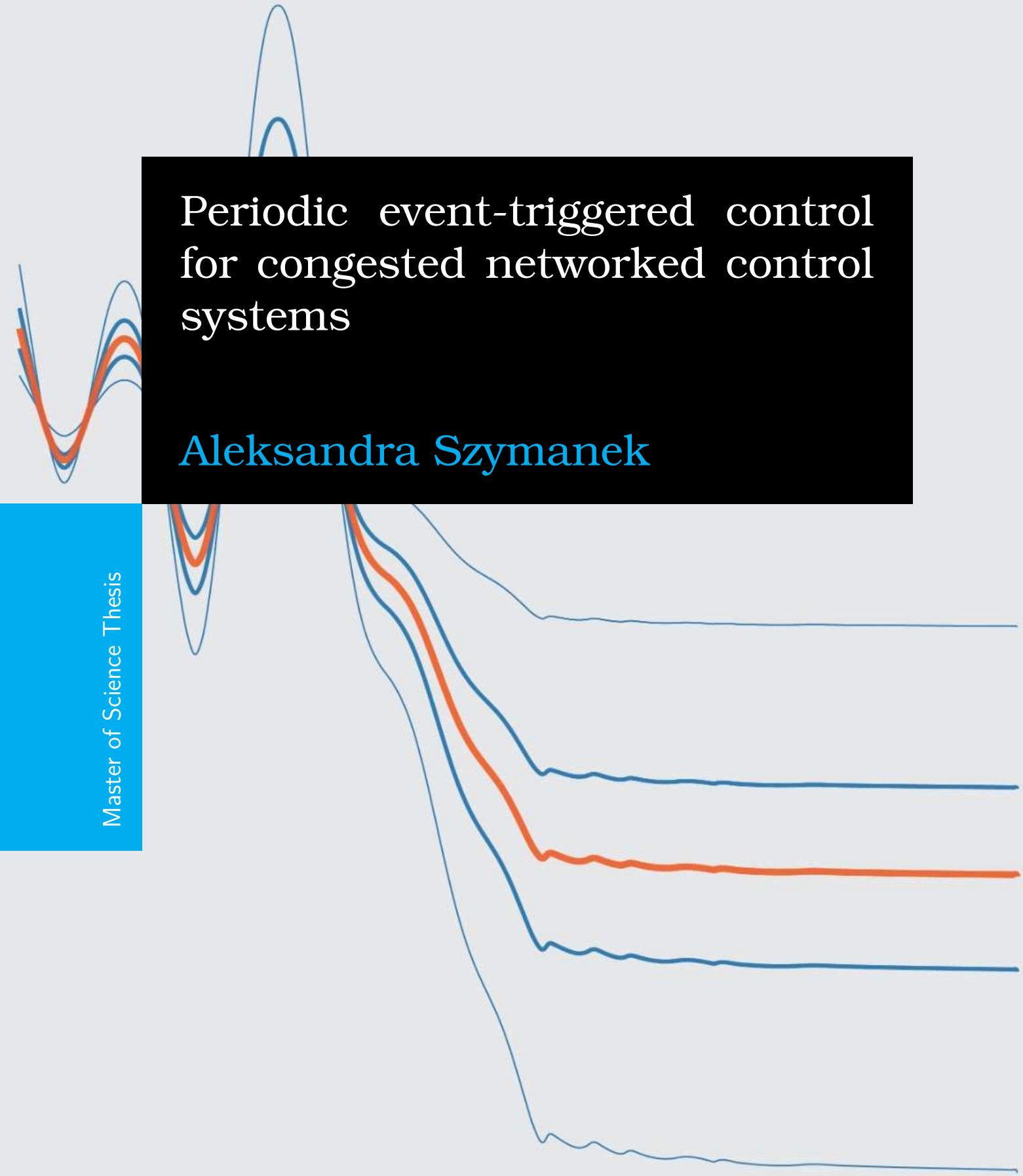
Periodic event-triggered control for congested networked control systems

Aleksandra Szymanek

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

Periodic event-triggered control for congested networked control systems

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Aleksandra Szymanek

July 28, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

With the recent development of control systems, event-triggered control (ETC) has been introduced to prevent unnecessary usage of resources, which often happens under time-based control implementations. This thesis presents a novel approach to periodic event-triggered control (PETC) that aims at reducing the number of transmissions between the controller and the sensors even further. This goal is particularly important in networked control systems (NCSs), where communication and computation resources are scarce. In this report, a relaxed triggering condition is introduced that relies on bounding the Lyapunov function of the continuous-time closed-loop system with an exponentially decaying function, rather than requiring its monotone decrease. The relaxed PETC achieves significantly less transmissions compared to existing PETC implementations. The thesis pushes the limit of event-triggered control even further, by introducing an algorithm for a scheduler of NCS that allows to skip some of the events. This can be seen as a 'last resort' approach, that postpones the transmission as much as possible. It is inspired by methods used in self-triggered control (STC) and scheduling event-based NCS. Reducing the communication between the plant and the controller introduces some trade-offs that are also discussed in this report. Finally, several modifications of presented ideas are given that can be applied depending on the main objectives on the performance of the control loop.

Table of Contents

List of Figures

List of Tables

Acknowledgements

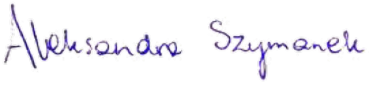
I would like to express my deep gratitude to Professor Manuel Mazo Jr. and Ir. Gabriel De Albuquerque Gleizer, my research supervisors, for the invaluable expertise, support and enthusiasm over this one-year project. It is thanks to them that it was a fascinating time, in which I have learned so much. I am also grateful to the whole research group for the helpful insights and constructive feedback.

In addition, I would like to thank my parents, who have made it possible for me to pursue my Master's Degree. Together with my friends, they are always there for me.

Special thanks to Maarten Van Vreeland, an amazing industrial designer, who helped me design the cover page of this thesis.

Delft, University of Technology
July 28, 2019

Aleksandra Szymanek



Chapter 1

Introduction

Fast development of technology in recent years allowed the design control systems in which sensors, controllers and actuators communicate through a shared digital network. This kind of design is called networked control system (NCS) and has numerous advantages. Most importantly the architecture is flexible, which results in an easier configuration of the elements of the network. Moreover the amount of wiring, as well as installation costs, can be reduced and the maintenance is less expensive. Because of that, NCSs have been used in many areas, including unmanned vehicles [4], remote surgeries [5], power plants [6], process control [7] and more.

However, the use of NCSs creates new challenges for the engineers. First of all, the communication resources are limited and usually only one element of the network can transmit at a particular time. This limits the number of control loops that can be connected using the same network and creates the possibility of communication conflicts when more than one element requests access to the network. Because of that, it is important to minimize the usage of the bandwidth by each controller. Another challenges introduced by NCSs are, e.g., communication delays or packet losses.

One solution to the problem of limited bandwidth resources can be event-triggered control (ETC), that first emerged in the '90s (e.g. [8] and [9]). Unlike in periodic control, which is a standard choice for digital implementation of a feedback controller [10], in ETC the communication between a sensor and a controller is aperiodic. The new measurement is sent to the controller only when a continuously monitored (usually state-dependent) triggering condition is violated. This way the number of communications through the network is reduced. A specific type of ETC is periodic ETC (PETC), which was introduced in [11]. Here, the triggering condition is verified only in periodic sampling instants, what is a more natural way for digital systems. Apart from a more direct design and easier analysis, PETC has the advantage of preventing Zeno behaviour by design, which may not be the case in a standard ETC ([12]).

Another aperiodic control strategy that aims at reducing the number of transmissions via network is self-triggered control (STC), first proposed in [13]. The idea is to predict the

next communication instant based on the triggering condition, instead of monitoring the plant. STC has the advantage over ETC that it does not require dedicated hardware used to monitor the triggering condition and it is easier to schedule, since the triggering time is known ahead. However, STC requires more computations performed online and is less robust against disturbances. Usually more conservatism has to be involved in order to include the effect of the disturbances into the prediction [14].

Aperiodic control strategies can sometimes provide better system performance, compared to periodic control, due to its ability to adjust to the current situation (e.g. how far from the equilibrium the states are or how large the disturbances are), as stated in [15]. Many of them, however, still involve a lot of conservatism, that can negatively affect the number of communication instants. One source of conservatism lies in imposing monotone decrease of the Lyapunov function of the system [15–20], while this is not necessary to maintain stability.

The main goal of this thesis is to find a PETC implementation that involves little conservatism and aims at minimizing the number of communication instants. The motivation behind it is reducing the usage of computational and communication resources in order to allow sharing one network by more plants, as well as saving energy. PETC is chosen as the primary control strategy, because of its robustness towards disturbances and less conservatism involved when compared to STC. In the case of highly congested NCSs, a scheduling technique for STC is used to postpone the event of the relaxed PETC even further. One of the drawbacks of STC, which is a considerable amount of computations online, is mitigated by introducing data structures that can be precomputed offline.

1-1 Notation

$\mathbb{R}^+$ denotes the set of non-negative real numbers, while $\mathbb{R}_0^+$ the same set including 0. $\mathbb{N}$ is a set of natural numbers excluding 0 and $\mathbb{N}_0$ including 0. For a vector $x \in \mathbb{R}^n$, we denote by $|x| := \sqrt{x^T x}$ its 2-norm. For a matrix $A \in \mathbb{R}^{n\times m}$, we denote by A^T its transpose. For a symmetric square matrix $P \in \mathbb{R}^{n\times n}$, we write $P \succ 0$ ($P \succeq 0$) if P is positive (semi–)definite. By $\lambda_m(P)$ and $\lambda_M(P)$ we denote the minimum and maximum eigenvalue of P, respectively. Solutions of an autonomous system with state x and initial condition x_0 are denoted by $x_{x_0}(t)$; if it has exogenous inputs u and δ, a trajectory is denoted by $x_{x_0 u\delta}(t)$ For a signal $w : \mathbb{R}_0^+ \to \mathbb{R}^{n_w}$, its $\mathcal{L}_2$-norm is denoted by $||w||_{\mathcal{L}_2} := (\int_0^\infty w(t)^2 dt)^{\frac{1}{2}}$, while its $\mathcal{L}_\infty$-norm is denoted by $||w||_\infty := \text{ess sup}_t |w(t)|$. We say $w \in \mathcal{L}_\infty$ if $||w||_\infty < \infty$. A function $\beta : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is said to be a $\mathcal{K}$-function if it is continuous, strictly increasing and $\beta(0) = 0$. Also, it is said to be a $\mathcal{K}_\infty$-function if it is a $\mathcal{K}$-function and $\beta(s) \to \infty$ as $s \to \infty$. For any function $f : \mathbb{R}_0^+ \to \mathbb{R}^n$ and $t \geq 0$, we use $f(t^+)$ to denote the limit $f(t^+) = \lim_{s\to t, s>t} f(s)$.

1-2 Problem formulation

Consider a linear time-invariant (LTI) system under a standard ETC strategy of the form:

$$\frac{d}{dt}x = A^p x + B^p \hat{u} + E\delta \tag{1-1}$$

$$\hat{u} = K\hat{x} \tag{1-2}$$

where $x \in \mathbb{R}^{n_x}$ denotes the states of the plant, $\hat{u} \in \mathcal{U} \subset \mathbb{R}^{n_u}$ is the control input, $\delta \in \mathcal{W} \subset \mathbb{R}^{n_\delta}$ is the unknown disturbance vector and $\hat{x} \in \mathbb{R}^{n_x}$ is the last measurement available to the controller. Matrices A^p, B^p and E are known and matrix K is designed such that the real part of the eigenvalues of $(A^p + B^p K)$ is negative. The linear feedback control law (1-2) is implemented with sample and hold strategy and is updated only when a new measurement $\hat{x}$ is sent to the controller.

