf{x}(t)$ is the solution to the above differential equation (4), and this solution is given in (3a). Next, at time t_{i+1}, the joint state gets reset as follows $\mathbf{z}(t_{i+1}) = A_r \lim_{\tau \to t_{i+1}} \mathbf{z}(\tau) =$

$$A_r \exp\left(A_c\,(t_{i+1} - t_i)\right)\mathbf{z}(t_i) + A_r \int_{t_i}^{t_{i+1}} \exp\left(A_c\,(\tau - t_i)\right) \begin{bmatrix} C \\ 0 \end{bmatrix} \mathbf{v}(\tau) d\tau \quad \square$$

The following lemma allows us to represent reachable sets as uniformly scaled convex sets containing the origin. This result will be used to latter derive a low complexity scheduling function ensuring safety.

Lemma 3.2. *Let* Γ *be a convex set such that* $0 \in \Gamma$ *and there exists a function* $\chi_{\min} : \mathbb{R}^{n+m} \to \mathbb{R}$ *such that* $\forall z \in \mathbb{R}^{n+m}$ $\chi_{\min}(z) =$

$\min(a \in \mathbb{R}_{\geq 0} \mid z \subseteq a\Gamma)$. *Let us consider that there exist finite real valued sequences* $(\beta_j)_{j=0}^M$ *and* $(\eta_j)_{j=0}^M$ *satisfying* $\forall j \in \{1,\ldots,M\}$,

$$\beta_0 = 1,\ \eta_0 = 1, \beta_{j+1} \geq \beta_j,\ \eta_{j+1} \geq \eta_j \tag{5a}$$

$$\bigcup_{\tau\in[0,j\epsilon]} \exp(A_c\tau)\,\Gamma \subseteq \beta_j\Gamma, \quad \bigcup_{\tau\in[0,j\epsilon]} \Upsilon(\tau) \subseteq \eta_j\Gamma \tag{5b}$$

Let us consider a joint trajectory $\mathbf{z} \in \mathtt{Trj}(\mathbb{L})$ *for which* $(t_0)_{i=1}^{\infty}$ *is the set of feedback sampling times. Then the following are true* $\forall i \in \mathbb{Z}_{\geq 1},\ t \in [t_i, t_{i+1})$.

$$\mathbf{z}(t_{i+1}) \subseteq \tag{6a}$$

$$A_r \exp(A_c\epsilon)\left(\beta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\right)\Gamma \oplus A_r\Upsilon(\epsilon).$$

$$\mathbf{z}(t) \subseteq \left(\beta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\right)\Gamma. \tag{6b}$$

Proof. As $t_{i+1} - t_i \geq \epsilon$ as assumed in (1a), we derive the following using (3b) to prove (6a).

$$\mathbf{z}(t_{i+1}) \subseteq A_r \exp(A_c\epsilon)\,\mathbf{z}(t_{i+1}-\epsilon) \oplus A_r\Upsilon(\epsilon)$$
$$\subseteq \exp(A_c\epsilon)\left(\exp(A_c(t_{i+1}-t_i-\epsilon))\,\mathbf{z}(t_i) \oplus \Upsilon(t_{i+1}-\epsilon)\right) \oplus \Upsilon(\epsilon)$$
% by (5b) and because Γ is convex set containing 0
$$\subseteq A_r \exp(A_c\epsilon)\left(\beta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\right)\Gamma \oplus A_r\Upsilon(\epsilon)$$

We next derive the following using (3a) to prove (6b).

$$\mathbf{z}(t) \subseteq \exp(A_c(t-t_i))\,\mathbf{z}(t_i)\,\Gamma \oplus \Upsilon(t-t_i)$$
% by (5b) and because Γ is convex set containing 0
$$\subseteq \left(\beta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\right)\Gamma. \qquad \square$$

The following lemma is a main result which gives a sufficient condition on the scheduling function δ that solves Problem 2.1.

Lemma 3.3. *Let us consider a set* $S \subseteq \mathbb{R}^{n+m}$, *called* safe set. *Let us consider that there exist a convex set* $\Gamma \subseteq \mathbb{R}^n$, , *called* sampling time invariant, *a function* $\chi : \mathbb{R}^{n+m} \to \mathbb{R}$, *called* inclusion scaling function, *and finite increasing sequences of positive real numbers* $(\beta_j)_{j=0}^M$ *and* $(\eta_j)_{j=0}^M$ *such that (5a,5b) and the following are true.*

$$0 \in \Gamma,\ \forall z \in \mathbb{Z}_{\geq 0},\ z \subseteq \chi(z)\,\Gamma \tag{7a}$$

$$A_r \exp(A_c\epsilon)\,\Gamma \oplus A_r\Upsilon(\epsilon) \subseteq \Gamma, \tag{7b}$$

$$I \subseteq \Gamma,\ (\beta_1 + \eta_1)\,\Gamma \subseteq S \tag{7c}$$

Let us consider that the number of arithmetic operations involved in evaluating $\chi(z)$ *for all* $z \in \mathbb{R}^{n+m}$ *is less than* N_χ. *Let us define a scheduling function* $\delta : \mathbb{R}^{n+m} \to \mathbb{R}_{\geq 0}$:

$$Inds(z) = \left\{j \in \{1,\ldots,M\} \mid \chi(z)\beta_j + \eta_j \leq 1\right\}, \tag{7d}$$

$$\text{if } Inds(z) \neq \emptyset, \text{ then } \delta(z) = \max Inds(z)\,\epsilon \tag{7e}$$

$$\text{otherwise } \delta(z) = \epsilon \tag{7f}$$

Then the following are true.

⟨1⟩ $\mathcal{R}(\mathbb{L}, I, [0,\infty)) \subseteq S$.

⟨2⟩ *Number of arithmetic operations for computing the value of* $\delta(z)$ *for all* $z \in \mathbb{R}^{n+m}$ *is not greater than* N_χ+2 $\left(\log_2(M)+1\right)$.

Proof. Proof of ⟨1⟩: We prove this by induction. Let us consider a joint trajectory $\mathbf{z} \in \mathtt{Trj}(\mathbb{L})$ where $(t_i)_{i=0}^{\infty}$ is the sequence of feedback sampling times. As we assumed that a function χ exists where $\forall z \in \mathbb{R}^{n+m}\ z \subseteq \chi(z)\,\Gamma$, so the function $\chi_{\min}$ exists where $\forall z \in \mathbb{R}^{n+m}\ \chi_{\min}(z) = \min(a \in \mathbb{R}_{\geq 0} \mid az \subseteq \Gamma)$. Moreover, we have $\forall z \in \mathbb{R}^{n+m}\ \chi_{\min}(z) \leq \chi(z)$.

Let us consider that for some $i \in \mathbb{Z}_{\geq 0}$ we have $\mathbf{z}(t_i) \in \Gamma$. Then we shall prove that

$$(\mathbf{z}(t_{i+1})) \in S,\ \ \forall t \in [t_i, t_{i+1})\ \mathbf{z}(t) \in \Gamma. \tag{8a}$$

Let us consider two cases, (i) $Inds(\mathbf{z}(t_i)) \neq \emptyset$, (ii) $Inds(\mathbf{z}(t_i)) = \emptyset$.