Before stating the objective of this work, let us define the necessary stability and performance notions.

Definition 1-2.1 (GES)**.** *The system (1-1)-(1-2) is said to be globally exponentially stable (GES), if there exist $\sigma \in \mathbb{R}^+$ and $\rho \in \mathbb{R}^+$, such that for any $x(0) = x_0 \in \mathbb{R}^{n_x}$ and $\delta \equiv 0$ all corresponding solutions to (1-1) satisfy: $|x(t)| \leq \sigma |x_0| e^{-\rho t}$ for all $t \in \mathbb{R}_0^+$.*

Definition 1-2.2 (EISS)**.** *The system (1-1) is said to be exponentially input-to-state stable (EISS), if there exist $\sigma \in \mathbb{R}^+$, $\rho \in \mathbb{R}^+$ and $\gamma \in \mathcal{K}_\infty$, such that for any $x(0) = x_0 \in \mathbb{R}^{n_x}$ and $\delta \in \mathcal{L}_\infty$ all corresponding solutions to (1-1) satisfy: $|x(t)| \leq \sigma |x_0| e^{-\rho t} + \gamma(||\delta||_\infty)$ for all $t \in \mathbb{R}_0^+$. Furthermore, if there is a $g < \infty$ satisfying $\gamma(d) \leq gd, \forall d \in \mathbb{R}^+$, we call g the $\mathcal{L}_\infty$ gain from disturbance to state.*

As described in the last chapter of my literature survey ([21]) the primary goal of this MSc project is to design a PETC strategy for (1-1), such that the number of communications between the sensor and the controller is reduced compared to existing event-triggered implementations. The roadmap of the project consists of 3 main parts:

1. **Designing a relaxed triggering condition for PETC**, that aims at postponing the occurrence of the event as much as possible, while still ensuring global exponential stability. In this part, a standard PETC is considered and the novelty lies mainly in the new triggering condition. The results of this step of the roadmap can be found in chapter 2. This part is also the subject of [22] that has been accepted to the Conference on Decision and Control in 2019.

2. **Predicting the time of several consecutive events in the future by a scheduler**, if the control action is not updated. There exist systems with an oscillatory behaviour that experience multiple events according to the triggering condition from the first part. This is illustrated in Fig. 1-1, which shows how the value of the triggering condition $\mathcal{C}$ could possibly evolve in time. In the classical ETC, the control action would be updated as soon as $\mathcal{C} > 0$, what corresponds to the first red dot in Fig. 1-1. However, if the control action is not updated, chances are that more events will occur in the future and more red dots will be present in the plot. The idea is to find a method of predicting their occurrences in a way that the number of computations performed online is kept small such that our PETC does not transform into STC. This part is inspired by methods used in scheduling of ETC and STC and therefore its end result is a scheduler that is able to efficiently predict future events. Detailed explanation on this topic can be found in chapter 3.

3. **Relaxed PETC combined with the scheduler**. This part merges the concepts introduced in the earlier points of the roadmap. The information from the intelligent sensory system under PETC is passed down to the scheduler. Based on this information

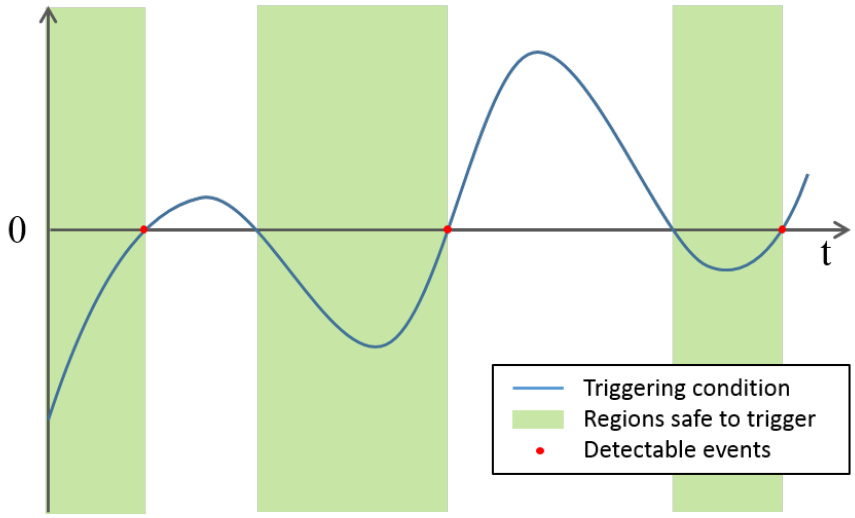


Figure 1-1: A possible trajectory of the triggering condition.

the next events can be predicted and the scheduler can grant access to the network only during the time interval when, e.g., the last predicted event will happen. This way, we guarantee that the control update will occur in a time interval that is safe to trigger (green regions in Fig. 1-1). All the previous events can be ignored which in turn contributes to reducing the overall number of communications through the network. This concept is the interest of chapter 4.

Chapter 2

A relaxed triggering condition for PETC

This chapter explains in detail how the first step of our roadmap has been done. The new triggering condition that reduces the number of communication instants is designed for a standard PETC, and hence some preliminaries about this control strategy are given first. After presenting the main results, a comparison of all the discussed control strategies is given. For an introduction to event-triggered control (ETC) the reader is referred to [10] or [21].

The subject of this chapter has been accepted to the 58th Conference on Decision and Control (2019) [22].

2-1 Periodic event-triggered control

2-1-1 Sampling strategy

Periodic event-triggered control (PETC) is a type of event-triggered control (Fig. 2-1) where the triggering condition is checked periodically. At every sampling instant $t_k = k\Delta$, $k \in \mathbb{N}$

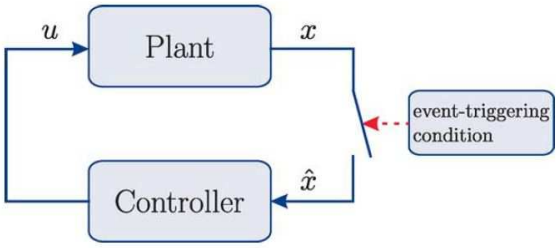


Figure 2-1: Schematic of ETC closed loop [1].

with Δ being some properly chosen sampling interval, an intelligent sensory system tests the current sample of the system's state against a given triggering condition $\mathcal{C} : \mathbb{R}^{2n_x} \to \mathbb{R}$. If it is enabled, the current measurement is sent to the controller that updates the control action. This situation is called an event and the time at which it happens - triggering time t_i, $i \in \mathbb{N}$. The control input is held constant until the next event, what can be described with:

$$u(t) = u(t_i) = \hat{u} = Kx(t_i) = K\hat{x}(t), \quad \forall t \in [t_i, t_{i+1}), i \in \mathbb{N}, \tag{2-1}$$

$$\hat{x}(t) = \begin{cases} x(t_k), & \text{when } \mathcal{C}(x(t_k), \hat{x}(t_k)) > 0, \\ \hat{x}(t_k), & \text{when } \mathcal{C}(x(t_k), \hat{x}(t_k)) \leq 0. \end{cases} \tag{2-2}$$

If the value of the triggering condition is below 0, then the system does not require attention. The new measurement is only sent to the controller when the value of the condition at the sampling time t_k is positive. The difference with respect to the standard ETC (called also continuous ETC or CETC) is that the triggering times can only be multiples of the sampling interval Δ. This way Zeno behaviour is prevented in PETC by design. Furthermore, since the condition is only checked in periodic sequence of measurement times $\{t_k\}_{k\in\mathbb{N}}$, it is an inequality, as opposite to the equality condition in CETC. It is due to the fact that the exact moment of equality could be missed by an intelligent sensory system, because it does not monitor the condition continuously.

2-1-2 Existing triggering conditions

The choice of a triggering condition is strictly related to the system's performance that one would like to achieve. One of its most important indicators is the convergence rate of the Lyapunov function, which is a measure of how fast the states converge to the equilibrium. In an 'ideal' continuous-time closed loop with state feedback, the system has the following dynamics:

$$\dot{x} = A^p x + B^p K x$$

The eigenvalues of $(A^p + B^p K)$ are negative and the system is asymptotically stable. In this case there exists a Lyapunov function $V(x) = x^T P x$ with some positive definite matrix P, which satisfies

$$(A^p + B^p K)^T P + P(A^p + B^p K) = -Q, \tag{2-3}$$

with $Q \succ 0$. Therefore, the Lyapunov function decreases with a rate specified by the matrix Q. We define the *decay rate* of the Lyapunov function as the largest $\lambda_0 \in \mathbb{R}^+$ that satisfies $V(x_{x_0}(t)) \leq V(x_0)e^{-\lambda_0 t}, \forall t \in \mathbb{R}_0^+, \forall x_0 \in \mathbb{R}^{n_x}$. For the continuous-time closed-loop system and given P and Q satisfying (2-3), this is $\lambda_0 = \lambda_m(P^{-1}Q)$.

The dynamics of the Lyapunov function is dependent on the values of x and $\hat{x}$, that can be put together in one vector $\xi := \begin{bmatrix} x^T & \hat{x}^T \end{bmatrix}^T$. A significant number of existing triggering conditions can be expressed in a quadratic form: $\mathcal{C}(\xi(t_k)) := \xi^T(t_k)Q\xi(t_k)$, $Q \in \mathbb{R}^{n_\xi \times n_\xi}$, which will also be adopted in this work. The advantage of this formulation is that most of the triggering conditions for a CETC can be written as quadratic conditions for PETC. Moreover, in [1] one can find a set of linear matrix inequalities (LMIs) to check guarantees on stability and performance of the system under PETC, if the condition is formulated as a quadratic one and the whole system as an impulsive one.

A common triggering condition used in ETC is based on the **state error**. The control action is updated when the difference between the last sent measurement and current state exceeds a state-dependent threshold. After translating it to PETC, one obtains:

$$|\hat{x}(t_k) - x(t_k)| > \sigma|x(t_k)| \quad \rightarrow \quad Q = \begin{bmatrix} (1-\sigma)^2 I & -I \\ -I & I \end{bmatrix} \tag{2-4}$$

As explained in [19] for a standard ETC, if $\sigma \in [0, 1)$ then the Lyapunov function is guaranteed to be decreasing.

Another triggering condition [1] is **based directly on the Lyapunov function** of the discretized system (1-1):

$$x(t_{k+1}) = Ax(t_k) + Bu(t_k) \tag{2-5}$$

with

$$A = e^{A^p \Delta} \quad \text{and} \quad B = \int_0^{\Delta} e^{A^p s} B^p ds. \tag{2-6}$$

When disturbances are not present, the system is fully deterministic. The idea is to update the control action when the predicted value of the Lyapunov function at the next sampling instant t_{k+1} is greater than the current one. This ensures that the Lyapunov function is decreasing from sample to sample and can be written with:

$$\begin{gathered} (Ax(t_k) + BK\hat{x}(t_k))^T P (Ax(t_k) + BK\hat{x}(t_k)) > \beta x^T(t_k) P x(t_k), \\ Q = \begin{bmatrix} A^T PA - \beta P & A^T PBK \\ (BK)^T PA & (BK)^T PBK \end{bmatrix}, \end{gathered} \tag{2-7}$$

where $\beta \in [0, 1)$ is a design parameter. After choosing all the required parameters, both above mentioned triggering conditions can be checked through solving LMIs. If there is a feasible solution, the PETC implementation is guaranteed to be stable with a chosen convergence rate ρ (Definition 1-2.1) and $\mathcal{L}_2$-gain from disturbance to state smaller than or equal to a desired value γ. An important advantage of this method is that it can be applied to any PETC with a quadratic triggering condition and LMIs can be solved efficiently. As far as disadvantages are concerned, LMIs are prone to numerical errors and introduce some conservatism. As a result, the true convergence rate is bigger than prescribed ρ what leads to an increase in the number of events.

One of the very few ETC implementations where the monotone decrease of the Lyapnuov function is not required was presented in [23]. Here, it is only bounded by a piecewise continuous linear function, which is sufficient for maintaining stability and results in a more relaxed triggering condition. However, it was only done for CETC and because of that the effect of disturbances can be neglected. The main result of this chapter uses a similar idea, but it is done for PETC and does not require an internal clock.

2-1-3 Dynamic PETC

Dynamic triggering [24] is one of the proposed control strategies in which the monotone decrease of the Lyapunov function is not required. Instead, the Lyapunov function should be decreasing on average. This is equivalent to saying that the triggering condition can be

violated at some instances, provided that it is non-positive on average. For that, a dynamic variable η has to be introduced, which for the case of CETC takes the form of:

$$\dot{\eta} = -\beta(\eta) - \mathcal{C}(x(t), \hat{x}(t)), \quad \eta(0) = \eta_0 \tag{2-8}$$

where β is a locally Lipschitz continuous class $\mathcal{K}_\infty$ function and $\eta_0 \in \mathbb{R}_0^+$. This dynamic variable can be seen as a filtered value of the triggering condition or as a buffer that stores unnecessary decrease of the Lyapunov function to balance it out when it is increasing. Events occur according to the new triggering condition:

$$t_0 = 0, \quad t_{i+1} = \inf\{t > t_i | \eta(t) - \theta\mathcal{C}(x(t), \hat{x}(t)) \leq 0, t \in \mathbb{R}\} \tag{2-9}$$

where $\theta \in \mathbb{R}_0^+$ is an additional design parameter. The greater the value of θ the closer this event-triggered mechanism (ETM) is to the static one.

Dynamic triggering has been also applied to PETC ([2]), where the main idea is the same but the dynamics of η are slightly modified.

$$\begin{aligned} &\frac{d}{dt}\eta(t) = -2\rho\eta, \quad t \in (t_k, t_{k+1}) \\ &\quad \eta(t^+) = \eta_T(\xi(t_k), \eta(t)), \quad t \in \{t_i\}_{i\in\mathbb{N}} \\ &\quad \eta(t^+) = \eta_N(\xi(t_k), \eta(t)), \quad t \in \{t_k\}_{k\in\mathbb{N}} \backslash \{t_i\}_{i\in\mathbb{N}} \end{aligned} \tag{2-10}$$

where $\eta_T : \mathbb{R}^{2n_x} \times \mathbb{R} \to \mathbb{R}$ and $\eta_N : \mathbb{R}^{2n_x} \times \mathbb{R} \to \mathbb{R}$. Similarly to [1], stability and performance of the system under dynamic PETC can also be checked with an LMI-based method. In this case also the functions η_N and η_T can be designed using the solution of the LMIs . The new sequence of triggering times is obtained from:

$$t_0 = 0, \quad t_{i+1} = \min\{t > t_i | \eta_N(t) \leq 0 \wedge \mathcal{C}(x(t), \hat{x}(t)) \geq 0, t = k\Delta, k \in \mathbb{N}\} \tag{2-11}$$

It is clear that the new triggering condition depends not only on the states but also the dynamic variable that keeps some information about the evolution of the Lyapunov function from the past. The triggering conditions from the previous subsection do not use this kind of information and therefore they are called static. However, any static condition $\mathcal{C}(x(t), \hat{x}(t))$ can be transformed to a dynamic one by incorporating the variable η.

To clearly see the difference between static and dynamic ETMs, both of them can be written in a similar way. For the case of quadratic triggering condition, the triggering conditions are the following:

$$\textbf{Static: } \xi^T Q \xi > 0, \quad \textbf{Dynamic: } \theta\xi^T Q\xi > \eta. \tag{2-12}$$

In the case of dynamic triggering the Lyapunov function is allowed to be increasing at some times, often resulting in a smaller number of events. Moreover, it does not degrade to time-triggered periodic communication when the state approaches its equilibrium in presence of disturbances [12]. The sampling period in periodic control should be chosen smaller than or equal to the minimum inter-event time. Since dynamic triggering does not degrade to time-triggered communication, effectively average inter-event time in this ETC strategy stays greater than the minimum inter-event time independently from the system's state.

2-2 Relaxed triggering condition

Unlike most of the existing triggering conditions, our relaxed triggering condition will not impose that the derivative of the Lyapunov function is always negative. Instead, like the STC in [25], an exponentially decaying function will bound the actual Lyapunov function: conceptually, the triggering condition is $x_{x_0}(t)^T P x_{x_0}(t) > x_0^T P x_0 e^{-\lambda t}$, with $0 < \lambda < \lambda_0$ being the desired decay rate. However, some modifications are made for PETC implementability. First, an auxiliary discrete-time variable $\eta : \mathbb{R} \to \mathbb{R}^{n_x}$ is introduced:

$$\eta(0) = P^{\frac{1}{2}} x(0), \tag{2-13a}$$

$$\eta(t_{k+1}) = I e^{-0.5\lambda\Delta} \eta(t_k^+), \tag{2-13b}$$

$$\eta(t_i^+) = P^{\frac{1}{2}} x(t_i), \tag{2-13c}$$

where $\{t_i\}_{i\in\mathbb{N}}$ are the triggering times. Denoting $\zeta := \begin{bmatrix} x^T & \hat{x}^T & \eta^T \end{bmatrix}^T$, the sequence of triggering times t_i is obtained from the following:

$$\begin{aligned} t_0 &= 0, \\ t_{i+1} &= \inf\{t > t_i | \zeta(t)^T Q_1 \zeta(t) > 0 \vee \zeta(t)^T Q_2 \zeta(t) > 0 \vee t = t_i + N_{max}\Delta, t = k\Delta, k \in \mathbb{N}\}, \end{aligned} \tag{2-14}$$

where $N_{max}\Delta$ is a designed maximum inter-event time and

$$Q_1 := \begin{bmatrix} P & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -I \end{bmatrix},$$

$$Q_2 := \begin{bmatrix} A^T P A & A^T P B K & 0 \\ (BK)^T P A & (BK)^T P B K & 0 \\ 0 & 0 & -I e^{-\lambda\Delta} \end{bmatrix},$$

$$A := e^{A^p \Delta}, \quad B := \int_0^\Delta e^{A^p s} B^p ds.$$

An intuitive explanation of this triggering condition is that $\zeta(t_k)^T Q_1 \zeta(t_k) > 0$ checks if, at the current sampling time t_k, the Lyapunov function of our interest is above the bound, while $\zeta(t_k)^T Q_2 \zeta(t_k) > 0$ checks if this will happen at the next sampling time t_{k+1}. If any of these two conditions is true, the mechanism triggers. It also triggers if neither is true, but the maximum inter-event time is reached. The one-step-ahead prediction from the second triggering condition is made without considering disturbances. In the disturbance-free case it prevents the Lyapunov function from exceeding the bound $\eta^T \eta$, which leads to tighter performance guarantees.

The proposed triggering condition has three parameters that have to be chosen: the triggering condition sampling interval Δ, a desired decay rate λ of the Lyapunov function, and the maximum inter-event number of steps N_{max}. Matrix P is taken from the Lypaunov function of the 'ideal' closed loop system. λ has to be chosen smaller than the decay rate of an 'ideal' closed-loop λ_0. This is needed in order to compute the minimum inter-event time of a CETC

implementation, which, in turn, influences the choice for Δ. For every chosen decay rate $\lambda < \lambda_0$, there exists a guaranteed minimum inter-event time τ_{min} given by [25, Lemma 4.1]

$$\tau_{min} = \min\{\tau \in \mathbb{R}^+ : \det M(\tau) = 0\}, \tag{2-15}$$

where:

$$M(\tau) := C(e^{F^T\tau}C^T P C e^{F\tau} - C^T P C e^{-\lambda\tau})C^T,$$
$$F := \begin{bmatrix} A^p + B^p K & B^p K \\ -A^p - B^p K & -B^p K \end{bmatrix}, \quad C := \begin{bmatrix} I & 0 \end{bmatrix}.$$

Once the minimum inter-event time is computed, the sampling time Δ has to be chosen such that $\Delta < \tau_{min}$. These guidelines for choosing λ and Δ are the necessary conditions for stability of the system under PETC. A detailed discussion on the influence of all the parameters on the system performance is given in the next section.

2-2-1 Stability analysis

The two theorems of this section show GES and EISS, respectively, of the PETC system with the relaxed triggering condition (2-14). Before introducing them, we have to define the following function:

$$g(\Delta, N_{max}) := e^{\frac{\omega\mu\Delta}{\mu-\omega}}\left(e^{\lambda\Delta} + e^{\lambda N_{max}\Delta}\right)^{\frac{\mu}{\mu-\omega}} \cdot \left(1 + e^{\lambda N_{max}\Delta}\right)^{\frac{-\omega}{\mu-\omega}} \tag{2-16}$$

where

$$\omega := \lambda_M(G), \quad \mu := \lambda_m(G),$$
$$G := \begin{bmatrix} P^{\frac{1}{2}}A^p P^{-\frac{1}{2}} + (P^{\frac{1}{2}}A^p P^{-\frac{1}{2}})^T & P^{\frac{1}{2}}B^p K P^{-\frac{1}{2}} \\ (P^{\frac{1}{2}}B^p K P^{-\frac{1}{2}})^T & 0 \end{bmatrix}.$$

Theorem 2.1. *If $\lambda < \lambda_0$ and $\Delta < \tau_{min}$, the sequence of control updates times given by (2-14) renders the closed loop system (1-1) GES with:*

$$\sigma = \left(\frac{\lambda_M(P)}{\lambda_m(P)}\right)^{\frac{1}{2}}(g(\Delta, N_{max}))^{\frac{1}{2}}, \quad \rho = \frac{1}{2}\lambda. \tag{2-17}$$

Now, we can give performance guarantees in the presence of additive bounded disturbances in the following theorem.

Theorem 2.2. *If $\lambda < \lambda_0$, $\Delta < \tau_{min}$ and $\delta \in \mathcal{L}_\infty$, the sequence of control update times given by (2-14) yields the closed loop system (1-1) EISS with*

$$\gamma(||\delta||_\infty) = \frac{\lambda_M(P)}{\lambda_m(P)}\int_0^\Delta |e^{A^p r}E|dr \cdot \left(\frac{(g(\Delta, N_{max}))^{\frac{1}{2}}}{1 - e^{-\frac{1}{2}\lambda\Delta}} + 1\right)||\delta||_\infty.$$

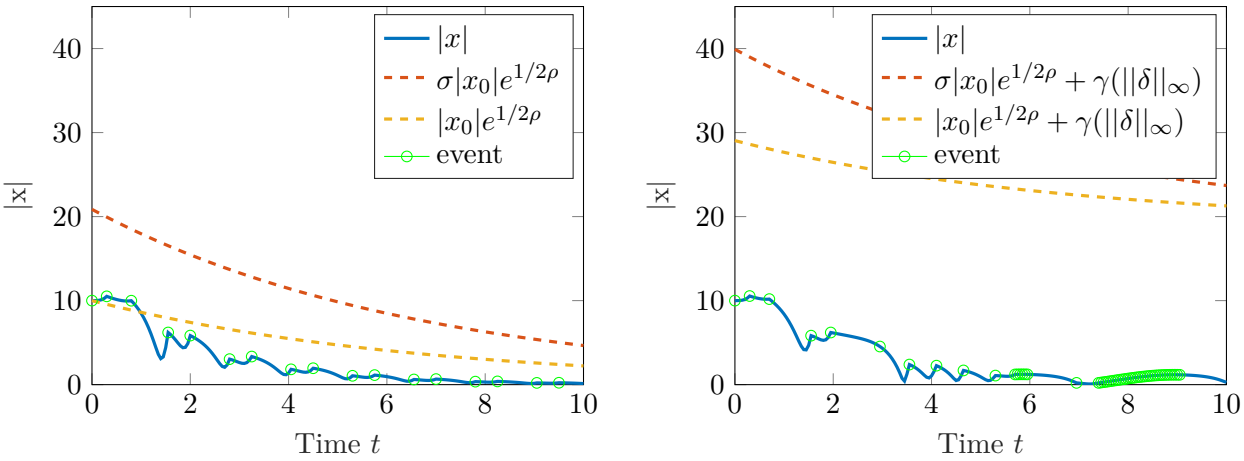
Proofs can be found in Appendices: A-1 and A-2.