In the first case, by (7e) and as $\chi_{\min} \leq \chi$, we get

$$\beta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}$$
$$\leq \beta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\chi(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil} \leq 1, \ \text{ % by (7e),} \tag{8b}$$

$$\beta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil} \leq \beta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\chi(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}$$
% by (7d,7e, 5a)
$$\leq 1 = \beta_0 + \eta_0 \leq \beta_1 + \eta_1. \tag{8c}$$

In the second case, we have $t_{i+1} - t_i = t - t_i = \epsilon$. So we have, $\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil = 0$ and $\left\lceil\frac{t-t_i}{\epsilon}\right\rceil = \left\lceil\frac{\epsilon}{\epsilon}\right\rceil = 1$. So, we get

$$\beta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t_{i+1}-t_i-\epsilon}{\epsilon}\right\rceil} = \beta_0\chi_{\min}(\mathbf{z}(t_i)) + \eta_0$$
% as $\mathbf{z}(t_i) \in \Gamma$, so $\chi_{\min}(\mathbf{z}(t_i)) \leq 1$
$$= 1 \cdot \chi_{min}(\mathbf{z}(t_i)) + 0 = \chi_{min}(\mathbf{z}(t_i)) \leq 1, \tag{8d}$$

$$\beta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil}\chi_{\min}(\mathbf{z}(t_i)) + \eta_{\left\lceil\frac{t-t_i}{\epsilon}\right\rceil} = \beta_1\chi_{\min} + \eta_1$$
% as $\mathbf{z}(t_i) \in \Gamma$, so $\chi_{\min}(\mathbf{z}(t_i)) \leq 1$
$$\leq \beta_1 + \eta_1 \ \text{ % as } \mathbf{z}(t_i) \in \Gamma \tag{8e}$$

Then by (6a, 8b,8d), we get,

$$\mathbf{z}(t_{i+1}) \subseteq A_r \exp(A_c\epsilon)\,\Gamma \oplus \Upsilon(\epsilon) \subseteq \Gamma. \ \text{ % by (7b)} \tag{8f}$$

Similarly, by (6b, 8c,8e), we get $\mathbf{z}(t) \subseteq \Gamma$. So, we have proved (8a).

Then by induction we get that if $I \subseteq \Gamma$, then ⟨1⟩ is true. But $I \subseteq \Gamma$ by (7c). So, we get ⟨1⟩.

Proof of ⟨2⟩: The arithmetic operations involved in evaluating $\delta(z)$ consist of computing the value of the function $\chi(z)$ and then computing the maximum of $Inds(z)$. The number of arithmetic operations involved in computing $\chi(z)$ is given to be upper bounded by N_χ. On the other hand, we evaluate the maximum value of $Inds(z)$ using binary search and this complexity in logarithmic in M. We thus get the upper bound given in the above lemma for the arithmetic complexity of evaluating $\delta(z)$. $\square$

4 OFFLINE COMPUTATIONS

We can improve the accuracy of online reachability analysis by precomputing a large sequence of reachable sets (5b) and using this sequence to approximate reachable sets online. Then the key idea in reducing complexity of online scheduling while utilizing precomputed reachable sets is that we represent the reachable

set as scalar multiples of a common convex set and used binary search on scalar variables to perform scheduling. In this regard, we describe in this section the procedure to compute offline the convex set Γ and scalar sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=1}^{M}$ satisfying Equations (7b,7c, 5a, 5b) which are preconditions in Lemma 3.3. The time of these offline computations does not add to the online computation time for evaluation the scheduling function δ. To solve for Γ, we need to parameterize the encoding of the set Γ and solve for the parameters that satisfy (7b,7c, 5a, 5b). In this paper, we choose the real projection of the complex zonotope [2] for encoding Γ because the class of complex zonotopes are guaranteed to contain solutions sets to fixed point equations like (7b) for stable systems with full rank eigenstructure (see Theorem 7 [2]). Also, they are closed under linear transformations and Minkowski sums, which benefits in improving accuracy of approximating reachable sets of linear systems under additive disturbance.

A complex zonotope is an encoding of a set of complex valued points where each point is a linear combination of complex valued vectors such that the combining coefficients are complex valued and bounded. The real projection of a complex zonotope can represent some non-polytopic sets in addition to usual polytopic zonotopes, and hence are more expressive than usual (real valued) zonotopes.

Definition 4.1 ([2]). Let $G \in \mathbb{C}^{rows(G)\times cols(G)}$ be a complex matrix, called generator matrix, $c \in \mathbb{R}^{rows(G)}$ be a real vector, called center and $s \in \mathbb{R}_{\geq 0}^{cols(G)}$ be a non-negative vector, called scale vector. The tuple (G, c, s) is a complex zonotope which represents the following set of points.

$$\mathcal{Z}(G, c, s) = \left\{ G\zeta + c \mid \zeta \in \mathbb{C}^{cols(G)}, |\zeta| \leq s \right\} \tag{9}$$

We shall revisit a relation between complex zonotopes, which can be used as a sufficient condition for checking the inclusion between complex zonotopes. This relation and other properties will be used later in this paper to to solve for Γ, $(\beta_j)_{j=0}^{M}$, $(\eta_j)_{j=1}^{M}$ in (7b, 7c, 5a,5b).

Definition 4.2 ([1]). Let us consider two complex zonotope encodings $(G, c, s) \subseteq \mathbb{C}^{n+m}$ and $(H, e, r) \subseteq \mathbb{C}^{n+m}$. Let us define a relation "$\sqsubseteq$" between the two complex zonotopes as $(H, e, r) \sqsubseteq (G, c, s)$ if the following is true.

$$\begin{aligned} &\exists X \in \mathbb{C}^{cols(G)\times cols(H)}, \ \exists y \in \mathbb{C}^{cols(G)} : \\ &GX = H\,\mathtt{Diag}(r), \quad Gy = r - c \\ &\forall j \in \{1, \ldots, cols(G)\} \quad |y_i| + \sum_{j=1}^{cols(H)} \left|X_{ij}\right| \leq s_i \end{aligned} \tag{10a}$$

Then the following implication holds between any two complex zonotopes (G, c, s) and (H, e, r).

$$(G, c, s) \sqsubseteq (H, e, s) \implies \mathcal{Z}(G, c, s) \subseteq \mathcal{Z}(H, e, s) \tag{11}$$

The linear transformation of a complex zonotope followed by Minkowski sum with another complex zonotope is a complex zonotope, computed as follows:

$$P\,\mathcal{Z}(G, c, s) \oplus \mathcal{Z}(H, e, r) = \mathcal{Z}\left(\begin{bmatrix} PG & H \end{bmatrix}, Pc + e, \begin{bmatrix} s \\ r \end{bmatrix}\right). \tag{12}$$

Algorithm 1 Offline computation of a set Γ and sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=1}^{M}$ satisfying (7b,7c, 5a, 5b)