2-2-2 Numerical example

To illustrate how the relaxed triggering condition reduces the number of communication instants we compare our approach with triggering conditions (7) and (14) from [1], where the monitored Lyapunov function is required to be decreasing from sample to sample, and with dynamic PETC [2]. As an example we take the plant from [19]:

$$\frac{d}{dt}x = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \delta. \tag{2-18}$$

We set $x_0 = \begin{bmatrix} 10 & 0 \end{bmatrix}^T$ as initial condition and a stabilizing controller of the form (2-1) with $K = \begin{bmatrix} 1 & -4 \end{bmatrix}$. The associated Lyapunov function and matrix Q satisfying (2-3) were chosen to be $P = \begin{bmatrix} 1 & 0.25 \\ 0.25 & 1 \end{bmatrix}$ and $Q = \begin{bmatrix} 0.5 & 0.25 \\ 0.25 & 1.5 \end{bmatrix}$, which give $\lambda_0 \approx 0.4836$. We chose the desired decay rate $\lambda = 0.3$, thus $\rho = 0.15$. The corresponding minimum inter-event time according to Lemma A.1 is $\tau_{min} = 0.3$, and so we set $\Delta = 0.05 \leq \tau_{min}$. The maximum inter-event time was taken as 1 second, thus $N_{max} = 20$. Using Theorems 2.1 and 2.2, the EISS parameters according to Definition 1-2.2 are $\sigma = 2.0864$ and $\gamma(d) = gd, g = 19.0437$, which can be seen in Fig. 2-2. We can see that some conservatism is introduced by the parameter σ (Fig. 2-2(a)), which accounts for the worst case scenario where the Lyapunov function forms spikes in between the samples. However, this hardly even happens and, if we neglect the effect of σ, the convergence of the states is nicely approximated by the bound. For the case with disturbances, the effect of the additional term cannot be neglected but the guaranteed gain is finite and around 20 times bigger than the $\mathcal{L}_\infty$-norm resulted by the sinusoidal disturbance $\delta(t) = \sin(t)$.



(a) Convergence in a disturbance-free case

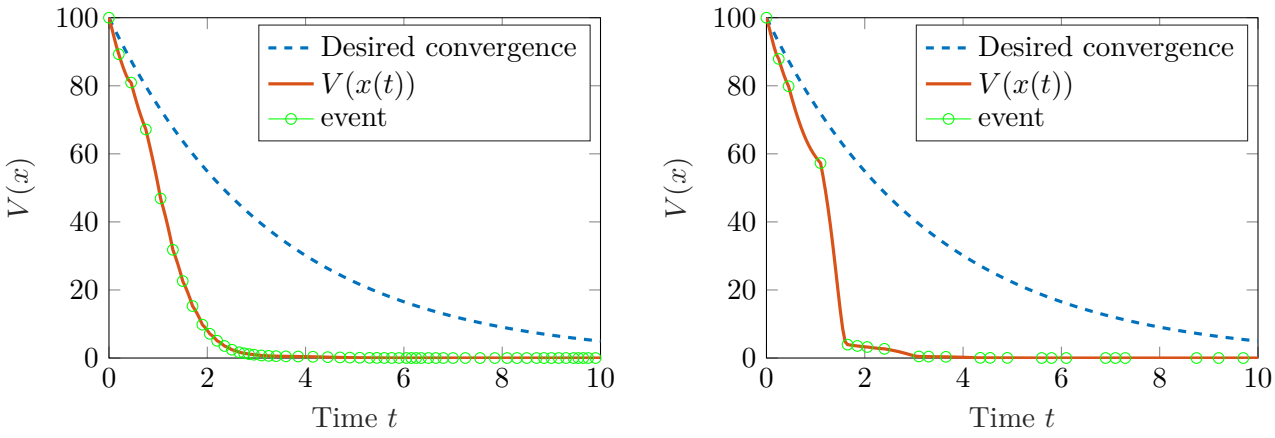
(b) Convergence with $\delta = \sin(t)$

Figure 2-2: Guaranteed bounds on the system's convergence under the relaxed triggering condition with $\Delta = 0.05$, $\rho = 0.15$ and $N_{max} = 20$.

For the triggering conditions (7) and (14) from [1] parameters σ, β, respectively, were chosen to minimise the number of events, while still ensuring the LMIs presented therein to be

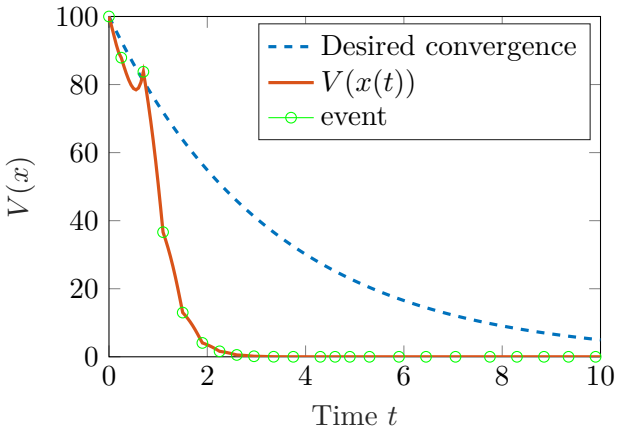
feasible, with $\rho = 0.3$. The values we found are $\sigma = 0.2$ and $\beta = 0.985$. Similar approach was applied in finding σ for dynamic PETC, where $\sigma = 0.18$ was found. Additionally, the value of acceptable $\mathcal{L}_2$ gain (θ) from disturbance to state had to be chosen. For a fair comparison we set it to $\theta = 20$, such that it is of comparable order of magnitude to the $\mathcal{L}_\infty$ gain resulting from our PETC.

Figure 2-3 shows the evolution of the monitored Lyapunov function for the four triggering conditions, with $\delta \equiv 0$.

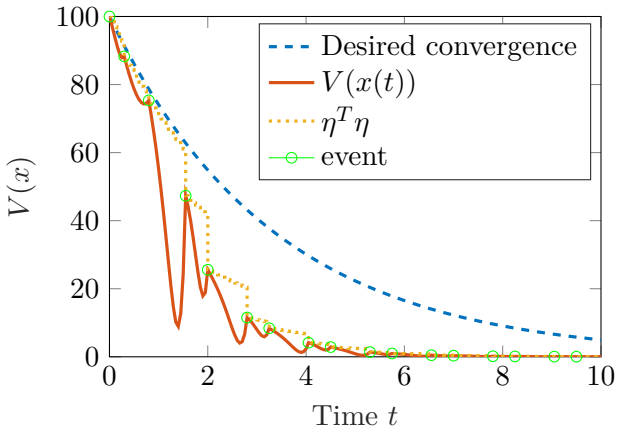


(a) Triggering condition (7) from [1]

(b) Triggering condition (14) from [1]



(c) Dynamic PETC from [2]



(d) Relaxed triggering condition

Figure 2-3: The evolution of the monitored Lyapunov function for three different triggering conditions. Common parameters are: $\Delta = 0.05$ and $\rho = 0.15$. For (7) from [1] $\sigma = 0.2$, for (14) from [1] $\beta = 0.985$, for dynamic PETC from [2] $\sigma = 0.18$ and for our relaxed triggering condition $N_{max} = 20$.

Table 2-1 presents a comparison among the number of events from all four triggering conditions, for other desired convergence rates and $\Delta = 0.05$. The average number of communications and their standard deviations were computed based on 10 simulations for each case. One set of 10 different initial conditions was randomly chosen, such that both states ranged from -10 to 10. For all triggering conditions, the parameters were chosen such that the number of

Table 2-1: Average (μ) and standard deviation (σ) of number of communications for different triggering conditions and decay rates

ρ	This work	(7) from [1]	(14) from [1]	Dynamic PETC[2]
0.1	$\mu = \mathbf{17.6}$ $\sigma = 0.699$	$\mu = 52.8$ $\sigma = 0.422$	$\mu = 19.8$ $\sigma = 0.789$	$\mu = 22.4$ $\sigma = 1.647$
0.125	$\mu = \mathbf{17.3}$ $\sigma = 0.823$	$\mu = 53.5$ $\sigma = 0.527$	$\mu = 23.2$ $\sigma = 1.317$	$\mu = 26.3$ $\sigma = 2.71$
0.15	$\mu = \mathbf{17.6}$ $\sigma = 0.843$	$\mu = 56.6$ $\sigma = 0.516$	$\mu = 22.2$ $\sigma = 1.751$	$\mu = 28.6$ $\sigma = 3.026$
0.175	$\mu = \mathbf{17.6}$ $\sigma = 0.843$	$\mu = 58.7$ $\sigma = 0.483$	$\mu = 21.3$ $\sigma = 1.252$	$\mu = 29.2$ $\sigma = 2.3$
0.2	$\mu = \mathbf{17.6}$ $\sigma = 0.843$	$\mu = 59.8$ $\sigma = 0.422$	$\mu = 23$ $\sigma = 2.582$	$\mu = 30.3$ $\sigma = 1.636$
0.225	$\mu = \mathbf{17.7}$ $\sigma = 0.823$	$\mu = 62.9$ $\sigma = 0.316$	$\mu = 33.2$ $\sigma = 2.348$	$\mu = 32.4$ $\sigma = 0.966$

transmissions is as small as possible. For all cases, the relaxed triggering condition yielded the fewest number of communications. This performance difference is bigger the faster the desired convergence is. Moreover, our triggering condition results in lower standard deviation compared to PETCs with second and third best number of communications.

A possible disadvantage of the relaxed triggering condition are the the oscillations of the monitored Lyapunov function, as can be seen in Figure 2-3(d); in our simulations, they corresponded to oscillations in the trajectories of the plant states (Figure 2-4), which can be undesired in many applications. This may be caused by the fact that the original controller is designed for a faster convergence rate than the one imposed by the triggering condition. If we choose the decay rate to be almost equal to λ_0, namely $\lambda = 0.45$ ($\rho = 0.225$), the oscillations are reduced to some extent. Studying the balance between relaxing the triggering condition for less updates, and the resulting oscillations of the trajectories is subject for future work but some ideas and preliminary results are discussed in the next section.

Figure 2-5(a) illustrates how the relaxed PETC deals with disturbances. Here, ρ was set to 0.225 and disturbance $\delta(t) = \sin(t)$ was present throughout the whole simulation. The total number of events in this case was 39, which was mostly because the disturbance started to dominate the dynamics at the final part of the simulation. Here, the EISS parameters are $\sigma = 2.1818$ and $\gamma(d) = gd, g = 13.3278$. Dynamic PETC (Fig. 2-5(b)) seems to be more robust, sampling-wise, at these regions, having had a total of 32 events. Parameter σ was obtain from LMIs for which the $\mathcal{L}_2$-gain was set to 14.