Choose $l \in \mathbb{Z}_{\geq 0}$. % order of complex zonotope
Choose $M \in \mathbb{Z}_{\geq 0}$. % length of reachable set sequences
$G \leftarrow \mathcal{I}_{n+m}$.
for i in $2 : l$ **do**
  $G \leftarrow \begin{bmatrix} G & \text{Eigenvectors}\left(A_r \exp\left(A_c i\epsilon\right)\right) \end{bmatrix}$
end for
$s \leftarrow \arg_s \max \ [1]_{1\times l} |G|\, s$ (G is constant
  (15a-15c) are satisfied. % convex optimization (13a)
$\rho \leftarrow \min \rho' \in [0, \infty) : \ (\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \sqsubseteq (\rho' G, 0, s)$
$\beta_0 \leftarrow \rho + \min \beta \in [1, \infty) : (\exp(A_c j\epsilon)\, G, 0, s) \sqsubseteq (\beta G, 0, s)$
$\eta_0 \leftarrow \min \eta \in [0, \infty) : \ \left(J(0), 0, [1]_{q\times 1}\right) \sqsubseteq (\eta G, 0, s)$
for $j = 1 : M$ **do**
  $\beta_j \leftarrow \max\left\{\beta_{j-1}, \rho + \min \beta \in [0, \infty) : (18e)\right\}$
  (18e): $(\exp(A_c j\epsilon)\, G, 0, s) \sqsubseteq (\beta G, 0, s)$
  $\eta_j \leftarrow \max\left\{\eta_{j-1}, \min \eta \in [0, \infty) : (18g)\right\}$
  (18g): $\left(J(j), 0, [1]_{q\times 1}\right) \sqsubseteq (\eta G, 0, s)$
end for
return $\Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s))$, $(\beta_j)_{j=0}^{M}$, $(\eta_j)_{j=1}^{M}$.

The interested reader may go through [2] for proofs of the above properties of complex zonotopes. Now we describe the offline computations to find a complex zonotopic set centered at the origin $\Gamma = \mathcal{Z}(G, 0, s)$ and sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=1}^{M}$ satisfying Equations (7b,7c, 5a, 5b), the preconditions in Lemma 3.3.

In a time interval $[j\epsilon, (j+1)\epsilon]$ where $j \in \{0, \ldots, M\}$, the contribution of the disturbance set to the reachable set approximation in that time interval is $\bigcup_{t\in[j\epsilon,(j+1)\epsilon]} \Upsilon(t)$ according to (3a). For small value of ϵ, this set can be approximated by a zonotope with very good accuracy using the algorithm of Girard [16]. We subsume this algorithm in this paper, and denote the resulting zonotopic over-approximation as $\mathcal{Z}\left(J(j), 0, [1]_{q\times 1}\right)$ where q is the maximum number of columns in the generator matrices, i.e.,

$$\bigcup_{t\in[j\epsilon,(j+1)\epsilon]} \Upsilon(t) \subseteq \mathcal{Z}\left(J(j), 0, [1]_{q\times 1}\right) \tag{14}$$

Then the following lemma will be used to compute a set Γ satisfying (7b,7c).

Lemma 4.3. *Let $G \in \mathbb{C}^{(n+m)\times l}$ be a complex matrix and $T \in \mathbb{R}^{p\times n}$ be a real matrix. Let us consider sets $S = \left\{z \in \mathbb{R}^{n+m} \mid \|Tz\|_\infty \leq 1\right\}$ and $I = [-r, r]$ for some $r \in \mathbb{R}^n$. Let $s \in \mathbb{R}_{\geq 0}$ be a solution vector to the following convex constraints.*

$$|TG|\, s + |TA_c G|\, s\epsilon \leq \left[1\right]_{p\times 1} \tag{15a}$$

$$\mathcal{Z}\left(\begin{bmatrix} A_r \exp(A_c\epsilon)\, G & J(0) \end{bmatrix}, 0, \begin{bmatrix} s \\ [1]_{q\times 1} \end{bmatrix}\right) \sqsubseteq \mathcal{Z}(G, 0, s) \tag{15b}$$

$$\mathcal{Z}(\mathcal{I}_{n+m}, 0, r) \sqsubseteq \mathcal{Z}(G, 0, s) \tag{15c}$$

Then $\Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s))$ (real projection) satisfies Equations (7b, 7c).

Proof. By (12) and (14), we get that the L.H.S. of (7b) is equal to the real projection of L.H.S of (15b). Next by (11), we get (7b) and $I \subseteq \Gamma$. Next, by Taylor remainder theorem, we get that for any $\tau \in [0, \epsilon]$ and $z \in \mathbb{R}^{n+m}$,

$$T \exp(A_c \tau)\, z \subseteq Tz \oplus [-|TA_c z|\, \epsilon, |TA_c z|\, \epsilon] \tag{16}$$

So, for any $z = \mathfrak{Re}(G\zeta) \in \Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s)) : |\zeta| \leq s$, we get $T \exp(A_c\tau)\, z \leq Tz + |TA_c z|\, \epsilon \leq \mathfrak{Re}(TG\zeta) + |TA_c G\zeta|\, \epsilon \leq |TG|\, s + |TA_c G|\, s\epsilon \leq [1]_{p\times 1}$. This proves (7c). □

Solving for set Γ optimally satisfying Equations (7b, 7c): We first fix the generator matrix G of the complex zonotope. For this, we first choose the eigenvectors of $A_r \exp(A_c i\epsilon)$ for a few different values of $j \in \mathbb{Z}_{\geq 0}$ to be among the columns of G, including $j = 0$. This is because according to Theorem 7 of [2], the existence of s satisfying the conditions (7b,7c) are guaranteed under certain stability and rank conditions, when G contains the eigenvectors of $A_r \exp(A_c\epsilon)$ among its columns. Then we can augment any number of additional generators to G because the scale factors can be adjusted to regulate the effect of each generator on the satisfaction of the above constraints. We therefore augment the identity matrix to generators of G. Now for the fixed G, the constraints above are convex in s and other auxiliary variables involved in the relations. So, we can use convex optimization to solve the constraints for the fixed G. While solving the convex constraints, we want to maximize the size of Γ, so that we can find a large bound on the sampling time period. So, we perform the following convex optimization to find a large sized real projection of complex zonotope $\Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s))$ satisfying (7b, 7c). The sum of projections of the complex zonotope along different axis is given by $[1]_{1\times l}\, |G|\, s$ based on Lemma 12 of [2]. So, we maximize this sum to increase the size of the complex zonotope.

$$max \quad [1]_{1\times l}\, |G|\, s : \text{ (15a-15c) are satisfied.} \tag{17}$$

Next, we shall compute the non-decreasing sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=1}^{M}$ satisfying (7b,7c, 5a, 5b) inductively using convex optimizations according to the following lemma.

Lemma 4.4. *Let us consider a set $\Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s))$. Let us consider a positive real $\rho \in \mathbb{R}_{\geq 0}$ and finite sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=0}^{M}$ defined inductively as optimal solutions to convex optimization problems (18a-18f) below.*

$$\rho = \min \rho' \in [0, \infty) : (\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \sqsubseteq (\rho' G, 0, s) \tag{18a}$$

$$\beta_0 = \rho + \min \beta \in [1, \infty) : (\exp(A_c j\epsilon)\, G, 0, s) \sqsubseteq (\beta G, 0, s) \tag{18b}$$

$$\eta_0 = \min \eta \in [0, \infty) : \left(J(0), 0, [1]_{q\times 1}\right) \sqsubseteq (\eta G, 0, s) \tag{18c}$$

$$\forall j \in \mathbb{Z}_{\geq 1}\ \beta_j = \max\left\{\beta_{j-1}, \rho + \min \beta \in [0, \infty) : (18e)\right\}, \tag{18d}$$

$$(\exp(A_c j\epsilon)\, G, 0, s) \sqsubseteq (\beta G, 0, s) \tag{18e}$$

$$\forall j \in \mathbb{Z}_{\geq 1}\ \eta_j = \max\left\{\eta_{j-1}, \min \eta \in [0, \infty) : (18g)\right\} \tag{18f}$$

$$\left(J(j), 0, [1]_{q\times 1}\right) \sqsubseteq (\eta G, 0, s) \tag{18g}$$

Then the finite sequences $(\beta_j)_{j=0}^{M}$ and $(\eta_j)_{j=0}^{M}$ satisfy (5a,5b).

Proof. By (18b,18c,18d,18f), we have $\forall j \in \mathbb{Z}_{\geq 0}$, $\beta_{j+1} \geq \beta_j$ and $\eta_{j+1} \geq \eta_j$, which proves (5a).

Next, let us consider a $t \in [j\epsilon, (j+1)\epsilon]$ for some $j \in \mathbb{Z}_{\geq 0}$. By Taylor remainder theorem, for any $z \in \mathbb{R}^{n+m}$, we can write

$$\exp(A_c t)\, z \subseteq \exp(A_c j\epsilon) \oplus [-|A_c z|\, \epsilon, |A_c z|\, \epsilon] \tag{19a}$$

$$= \exp(A_c j\epsilon)\, \mathcal{Z}(G, 0, s) \oplus \mathcal{Z}(\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \tag{19b}$$

$$= \mathcal{Z}(\exp(A_c j\epsilon), 0, s) \oplus \mathcal{Z}(\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \ \%\text{ by (12)} \tag{19c}$$

By (18a), we get that

$$\mathcal{Z}(\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \subseteq \rho\, \mathcal{Z}(G, 0, s) \tag{19d}$$

So, by (12,18b,18d,19d), and that $\mathcal{Z}(G, 0, s)$ is a convex set containing 0, we get that $\forall j \in \mathbb{Z}_{\geq 0}$,

$$\mathcal{Z}(\mathcal{I}_{n+m}, 0, |A_c G|\, s\epsilon) \oplus \exp(A_c j\epsilon)\, \mathcal{Z}(G, 0, s) \subseteq \beta_j\, \mathcal{Z}(G, 0, s)$$

% By (19c)

$$\implies \forall t \in [j\epsilon, (j+1)\epsilon]\ \exp(A_c t)\, \Gamma \subseteq \beta_j \Gamma \tag{19e}$$

The above proves that (5b) is satisfied. Next by (18c,18f), we get that $\forall j \in \mathbb{Z}_{\geq 0}, \forall t \in [j\epsilon, (j+1)\epsilon]$, $\Upsilon(t) \subseteq \mathfrak{Re}(\mathcal{Z}(G, 0, s) = \Gamma)$. This proves (5b). □

5 DERIVING ONLINE SCHEDULING FUNCTION

After computing the complex zonotopic projection set Γ according to Equation (17) and Lemma 4.4, we have to first derive the inclusion scaling function χ given in Lemma 3.3. The function checks the amount of scaling required to include a given point inside the set Γ. This function will be evaluated online, thus we want the computational complexity of evaluating χ to be very small. Such a function is given in the following lemma, and the complexity of evaluating the function is also given.

Lemma 5.1. *Let $\mathcal{Z}(G, 0, s) \subseteq \mathbb{C}^{n+m}$ be a complex zonotope such that $(G\,\mathtt{Diag}(s))$ has rank $n+m$ and $(G\,\mathtt{Diag}(s))^{\dagger}$ denote the Moore-Penrose pseudo-inverse of $(G\,\mathtt{Diag}(s))$. Let us define a function,*

$$\chi : \mathbb{R}^{n+m} \to \mathbb{R}_{\geq 0},\ \forall z \in \mathbb{R}^{n+m},\ \chi(z) = \left\|(G\,\mathtt{Diag}(s))^{\dagger} z\right\|_{\infty}. \tag{20}$$

Then $z \subseteq \chi(z)\, \mathcal{Z}(G, 0, s)$.

Since G, s are precomputed offline, so the matrix $(G\,\mathtt{Diag}(s))^{\dagger}$ can be precomputed offline before evaluating $\delta(z)$ online. . Then the number of arithmetic operations required for computing $\chi(z)$ for any $z \in \mathbb{R}^{n+m}$ is not greater than $(8(n+m)+3)$ cols (G).

Proof. Let us consider $z \subseteq \mathbb{R}^{n+m}$ and $\zeta = \frac{\mathtt{Diag}(s)(G\,\mathtt{Diag}(s))^{\dagger} z}{\chi(z)}$. As $\chi(z) = \left\|(G\,\mathtt{Diag}(s))^{\dagger} z\right\|_{\infty}$, we get $|\zeta| = \frac{\left|\mathtt{Diag}(s)(G\,\mathtt{Diag}(s))^{\dagger}\right| z}{\chi(z)}$

$$\leq |s|\, \frac{\left\|(G\,\mathtt{Diag}(s))^{\dagger} z\right\|_{\infty}}{\chi(z)} = |s|\, \frac{\chi(z)}{\chi(z)} = |s| \tag{21a}$$

Then we can write

$$z = (G\,\mathtt{Diag}(s))\,(G\,\mathtt{Diag}(s))^{\dagger} z = G\chi(z)\,\zeta \tag{21b}$$

$$\subseteq \chi(z)\, \mathcal{Z}(Z, 0, s) \quad \%\text{ (as } |\zeta| \leq s \text{ by (21a))} \tag{21c}$$

Next, the upper bound on the arithmetic complexity of computing $\chi(z)$ is an upper bound on the arithmetic complexity of computing the complex valued matrix multiplication plus complexity of computing the infinity norm in (20). The number of complex

number multiplications involved in the complex valued matrix multiplication is less than $(n+m)\ cols\,(G)$. So, the number of real valued multiplications is less than 4 times the former, which is $4(n+m)\ cols\,(G)$. During a single complex valued scalar multiplication, there are 2 real valued additions involved. So, the number of real valued additions resulting from complex valued multiplications is $2(n+m)\ cols\,(G)$. Also, during complex valued matrix multiplication, there are less than $(n+m)\ cols\,(G)$ complex valued additions, which amounts to another $2(n+m)\ cols\,(G)$ real valued additions. So, the total number of arithmetic operations resulting from the complex valued matrix multiplication is less than $(4+2+2)(n+m)\ cols\,(G) = 8(n+m)\ cols\,(G)$. Next, we have to compute the maximum of the absolute values of the complex vector after the matrix multiplication. For each absolute value calculation, there will be two real valued multiplications and one addition. So, the total number of arithmetic operations is bounded by $(8(n+m))\ cols\,(G) + (2+1)\ cols\,(G) = (8(n+m)+3)\ cols\,(G)$. □

Now we derive an online scheduling function δ that solves Problem 2.1. The derivation makes use of the function χ derived in Lemma 5.1 and the sequences $(\beta_i)_{i=0}^{M}, (\eta_i)_{i=0}^{M}$ computed according to Lemma 4.4. This is given in the following theorem.