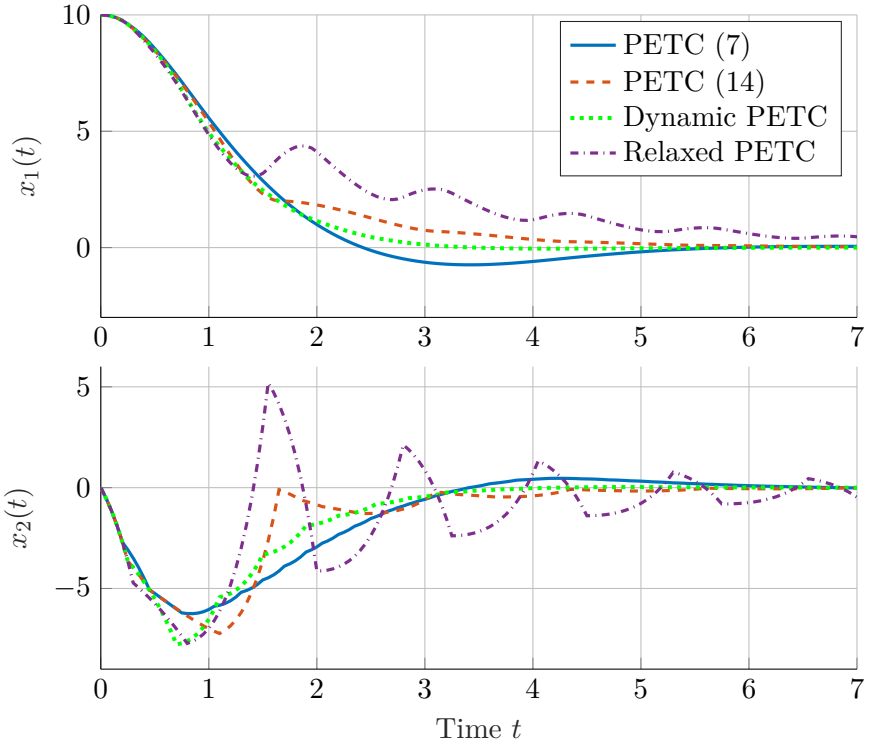
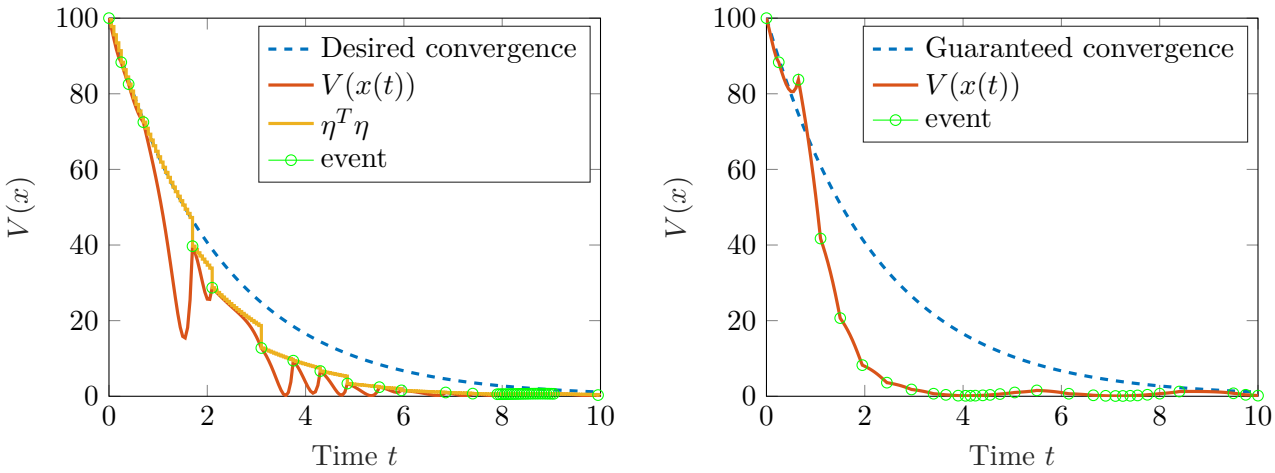


Figure 2-4: System's trajectories under different PETC implementations.



(a) Relaxed triggering condition with $N_{max} = 20$

(b) Dynamic PETC from [2] with parameter $\sigma = 0.16$

Figure 2-5: Evolution of the Lyapunov function with $\Delta = 0.05$, $\rho = 0.225$ and $\delta = \sin(t)$ for two different triggering conditions.

2-3 Approaches to reduce oscillations

The following section discusses the problem of the oscillatory system trajectories under relaxed PETC and reviews two methods of mitigating this issue.

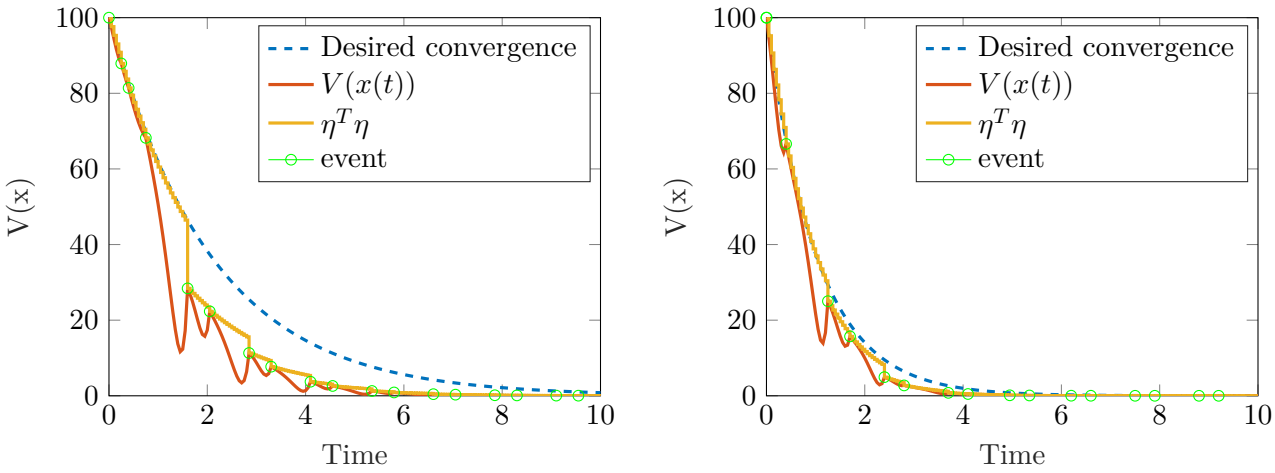
2-3-1 Higher decay rates

The first approach is based on the observation that the oscillations of the states are often correlated to the oscillations of the Lyapunov function. This, in turn, happens when the Lyapunov function is allowed to increase too much. In order to restrict that, we can impose a higher desired convergence rate. Fig. 2-6(a) was generated for almost an identical case as in the numerical example from the previous section, but the convergence rate was increased to $\lambda = 0.48$. This value was only slightly smaller than $\lambda_0 = 0.4836$. When compared to Fig. 2-3(d) with $\lambda = 0.3$, the oscillations were of smaller magnitude.

Going one step further, one can find another Lyapunov function for the same system, that has a greater decay rate λ_0. An example of a steeper Lyapunov function for system (2-18) is

$$P = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}, \quad Q = P.$$

In this case, $\lambda_0 = \lambda_m(P^{-1}Q) = 1$. This value allows to impose a higher convergence rate for the same system. The result for $\lambda = 0.98$ can be seen in Fig. 2-6(b).



(a) Original Lyapunov function - $\lambda = 0.48$

(b) A steeper Lyapunov function - $\lambda = 0.98$

Figure 2-6: Evolution of two different Lyapunov functions for the same system under the relaxed PETC with $\Delta = 0.05$ and $N_{max} = 20$.

For a greater decay rate the Lyapunov function experiences smaller oscillations. This translates to a smaller magnitude of system trajectories, as shown in Fig. 2-7. What is important is that the number of communication instants did not increase for a bigger desired convergence. It actually slightly decreased from 18 to 16. The results suggest that it is particularly beneficial to choose steeper Lyapunov functions and λ close to λ_0. Then one gets not only a

better guaranteed convergence, but also less oscillatory system trajectories while maintaining a low number of transmissions.

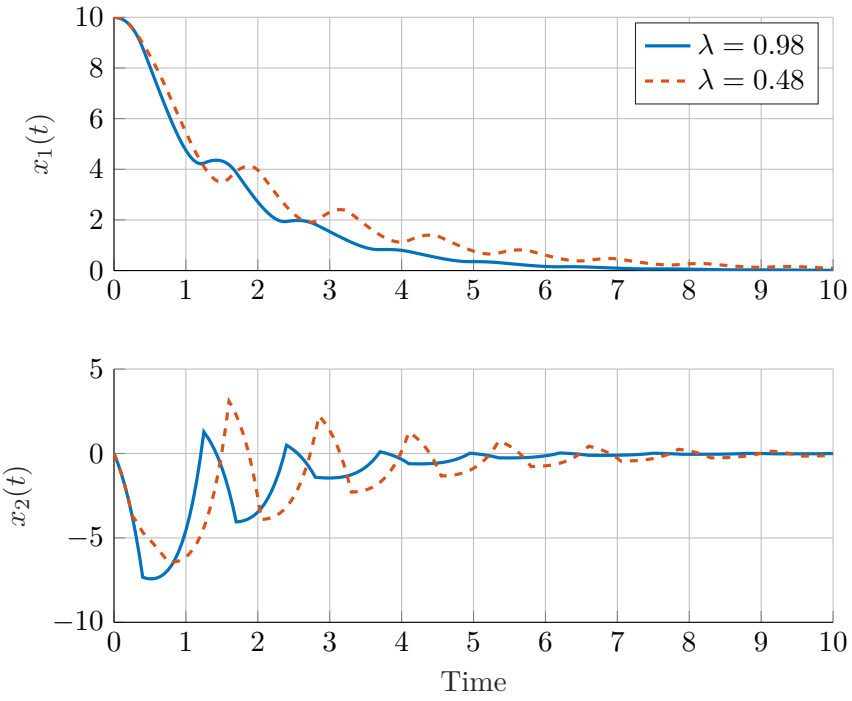


Figure 2-7: System's trajectories under the relaxed PETC with different Lyapunov functions and convergence rates.

2-3-2 Performance check

The difficult part in assessing the level of oscillations is that there are no performance measures that would take them into consideration. Time domain parameters like the convergence rate, settling time, overshoot or steady state only describe the sketch of the system trajectories. If we want to avoid excessive oscillations, we first have to define what do we mean by 'excessive'.

One way to do that could be to only allow the states to deviate from a reference trajectory to some extent and formulate it as an additional triggering condition. A reasonable choice for the reference are state trajectories of the plant with a periodic controller $x_{i,ref}$. Although they might not always be free from oscillations, comparing to them is a good measure of how inferior is PETC from the time-triggered implementation in terms of added oscillations. The region for allowable oscillations should also shrink with the magnitude of the reference trajectory. Similarly to the convergence rate, the bounds on the oscillations can decrease exponentially. If any of the states x_i under the relaxed PETC exceeds the upper or lower bound, the intelligent sensory system should trigger, what can be described with the following inequality:

$$|x_i(t_k) - x_{i,ref}(t_k)| > \beta e^{-\lambda t_k}$$

where $\beta \in \mathbb{R}^+$ is an additional parameter that indicates the size of the allowable region at time instant $t = 0$. The additional triggering condition also assumes the same convergence rate

λ for the allowable region as for the Lyapunov function. Similarly to the original triggering condition from the relaxed PETC, instead of checking if the states are within the allowable region at the current sampling time, we could check one time step ahead. This way the sensors would trigger before the states escape the region. Checking the condition for all the states $i \in \{1, ..., n\}$, can then be compactly written with:

$$||Ax(t_k) + BKx(t_i) - (A + BK)x_{ref}(t_k)||_\infty > \beta e^{-\lambda(t_k+\Delta)}.$$

The additional triggering condition means that the reference states x_{ref} have to be kept in memory. Moreover, an internal clock has to be included in order to keep track of the bounds $\beta e^{-\lambda t_k}$. When this is done it is possible to transform the additional triggering condition into a quadratic one by using the fact that:

$$|v| \geq ||v||_\infty$$

for any vector $v \in \mathbb{R}^n$. The additional triggering condition with the Euclidean norm

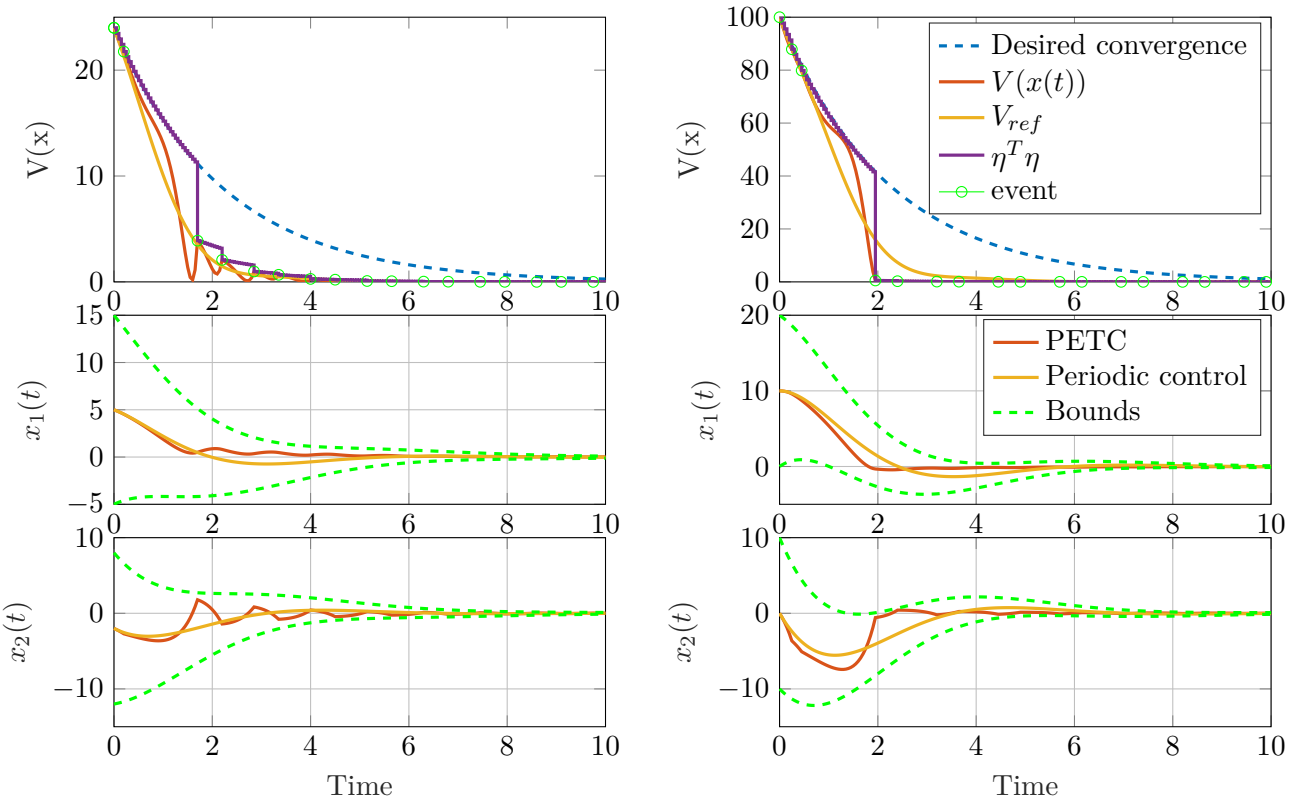
$$|Ax(t_k) + BKx(t_i) - (A + BK)x_{ref}(t_k)| > \beta e^{-\lambda(t_k+\Delta)}. \tag{2-19}$$

represents the quadratic form similarly to (2-4).

Fig. 2-8 shows the Lyapunov function and state trajectories for the system (2-18) under the relaxed PETC with the additional triggering condition (2-19) with $\beta = 10$. It can be seen that depending on the initial conditions ($x_{0,1} = \begin{bmatrix}10 & 0\end{bmatrix}^T$, $x_{0,2} = \begin{bmatrix}5 & -2\end{bmatrix}^T$) the oscillations are reduced to a different level. In both cases the number of transmissions remained small (17). Decreasing parameter β to 6 reduces the oscillations for the case shown in Fig. 2-8(a) without additional communication instants. However, for the other case (Fig. 2-8(b)) it increases the number of triggering times to 54. Therefore, there has to be a trade-off made when choosing β such that for any initial conditions the oscillations are reduced in comparison to the original relaxed PETC without increasing drastically the number of transmissions. Table 2-2 summarizes the number of communication instants for different parameters of the triggering condition tested on 10 different initial conditions. The higher the desired decay rate is, the bigger standard deviation, what implies that the number of transmissions is highly dependent on the initial conditions. The result of this experiment suggests that for smaller values of β (allowing only small oscillations) choosing a milder value of convergence rate will result in a more consistent number of transmissions regardless of the initial conditions.

Table 2-2: Average (μ) and standard deviation (σ) of number of communications for different values of performance parameter β and decay rates

	$\lambda = 0.2$	$\lambda = 0.25$	$\lambda = 0.3$	$\lambda = 0.35$	$\lambda = 0.4$	$\lambda = 0.45$
$\beta = 3$	$\sigma = 12.76$ $\mu = 28.6$	$\sigma = 16.82$ $\mu = 35.1$	$\sigma = 21.94$ $\mu = 41.4$	$\sigma = 28.96$ $\mu = 45.5$	$\sigma = 37.43$ $\mu = 73.6$	$\sigma = 46.50$ $\mu = 28.6$
$\beta = 6$	$\sigma = 0.92$ $\mu = 16.8$	$\sigma = 8.80$ $\mu = 19.1$	$\sigma = 11.93$ $\mu = 22.4$	$\sigma = 16.97$ $\mu = 30.6$	$\sigma = 22.30$ $\mu = 41.2$	$\sigma = 30.26$ $\mu = 28.6$
$\beta = 9$	$\sigma = 0.82$ $\mu = 16.7$	$\sigma = 2.50$ $\mu = 17.3$	$\sigma = 6.46$ $\mu = 18.7$	$\sigma = 12.37$ $\mu = 23.9$	$\sigma = 14.94$ $\mu = 28.0$	$\sigma = 17.77$ $\mu = 28.6$
$\beta = 12$	$\sigma = 0.82$ $\mu = 16.7$	$\sigma = 0.79$ $\mu = 16.8$	$\sigma = 0.67$ $\mu = 17.0$	$\sigma = 8.49$ $\mu = 19.9$	$\sigma = 13.19$ $\mu = 24.6$	$\sigma = 14.59$ $\mu = 28.6$



(a) Initial condition $x_{0,2}$

(b) Initial condition $x_{0,1}$

Figure 2-8: Evolution of the Lyapunov function and state trajectories of the system (2-18) under the relaxed PETC with $\Delta = 0.05$, $N_{max} = 40$, $\lambda = 0.45$ and with the additional triggering condition (2-19) with $\beta = 10$.

Bounds on system trajectories that converge exactly to the equilibrium might be dangerous when disturbances come into play. Not resetting the bounds and keeping them very tight will result in degrading PETC to almost periodic control. It is then reasonable to allow convergence of the states to a ball instead one value. The size of the ball should be chosen such that it applies to both - PETC and periodic control. The radius that satisfies this requirement is $\gamma||\delta||_\infty$, where γ is the $\mathcal{L}_\infty$ gain from disturbance to the output of the system with the periodic controller. This value informs how much the input can be attenuated or magnified by the system and thus how it affects the output. In order to find it, the closed loop system in continuous time

$$\frac{d}{dt}x = \underbrace{(A^p + B^p K)}_{A_{cl}} x + E\delta \tag{2-20}$$

has to be discretized with Δ according to equation (2-6). Moreover, an output $y = Cx$ of the system has to be chosen. If we care about all the states, we can set $C = I \in \mathbb{R}^{(n_x \times n_x)}$. If $n_x > 1$ the system is MIMO and to find its $\mathcal{L}_\infty$ gain we have to look at the peak value of the largest singular value of the system across all frequencies. For SISO systems it only requires finding the peak value of the magnitude Bode plot. Once this value is known we can rewrite

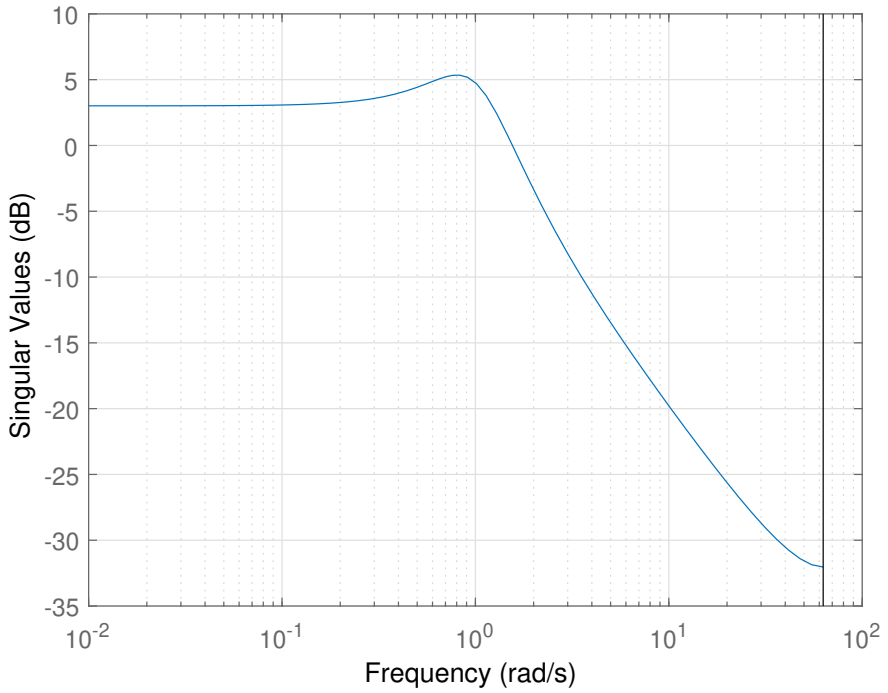


Figure 2-9: Non-zero singular value of the discretized system (2-18) with $y = x$.