Theorem 5.2. *Let us consider an integer $N > 0$. Let us choose hyperparameters $l \in \mathbb{Z}_{\geq 1}$ (order of complex zonotope) and M (length of reachability sequences) in Algorithm 1 such that*

$$(8(n+m)+3)\, l(n+m) + 2\left(\log_2(M)+1\right) \leq N \tag{22}$$

Let us consider the complex zonotope tuple $(G, 0, s)$ and the sequences $(\beta_i)_{i=0}^{M}, (\eta_i)_{i=0}^{M}$ computed by Algorithm 1, and the inclusion scaling function $\chi : \mathbb{R}^{n+m} \to \mathbb{R}_{\geq 0}$ defined in Equation (20). Let us define a scheduling function $\delta : \mathbb{R}^{n+m} \to \mathbb{R}_{\geq 0}$ according to Equation (7d-7f) of Lemma 3.3 for the above sequences $(\beta_i)_{i=0}^{M}, (\eta_i)_{i=0}^{M}$. Then we have $\mathcal{R}(I, [0,\infty)) \subseteq S$ and the number of arithmetic operations required for evaluating $\delta(z)$ for any $z \in \mathbb{R}^{n+m}$ is not greater than N.

Proof. According to Lemmas 4.3 and 4.4, we get that the set $\Gamma = \mathfrak{Re}(\mathcal{Z}(G, 0, s))$ and sequences $(\beta_i)_{i=0}^{M}, (\eta_i)_{i=0}^{M}$ satisfy the Equations (7b, 7c, 5a,5b) of Lemma 3.3. Therefore, by Lemma 3.3, we have $\mathcal{R}(I, [0,\infty)) \subseteq S$.

Next, by Lemma 3.3⟨2⟩, we get that the arithmetic complexity of evaluating δ is bounded above by $N_\chi + 2\left(\log_2(M)+1\right)$ where N_χ is the arithmetic complexity of evaluating χ. But by Lemma 5.1, we have $N_\chi = (8(n+m)+3)\ cols\,(G)$, and by Algorithm 1, we have $cols\,(G) = l(n+m)$. Thus the complexity of evaluating δ is bounded above by $(8(n+m)+3)\, l(n+m) + 2\left(\log_2(M)+1\right)$. Then by (22), this complexity is bounded above by N. □

Reducing complexity while improving accuracy. According to the above theorem, the online computational arithmetic complexity is bounded by $(8(n+m)+3)\, l(n+m) + 2\left(\log_2(M)+1\right)$. This online complexity is negligible compared to an accurate reachability analysis algorithm like [16]. For example, in a step by step reachability analysis of Girard, we have to perform at least $\delta(z)/\epsilon$ matrix multiplications of an $n \times n$ matrix with $n \times cols\,(G)$ generator matrix G, assuming a time step size ϵ. The latter requires at least $(n+m)\ cols\,(G)$ multiplications by school book style matrix multiplication. Since $\delta(z)/\epsilon$ is upper bounded by M, so the worst case complexity of [16] is at least M times the worst case complexity of our algorithm. So, we can reduce the complexity of scheduling function by M times compared to performing online accurate reachability analysis. But will still improve the accuracy of online reachability analysis because of precomputing a large sequence of reachable sets.

6 EVALUATION

We evaluate our procedure on three examples: an underwater vehicle depth control model having 4-dimensional state and 1-dimensional feedback input, a quadcopter height control model having 8-diemsional state and 4-dimensional feedback, and a vehicle platoon having 9-dimensional state and 3-dimensional feedback. For these examples, we derive a scheduling function based on Theorem 5.2. Our aim is to show that the scheduling function based on precomputed reachable set sequences significantly improves the computation speed of finding a safe upper bound on sampling time as compared to performing an accurate reachability analysis online. For this purpose, we first compute the upper bounds on the sampling time period for a large number of randomly sampled points in the state space using the scheduling function. Next, we verify the schedules for all sampled points using the highly accurate reachability algorithm developed by Girard [16]. For a fair comparison, the length of time step used in the reachability analysis by [16] is the same as the length of time step used in computing the reachability sequences in (5b). The maximum order of the zonotopes resulting from additive disturbance (see (14)) is also kept the same. Then we note the ratio of the computation times of both approaches on the same computing platform, an M1 Macbook Pro laptop with 16 GB RAM. We take the value of l and M in Algorithm 1 to be 3 and 400, respectively. So, the number of generators of invariant complex zonotope is $l(n+m) = 3(n+m)$, where n and m are the state and input dimensions, respectively. The numbers are given in Table 1.

Impelementation software. The software implementation is uploaded in the Github repository
https://github.com/asarvind/SafeSelfTriggered.jl
as a Julia package. For set-based reachability analysis, we used LazySets Julia library[1] [13]. For convex optimization, we used JuMP Julia library[2] [11] with Mosek optimizer[3].

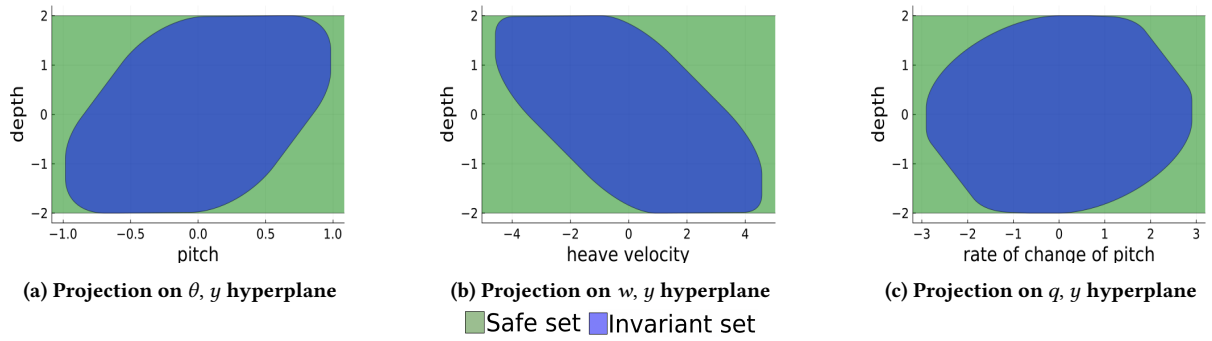
6.1 Underwater vehicle depth control

The dynamics of an underwater vehicle with the dimension of state space $n = 4$, and the dimension of feedback input $m = 1$, was described earlier in Example 2.1. The projection of the complex zonotope invariant on different hyperplanes are given in Figure 1.