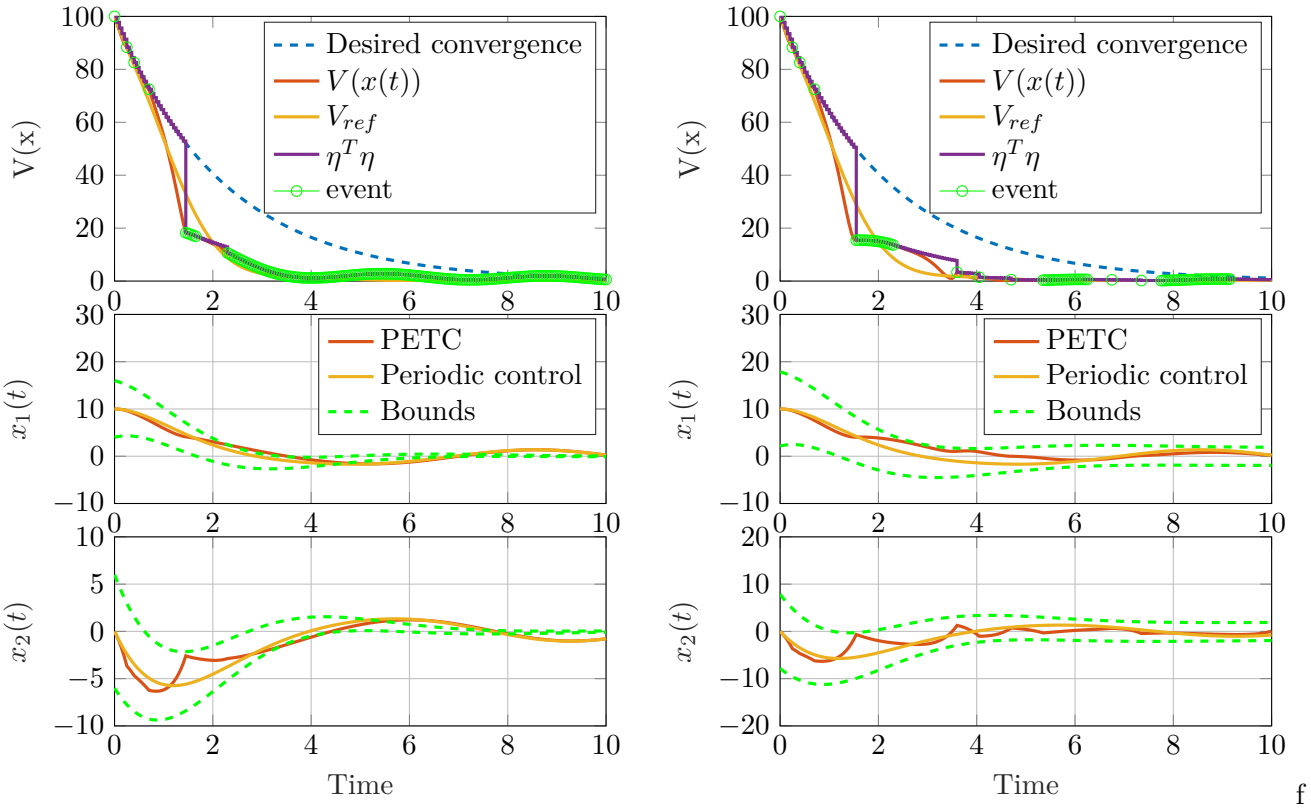
the additional triggering condition (2-19) as

$$|Ax(t_k) + BKx(t_i) - (A + BK)x_{ref}(t_k)| > \beta e^{-\lambda(t_k+\Delta)} + \gamma||\delta||_\infty. \tag{2-21}$$

For our 2D system (2-18) after discretization, if we choose $C = I$ we end up with two outputs. However, there is only one non-zero singular value because the disturbance input is one-dimensional. Its value across all frequencies is shown in Fig. 2-9. The peak value is equal to 5.34 dB at frequency 0.83 rad/s. After converting from decibels we obtain $\gamma = 10^{\frac{5.34}{20}} \approx 1.85$.

Results of experiments comparing the additional triggering condition with and without using the $\mathcal{L}_\infty$ gain γ are illustrated in Fig. 2-10. Both sub-figures were generated for the same initial condition under the same disturbances. Fig. 2-10(a) used the oscillation bounds that converged to zero, while for Fig. 2-10(b) they converged to a ball of radius $\gamma = 1.85$. As a result the first approach experienced 164 transmissions, because from time $t = 2.1$ it already degraded to periodic control. On the other hand, convergence to a ball allowed to keep the number of transmissions at 73 and the control remained aperiodic thorough the whole simulation.

The oscillations in both cases are smaller compared to the original relaxed PETC. It is important to notice that adding a constant to the additional triggering condition enlarges the region allowable for oscillations. Therefore one may consider reducing the value of the parameter β in that case, such that the oscillations are kept small.



(a) Convergence to zero (2-19).

(b) Convergence to a ball of radius $\gamma = 1.85$ (2-21)

Figure 2-10: Evolution of the Lyapunov function and state trajectories of the system (2-18) under the relaxed PETC with $x_0 = [10\ 0]^T$, $\Delta = 0.05$, $N_{max} = 40$, $\lambda = 0.45$ and with the additional triggering condition (2-19) with $\beta = 6$ in the presence of disturbances $\delta = \sin(t)$.

Chapter 3

Predicting future events

The triggering condition introduced in the previous chapter postpones the occurrence of events compared to other PETC strategies, effectively resulting in a smaller number of triggers. The question that arises is if there is anything more that can be done to reduce the communication between the controller and the sensor. One solution would be to ignore some of the events and trigger only for the remaining ones. However, one has to answer one fundamental question: is it possible while still maintaining stability of the system?

First and foremost, when we decide to ignore an event we have to make sure that there will be more of them in the future. This is not trivial for the case of PETC, since it does not make any predictions like e.g. STC. It decides whether to trigger or not based on the current measurement by checking a simple condition without extensive computation. Therefore, predicting future events in an efficient way is a problem that needs a solution and will be the subject of this chapter.

A less obvious issue that has to be taken into consideration while answering the same question is the value of the Lyapunov function at future events. In order to ensure convergence of the Lyapunov function its value should be decreasing from one triggering instant to the next one when no disturbances are present. Fortunately, since the relaxed triggering condition detects crossings of the Lyapunov function with an exponentially decaying bound, we can be sure that at any event the value is smaller than at the previous one. Therefore, if we can guarantee that there will be more events in the future, when the control action is not updated at the current event, the triggering condition ensures that also the value of the Lyapunov function will be smaller. This is illustrated on Fig. 3-1.

It is important to mention that the above property cannot be guaranteed for all PETC implementations. If the triggering condition detects e.g. the moment when the Lyapunov function starts increasing, we have no guarantees that (if the control action is not updated) its value will be smaller at the next triggering instant. Therefore, this chapter focuses on predicting future events for the relaxed triggering condition for PETC introduced in the previous chapter. The main result uses ideas that were originally applied to scheduling of NCSs and thus a short introduction will be given first.

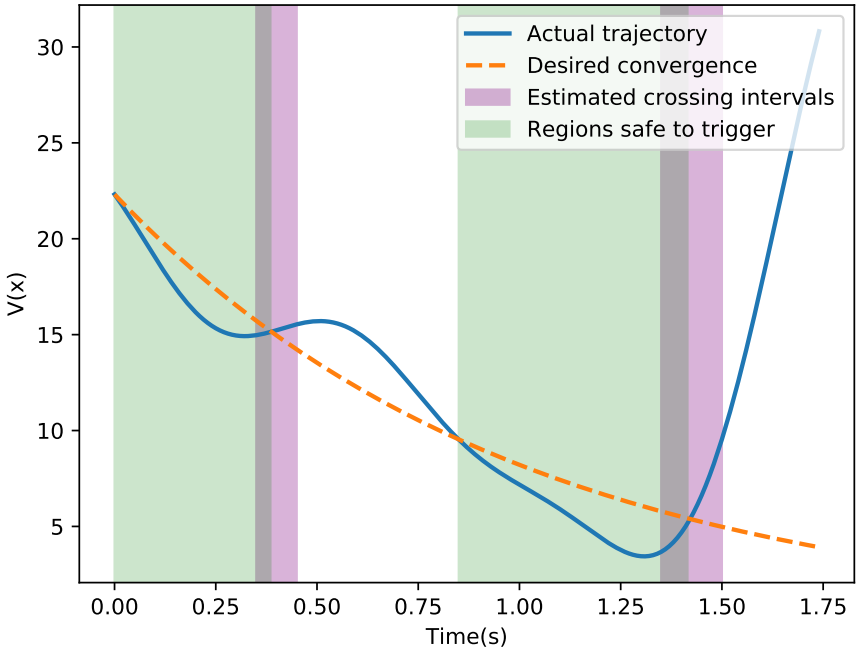


Figure 3-1: A sample trajectory of a Lyapunov function for a 3D system.

3-1 State dependent sampling

An important observation that can make predicting the next triggering instant easier is that, for LTI systems, a state $x \neq 0$ and λx, $\lambda \in \mathbb{R}\backslash\{0\}$, have the same triggering time. A quadratic triggering condition is violated at the same time, since $x^T Q x > 0 \iff \lambda^2 x^T Q x > 0$. This means that states lying on the same ray going through the origin have the same sequence of triggering times. In [3], this observation is used to partition the state space into regions by grouping together the neighbouring rays and using the spherical coordinates of $x \in \mathbb{R}^n$: $(r, \theta_1, ..., \theta_{n-1})$. In addition, it is only necessary to partition half of the space because x and $-x$ have the same triggering time. Therefore $\theta_1, \theta_2, ..., \theta_{n-1} \in [0, \pi]$. Each region is associated with some range of the angular coordinates θ_i: $x \in \mathcal{R}_s \Leftrightarrow \forall i \in \{1, ..., n-1\}, \theta_i \in [\theta_{i,s}^-, \theta_{i,s}^+]$. An example of covering a 2-dimensional system into conic regions is shown in Fig. 3-2.

Once the regions are created one can find a lower bound on the triggering time for all the states inside one region. Associating a triggering time τ_s with a region $\mathcal{R}_s$, $s \in \{1, 2, ..., q\}$ can be done online, what reduces the number of computations online. When controlling the plant, every time the controller gets a new sample, it has to locate it in one of the predefined regions and the triggering time is known immediately. Assigning the current state to one of the regions is not expensive since it only requires computing $n-1$ angles from the sample.

Apart from partitioning the state space, the process of finding the lower bound on the triggering time requires also convex embedding according to time. Since the system is deterministic and if there are no disturbances we can express the value of the triggering condition at some time $\sigma \in [t_i, t_{i+1}]$ in the future with $x(t_i)^T \Phi(\sigma) x(t_i)$, where $\Phi(\sigma)$ depends on the system ma-

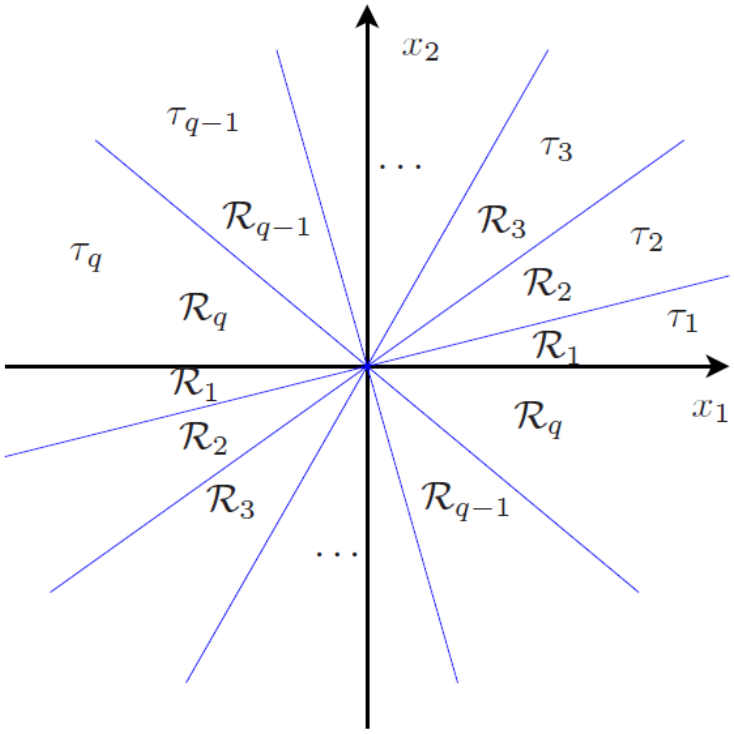


Figure 3-2: Conic partitioning the 2D state space into q regions [3].

trices and the quadratic triggering condition matrix. To make sure that $x(t_i)^T\Phi(\sigma)x(t_i) \leq 0$, $\forall\sigma \in [t_i, t_{i+1}]$, we would have to check an infinite number of inequalities. Therefore it is necessary to construct convex polytopes around the matrix $\Phi(\sigma)$ to obtain a finite set of inequalities using vertices $\Phi_{\kappa,s}$, $\kappa \in \mathcal{K}_s$ (a finite set of indices). One example of how to construct these matrices using Taylor series expansion is given in [3]. With that, one can find the lower bound on the triggering time $\underline{\tau}_s$ for every region $\mathcal{R}_s$ solving a finite number of LMIs. A similar idea was used in [26], where also a method of finding the upper bound on the triggering time $\bar{\tau}_s$ is given.