6.2 Quadcopter

We consider the model of a quadcopter describing the control of height and orientation, which is adapted from [36]. The state of the system is an 8-dimensional vector $\left[y, \dot{y}, \phi, \dot{\phi}, \theta, \dot{\theta}, \phi, \dot{\psi}\right]$, where y is the deviation of the height of quadcopter from a reference value, ϕ is the pitch angle of the quadcopter, θ is the roll angle of the quadcopter, and ψ is the yaw angle, while $\dot{e} : e \in \{y, \phi, \theta, \psi\}$, represents the rate of change of the respective quantities. A 4-dimensional

[1] https://juliareach.github.io/LazySets.jl/dev/
[2] https://jump.dev/JuMP.jl/stable/
[3] https://www.mosek.com, https://github.com/MOSEK/Mosek.jl



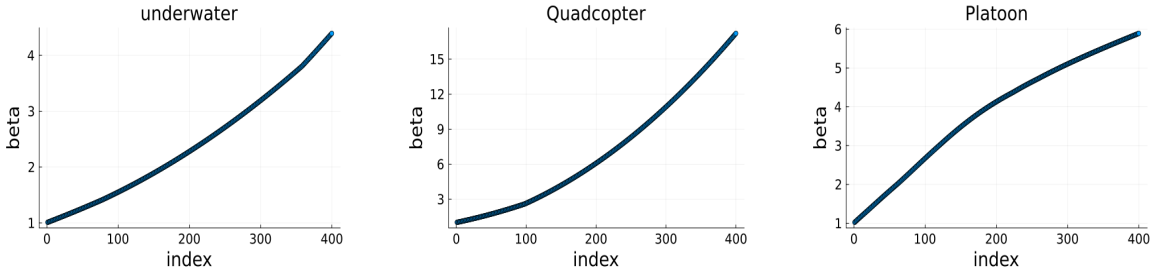
(a) Projection on θ, y hyperplane **(b) Projection on w, y hyperplane** **(c) Projection on q, y hyperplane**

Figure 1: Projection of sampling time invariant complex zonotope and safe set

$$A = \begin{bmatrix} 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1.0000 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.6050 & 4.8680 & -3.5754 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 & 0 & -1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.1936 & 3.6258 & -3.2396 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & -1.0000 \\ 0.7132 & 3.5730 & -0.0964 & 0.8472 & 3.2568 & -0.0876 & 1.2726 & 3.0720 & -3.1356 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad C = \begin{bmatrix} 0 \\ 4.5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

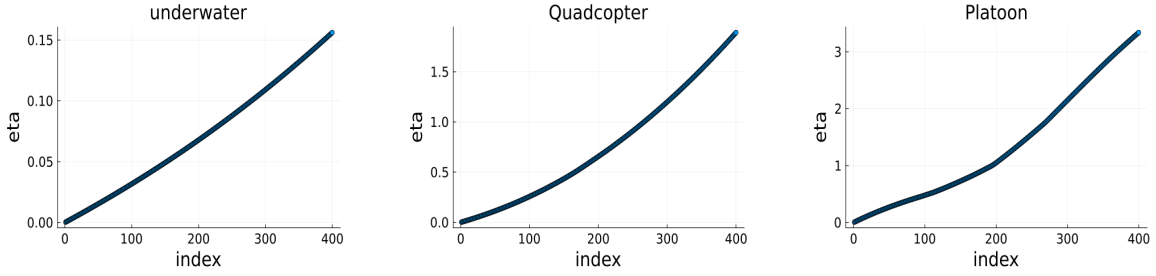
$$K = \begin{bmatrix} 0 & 0 & 0 & -0.8198 & 0.4270 & -0.0450 & -0.1942 & 0.3626v - 0.0946 & \\ 0.8718 & 3.8140 & -0.0754 & 0 & 0 & 0 & -0.5950 & 0.1294 & -0.0796 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad T = \begin{bmatrix} -1/35 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1/35 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1/40 & 0 & 0 \end{bmatrix}$$

Figure 2: Matrices of platoon model



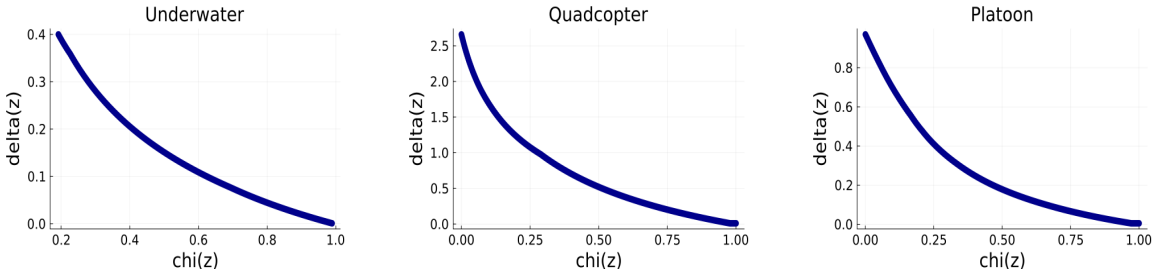
(a) Underwater vehicle **(b) Quadcopter** **(c) Platoon model**

Figure 3: Plots of β sequence w.r.t index



(a) Underwater vehicle **(b) Quadcopter** **(c) Platoon model**

Figure 4: Plots of η sequence w.r.t index



(a) Underwater vehicle **(b) Quadcopter** **(c) Platoon model**

Figure 5: Sampling period bound $\delta(z)$ as a function of scale factor $\chi(z)$

Table 1: Offline computation time and upper bound on arithmetic complexity of evaluating scheduling function (22)

Example	Sequence length M	State Dimension n	Input Dimension m	Number of generators	Offline computation time	Bound on no. of arithmetic operations for evaluating $\delta(z)$
8 Underwater	400	4	1	15	6.45	665
Quadcopter	400	8	4	36	43.86	3584
Platoon	400	9	3	36	66.61	3584

Table 2: Comparison of speed of computing a schedule by our method vs verification of the schedule by [16]

Example	Sampling Region	No. of Samples	Comp. time evaluating scheduling function on all samples (secs)	Verification time of all schedules (secs)	Speed gain
Underwater	$1 \cdot \Gamma$	100000	0.168111834	27.128271929	161
	0.5Γ	100000	0.142447833	33.272043682	233
	0.25Γ	100000	0.152343667	33.236158057	218
	0.1Γ	100000	0.158015875	33.992097849	215
Quadcopter	0.1Γ	100000	0.199598708	42.238855978	211
	0.05Γ	100000	0.211739542	47.90259098	226
	0.025Γ	100000	0.228992125	51.811915023	226
	0.01Γ	100000	0.179047042	53.478963733	298
Platoon	0.1Γ	100000	0.221021458	38.104817226	172
	0.05Γ	100000	0.267661167	39.288019518	146
	0.025Γ	100000	0.202916	37.291044559	183
	0.01Γ	100000	0.155351209	39.87190756	256

feedback input changes the angular velocity of the rotors of the quadcopter. The feedback input tries to keep the height and quadcopter state orientation within a safe set where $y \in [-1, 1]\, m$, $\phi \in [-0.1, 0.1]\, rad$, $\theta \in [-0.2, 0.2]\, rad$ and $\psi \in [-1, 1]\, rad$. So, the combined state space has $8 + 4 = 12$ dimensions. The lower bound on the sampling period is $\epsilon = 0.01$. The matrices of the system dynamics A, B, C, K are,

$$A \in \mathbb{R}^{8\times 8},\ A_{ij} = \begin{cases} 1 & (i,j) \in \{(1,2),(3,4),(7,8)\} \\ 0 & otherwise \end{cases}, \tag{23a}$$

$$B \in \mathbb{R}^{8\times 4},\ B_{ij} = \begin{cases} 1 & (i,j) \in \{(2,1),(4,2),(6,3),(8,4)\} \\ 0 & otherwise \end{cases}, \tag{23b}$$

$$C = 0.001B, \tag{23c}$$

$$K \in \mathbb{R}^{4\times 8},\ K_{ij} = \begin{cases} -0.6325 & (i,j) \in \{(1,1),(2,3),(3,5)\} \\ -1.2903 & (i,j) \in \{(1,2),(2,4),(3,6)\} \\ -1 & (i,j) = (4,7) \\ -1.7321 & (i,j) = (4,8) \end{cases}. \tag{23d}$$

6.3 Networked platoon of vehicles

We consider a model of a networked platoon of vehicles adapted from [25]. The model describes the dynamics of a platoon of four vehicles where the inter-vehicle distances, relative speeds, and accelerations of the vehicles constitute a 9-dimensional state of the system. The system has a 3-dimensional feedback input acceleration applied on each of the three follower vehicles. The acceleration of the leading vehicle is uncertain and lies in the interval $[-9, 1]\, m/s^2$.

However, in this paper, we shift the dynamics to the equilibrium point to be consistent with the analysis in this paper. Then the disturbance input lies in the range $[-5, 5]\ m/s^2$. The matrices corresponding to the system dynamics are given in Figure 2. The safe set is the set of states where the inter-vehicle distances are greater than a threshold so that collision is avoided. The corresponding matrix T and vector of offsets d specifying the half-space representation of the safe set are given in Figure 2. The lower bound ϵ on the sampling time period is 0.005s.

6.4 Results

For all the three models, we successfully find a scale factor s that solves the constrained convex optimization in (17). This ensures that $\Gamma = \mathfrak{Re}\left(\mathcal{Z}\left(G, 0, s\right)\right)$ satisfies the conditions (7b,7c) in Lemma 3.3. The inclusion of Γ inside the safe set for the underwater vehicle example is depicted in Figure 1. Next, we compute the sequences β and η satisfying the conditions (5a-5b) in Lemma 3.3. The increasing sequences are plotted with respect to their indices in Figures 3, 4. Based on the sequences and the set Γ, we can derive a scheduling function according to Theorem 5.2. The offline computation time for finding the set Γ and sequences β, η, and upper bound on arithmetic complexity of computing the scheduling function based on Theorem 22 are given in Table 1. We evaluate the scheduling function for 100000 randomly sampled points in the joint state space of state and feedback input. Then we plot the scheduling function δ versus the inclusion scaling function $\chi\left(z\right)$. These plots are given in Figure 5.

We then ran the algorithm in [16] to verify the sampling period bounds for all the 100000 samples and recorded the computation times on the same computing system. We observed a speed gain of $160 - 250$ times in evaluating online our scheduling function as compared to the step-by-step reachability analysis method of [16]. This means that our algorithm is efficient in terms of speed for scheduling a self-triggered controller. The time of evaluating the scheduling function is also negligible since, for 100000 samples, the time of evaluation on M1 Mac processor is less than a fraction of a second. This is shown in Table 2.

7 RELATED WORK

The major focus of designing self-triggered control has been to ensure the stability of the closed-loop system. The initial works on self-triggered control focused on input-to-state stability for linear systems [20, 21, 26]. They dealt with linear plant and linear controller, and derived the self-triggered control law by exploiting the closed-form expressions of the trajectories of the closed-loop system. Wang and Lemmon addressed the problem of designing linear self-triggered control law for linear systems with the goal of enforcing a desired level of $\mathcal{L}_2$ gain on the closed-loop system [32, 33]. The design of a self-triggered control scheme for nonlinear systems to ensure input-to-state stability was addressed in [5]. Several papers (e.g. [7, 12, 19]) have shown that self-triggered control can reduce the computation load of model predictive control while ensuring the guarantee on the asymptotic stability of the closed loop systems.

Compared to stability, the design of self-triggered control ensuring safety has received limited attention in the literature. The safety aspect of self-triggered control was first considered in [9], where the authors presented a self-triggered control scheme for a nonlinear system with the goal of keeping the closed-loop system within a compact set. Recently, Yang et al. [35] propose a self-triggered control mechanism based on a combination of Control Lyapunov Function and Control Barrier Function. Their control mechanism ensures that the trajectory of the closed-loop system never violates the safety condition captured in a positive invariant set. In another recent work, Kooi et al. solve the safe self-triggered control problem for a constrained control system represented as a control differential inclusions with a continuous-time feedback law [22]. Their technique relies on the availability of a barrier function ensuring the forward invariance property with a degree of robustness against input perturbation. The major limitation of the above-mentioned works is that they overapproximate the rate of change of the system to compute the schedule in real-time. However, such an overapproximation can be conservative in the sense that it can reduce the possible range of the sampling time period. On the other hand, our paper precomputes the reachable sets without overapproximating the rate of state change.

The approach in Findeisen [23] proposes to apply a set-based reachability analysis online for discrete-time systems. The proposed reachability analysis can be accurate and compute larger bounds on the sampling period. However, the computation time will significantly increase such that real-time constraints may be violated. In Yulon et al. [15], evaluating a safe scheduling function requires solving a mixed integer linear program. But the computational complexity of mixed integer programming can be very high compared to a simple matrix multiplication followed by binary search in our algorithm (22). In Hashimoto et al. [17], a graph search algorithm is proposed for scheduling the sampling time so that safety constraints are satisfied. But this approach requires discretization of the state space, which can be intractable in real-time for higher dimensional systems.

Although previous approaches on self-triggered controller synthesis perform offline computation of invariant sets (or invariant functions) [9, 15, 17, 23, 35], they do not precompute offline the other sets reachable in between sampling times, which are also required for online scheduling. An important difference of the approach in this paper is that additional sets reachable in between sampling times, which are required for online scheduling, are also pre-computed offline, apart from the invariant set.

8 CONCLUSION

We presented a self-triggered control scheme for uncertain continuous-time linear systems. Our control mechanism relies on precomputed reachable sets of the closed-loop system, which ensures negligible online computation required for deciding the next trigger time without being too conservative. Our experimental results on standard higher dimensional control benchmarks and the computational complexity analysis establish the feasibility and scalability of our proposed method. In our future work, we plan to extend our technique to nonlinear hybrid systems.

REFERENCES

[1] Santosh Arvind Adimoolam. 2018. *A calculus of complex zonotopes for invariance and stability verification of hybrid systems.* Ph. D. Dissertation. Université

Grenoble Alpes.
[2] Santosh Arvind Adimoolam and Thao Dang. 2021. Safety Verification of Networked Control Systems by Complex Zonotopes. *Leibniz Transactions on Embedded Systems* (2021).
[3] Mohammad Al Khatib, Antoine Girard, and Thao Dang. 2017. Stability verification and timing contract synthesis for linear impulsive systems using reachability analysis. *Nonlinear Analysis: Hybrid Systems* 25 (2017), 211–226.
[4] João Almeida, Carlos Silvestre, and António M. Pascoal. 2010. Self-triggered state feedback control of linear plants under bounded disturbances. In *49th IEEE Conference on Decision and Control (CDC)*. 7588–7593.
[5] Adolfo Anta and Paulo Tabuada. 2010. To Sample or not to Sample: Self-Triggered Control for Nonlinear Systems. *IEEE Trans. Automat. Control* 55, 9 (2010), 2030–2042.
[6] K. J. Aström and B. Wittenmark. 1984. *Computer Controlled Systems: Theory and Design*. Prentice Hall.
[7] Florian David Brunner, Maurice Heemels, and Frank Allgöwer. 2016. Robust self-triggered MPC for constrained linear systems: A tube-based approach. *Automatica* 72 (2016), 73–83.
[8] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.-E. Arzen. 2003. How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime. *IEEE Control Systems Magazine* 23, 3 (2003), 16–30.
[9] M.D. Di Benedetto, S. Di Gennaro, and A. D'Innocenzo. 2011. Digital self triggered robust control of nonlinear systems. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*. 1674–1679.
[10] Parasara Sridhar Duggirala and Ashish Tiwari. 2013. Safety verification for linear systems. In *2013 Proceedings of the International Conference on Embedded Software (EMSOFT)*. 1–10.
[11] Iain Dunning, Joey Huchette, and Miles Lubin. 2017. JuMP: A Modeling Language for Mathematical Optimization. *SIAM Rev.* 59, 2 (2017), 295–320. https://doi.org/10.1137/15M1020575
[12] Alina Eqtami, Shahab Heshmati-alamdari, Dimos V. Dimarogonas, and Kostas J. Kyriakopoulos. 2013. Self-triggered Model Predictive Control for nonholonomic systems. In *European Control Conference (ECC)*. 638–643.
[13] Marcelo Forets and Christian Schilling. 2021. LazySets. jl: scalable symbolic-numeric set computations. *arXiv preprint arXiv:2110.01711* (2021).
[14] Gene F. Franklin, Michael L. Workman, and Dave Powell. 1997. *Digital Control of Dynamic Systems* (3rd ed.). Addison-Wesley Longman Publishing Co., Inc., USA.
[15] Yulong Gao, Pian Yu, Dimos V Dimarogonas, Karl H Johansson, and Lihua Xie. 2019. Robust self-triggered control for time-varying and uncertain constrained systems via reachability analysis. *Automatica* 107 (2019), 574–581.
[16] Antoine Girard. 2005. Reachability of uncertain linear systems using zonotopes. In *International Workshop on Hybrid Systems: Computation and Control*. Springer, 291–305.
[17] Kazumune Hashimoto, Shuichi Adachi, and Dimos V Dimarogonas. 2017. Self-triggered control for constrained systems: a contractive set-based approach. In *2017 American Control Conference (ACC)*. IEEE, 1011–1016.
[18] W. P. M. H. Heemels, J. H. Sandee, and P. P. J. Van Den Bosch. 2008. Analysis of event-driven controllers for linear systems. *Intl. J. of Control* 81, 4 (2008), 571–590.
[19] Erik Henriksson, Daniel E. Quevedo, Henrik Sandberg, and Karl Henrik Johansson. 2012. Self-Triggered Model Predictive Control for Network Scheduling and Control. *IFAC Proceedings Volumes* 45, 15 (2012), 432–438.
[20] Manuel Mazo Jr., Adolfo Anta, and Paulo Tabuada. 2009. On self-triggered control for linear systems: Guarantees and complexity. In *10th European Control Conference, ECC 2009, Budapest, Hungary, 23-26 August 2009*. IEEE, 3767–3772.
[21] Manuel Mazo Jr. and Paulo Tabuada. 2009. Input-to-state stability of self-triggered control systems. In *Proceedings of the 48th IEEE Conference on Decision and Control, CDC 2009, combined withe the 28th Chinese Control Conference, December 16-18, 2009, Shanghai, China*. IEEE, 928–933.
[22] David Kooi, Mohamed Maghenem, and Ricardo G. Sanfelice. 2021. Self-Triggered Control to Guarantee Forward Pre-Invariance with Uniformly Positive Inter-Event Times. In *2021 American Control Conference (ACC)*. 2278–2283.
[23] Markus Kögel and Rolf Findeisen. 2014. On self-triggered reduced-attention control for constrained systems. In *53rd IEEE Conference on Decision and Control*. 2795–2801.
[24] Colas Le Guernic and Antoine Girard. 2010. Reachability analysis of linear systems using support functions. *Nonlinear Analysis: Hybrid Systems* 4, 2 (2010), 250–262.
[25] Ibtissem Ben Makhlouf and Stefan Kowalewski. 2014. Networked Cooperative Platoon of Vehicles for Testing Methods and Verification Tools.. In *ARCH@ CPSWeek*. 37–42.
[26] Manuel Mazo, Adolfo Anta, and Paulo Tabuada. 2010. An ISS self-triggered implementation of linear controllers. *Automatica* 46, 8 (2010), 1310–1314.
[27] Stephen Prajna, Ali Jadbabaie, and George J. Pappas. 2007. A Framework for Worst-Case and Stochastic Safety Verification Using Barrier Certificates. *IEEE Trans. Autom. Control.* 52, 8 (2007), 1415–1428.
[28] Manuel Rauscher, Melanie Kimmel, and Sandra Hirche. 2016. Constrained robot control using control barrier functions. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 27–285.
[29] Nanang Syahroni, Young Bong Seo, and Jae Weon Choi. 2008. Depth control of autonomous underwater vehicle based on open control platform. *IFAC Proceedings Volumes* 41, 2 (2008), 3707–3712.
[30] Paulo Tabuada. 2007. Event-Triggered Real-Time Scheduling of Stabilizing Control Tasks. *IEEE Trans. Autom. Control.* 52, 9 (2007), 1680–1685.
[31] M. Velasco, P Marti, and J. M. Fuertes. 2003. The self-triggered task model for real-time control systems. In *Work In Progress Proceedings of RTSS*. 67–70.
[32] Xiaofeng Wang and Michael D. Lemmon. 2009. Self-Triggered Feedback Control Systems With Finite-Gain $\mathcal{L}_2$ Stability. *IEEE Trans. Automat. Control* 54, 3 (2009), 452–467.
[33] Xiaofeng Wang and Michael D. Lemmon. 2010. Self-Triggering Under State-Independent Disturbances. *IEEE Trans. Automat. Control* 55, 6 (2010), 1494–1500.
[34] Peter Wieland and Frank Allgöwer. 2007. CONSTRUCTIVE SAFETY USING CONTROL BARRIER FUNCTIONS. *IFAC Proceedings Volumes* 40, 12 (2007), 462–467.
[35] Guang Yang, Calin Belta, and Roberto Tron. 2019. Self-triggered Control for Safety Critical Systems Using Control Barrier Functions. In *American Control Conference (ACC)*. 4454–4459.
[36] Betsega Yosef. 2021. *Feedback Linearization with LQR Control Approach for Quadrotor Trajectory Tracking*. Ph. D. Dissertation. Addis Ababa Institute of Technology.
[37] Karl-Erik Åarzén. 1999. A simple event-based PID controller. *IFAC Proceedings Volumes* 32, 2 (1999), 8687–8692.