The described procedure could be used to find more than one time interval in the future for each region when the consecutive events are expected to occur if the control action is not updated. However, this approach has one possible drawback - the partition of the state space does not take into account the behaviour of the system. Usually one imposes an isotropic conic covering where every angular coordinate θ_i is divided into m equidistant intervals. Such partitioning does not necessarily coincide with an 'optimal' method of dividing the state space. For instance, in a theoretical case, for a cone $\mathcal{R}_s$ in 2D with $\theta \in [15°, 20°]$, one could find a time interval of $[0.1, 0.3]$ seconds, while decreasing the size of cone to $\theta \in [15°, 19°]$ would result in $\tau_4 \in [0.1, 0.15]$ seconds. We can see that a very small change in the partitioning could significantly reduce the time interval associated to that region. One could divide the state space into very small cones but then the number of regions grows very fast on top of the exponential increase with the dimension of the system. Hence, there is a trade-off between the number of regions and the accuracy of the time interval approximation.

A more natural way of dividing the state space uses the discrete-time model behaviour also

presented in [3]. In this approach every region is associated with the value of the triggering condition at a given sampling time t_k. Because the triggering condition is quadratic, $x(t_i)^T Q_k x(t_i) \leq 0$, where Q_k is equivalent to $\Phi(\sigma)$ but in discrete time instances (it uses discretized system matrices), also forms a cone but of an unknown shape. The advantage of this approach is that the number of regions does not depend on the dimension of the system and the choice of the sampling time is dependant on the precision of time intervals that one wants to obtain. For the case of PETC it is natural to choose the same sampling time as Δ. Moreover, it could be used without any modifications to find more consecutive events by choosing a bigger prediction horizon.

Although this approach allows us to find more accurate time intervals and does not impose a shape on the cones, it has a drawback of requiring more computations online. In order to find the next triggering time for a given sample $x(t_i)$, one has to perform several $x(t_i)^T Q_k x(t_i)$ multiplications until the resulting value is greater than 0. The situation is even worse if one wants to find more triggering instants because it extends the prediction horizon. The online computational complexity of this approach is $O(qn^2)$ (where q is the number of regions), while for the previous approach (with spherical coordinates) it is only $O(n)$ ([3]). In the next section, we introduce an approach that aims at reducing the number of computations that have to be performed online for the partitioning according to the discrete-time model behaviour of the system.

3-2 Approaches to reduce computational effort

Let us start by formalizing matrix Q_k. To find the value of the triggering condition at some point $r_k = t_i + k\Delta$, $k \in \mathbb{N}_0$ based on the current sample $x(t_i)$, one has to find the value of the states at time r_k first. Since we are only interested in finding a value in a sequence of times $\{r_k\}$, we use discrete-time system matrices as given by (2-6). Then, we can compute:

$$\begin{aligned} x(r_k) =& R(k)x(t_i), \\ R(k) :=& A^k + \sum_{i=0}^{k-1} A^i B K, \end{aligned} \tag{3-1}$$

as given in e.g. [27]. The scheduler that will be responsible for granting access to the network has to have an internal clock and therefore we can make our triggering condition time-dependent. To find out how far the Lyapunov function is from the imposed bound at time r_k, the following multiplication has to be performed:

$$\psi(x(r_k), x(t_i)) \coloneqq \begin{bmatrix} x(r_k)^T & x(t_i)^T \end{bmatrix} \underbrace{\begin{bmatrix} P & 0 \\ 0 & -Pe^{-\lambda k \Delta} \end{bmatrix}}_{\bar{Q}(k)} \begin{bmatrix} x(r_k) \\ x(t_i) \end{bmatrix}. \tag{3-2}$$

Substituting (3-1) into (3-2) we arrive at the expression that depends solely on time instant r_k and available sample $x(t_i)$:

$$x(t_i)^T \underbrace{\begin{bmatrix} R(k)^T & I \end{bmatrix} \begin{bmatrix} P & 0 \\ 0 & -Pe^{-\lambda k \Delta} \end{bmatrix} \begin{bmatrix} R(k) \\ I \end{bmatrix}}_{\coloneqq Q_k} x(t_i). \tag{3-3}$$

The important observation is that matrices Q_k can be computed offline, since they only depend on the model of the system and the Lyapunov function. The number of these matrices depends on how far in the future one wants to search for events. A time horizon should be defined as $t_i + k_{max}\Delta$.

Now, to find the last event that will happen in a predefined time horizon, the scheduler has to perform online computations using available sample $x(t_i)$. Since we are looking for the last event, it is natural to perform multiplications starting from the last available matrix $Q(k_{max})$. An event in our case corresponds to a situation when

$$x(t_i)^T Q_{k-1} x(t_i) \leq 0 \quad \wedge \quad x(t_i)^T Q_k x(t_i) > 0. \tag{3-4}$$

This means that the plant has to get access to the network at time instant r_{k-1} at latest. However, it is sufficient to only look for Q_k such that the triggering condition is smaller than or equal to 0. Then we are sure that the Lyapunov function is below the bound and its value at time t_{i+1} will be smaller than at time t_i. The remaining of this chapter aims at finding a solution to the following problem:

Problem 3.1. *For a given $x(t_i)$, find*

$$k_{trig} = \max\{k \leq k_{max} | x(t_i)^T Q_k x(t_i) \leq 0, k \in \mathbb{N}\} \tag{3-5}$$

with Q_k defined as in (3-3), such that the computational complexity is reduced.

3-2-1 Sub-sampling

The primary goal of the scheduler is to predict if there will be more events if the control action is not updated. The exact times of the upcoming events do not have to be known by the scheduler. It is sufficient that the scheduler grants access to the network during a larger time interval, in which the event is estimated to happen. The PETC strategy that monitors the triggering condition at every sampling instant will trigger at the right time in the given interval. In other words the scheduler should only know a rough estimation of the triggering time and the intelligent sensory system is responsible for detecting the exact moment.

Therefore, the first thing that can be done to reduce the number of computations online is sub-sampling from the whole set of matrices Q_k. One can simply check the triggering condition every p-th sampling time, reducing the number of computations p-fold. With the new subsequence of matrices Q_k, Problem 3.1 still holds, but there has to be the following modification made to the definition of Q_k:

$$Q_k \coloneqq \begin{bmatrix} R(pk)^T & I \end{bmatrix} \begin{bmatrix} P & 0 \\ 0 & Pe^{-\lambda pk\Delta} \end{bmatrix} \begin{bmatrix} R(pk) \\ I \end{bmatrix}, \quad p, k \in \mathbb{N}. \tag{3-6}$$

By sub-sampling we introduce a trade-off between the computation time and accuracy of the event search. The bigger p is, the higher the chance that the scheduler will miss an event. This can happen especially when the frequency of the oscillations of the Lyapunov function is higher than $p\Delta$. It is then possible that at time instants $pk\Delta$ and $p(k+1)\Delta$ the value of the triggering condition is positive, but between them there exists a time interval where the value is smaller than zero.

However, as mentioned before, the scheduler's task is to roughly estimate the time at which the event will occur. The fact that it can possibly miss the event does not impede stability of the system, because the access to the network will be given earlier. Therefore, the disadvantages of sub-sampling are not very significant and in return we get a substantial cut on the computational complexity. It is therefore advisable to choose $p \geq 2$, $p \in \mathbb{N}$.

3-2-2 Projections

As discussed in section 3-1, locating the current sample in one of the regions is less expensive in terms of computations when using spherical coordinates. But since we do not want to impose a structure on the partition, we have to evaluate the shape of regions $x^T Q_k x \leq 0$, that are created naturally by the discrete-time behaviour of the system. Finding shapes of such regions is not a trivial task because generally they are not polytopic and very often not even convex. One thing we can do to overapproximate the shape of a cone is to project it on $n-1$ planes, where n is the dimension of the system. As a result, we get a range for every spherical coordinate of $x(t_i)$ that it has to belong to in order for $x(t_i)$ to satisfy $x(t_i)^T Q_k x(t_i) \leq 0$. Since the angular ranges are overapproximations, this condition is necessary but not sufficient. Nevertheless, it has the potential to reduce the number of computations online.

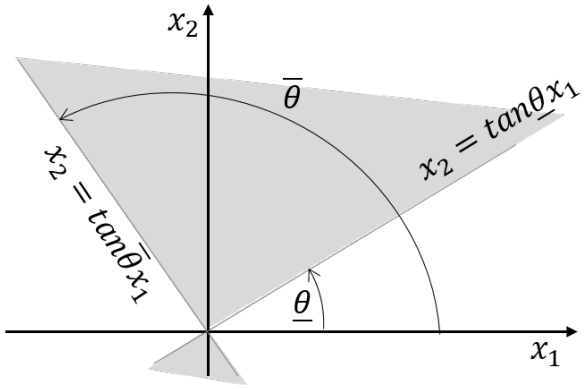


Figure 3-3: A 2D projection of an n-D cone.

A 2D projection of a cone pointing at the origin in n-D is also a cone and it can be described with a quadratic form. As shown if Fig. 3-3, it is the region between two straight lines going through the origin. These rays can be described by equation: $x_2 = \tan\theta x_1$. We are interested in finding the projection only on $[0, \pi]$ since the projection on $[\pi, 2\pi]$ is identical. Because cosine changes its sign at $\pi/2$, we have to consider two regions: $[0, \pi/2]$ and $[\pi/2, \pi]$ in order to derive inequalities describing the projection cone. Moreover, let us denote $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$.

1. $\theta \in [0, \pi/2]$

   If $\underline{\theta}$ belongs to this region it will act as a lower bound when $x_2 \geq \tan\underline{\theta} x_1$:

$$\begin{aligned} x_2 &\geq \frac{\sin\underline{\theta}}{\cos\underline{\theta}} x_1 \\ \cos\underline{\theta} x_2 &\geq \sin\underline{\theta} x_1 \\ \underbrace{\begin{bmatrix} -\sin\underline{\theta} & \cos\underline{\theta} \end{bmatrix}}_{:=a_1} \mathbf{x} &\geq 0 \end{aligned}$$

   If $\bar{\theta}$ belongs to this region it will act as an upper bound when $x_2 \leq \tan\bar{\theta} x_1$:

$$\begin{aligned} x_2 &\leq \frac{\sin\bar{\theta}}{\cos\bar{\theta}} x_1 \\ \cos\bar{\theta} x_2 &\leq \sin\bar{\theta} x_1 \\ \underbrace{\begin{bmatrix} \sin\bar{\theta} & -\cos\bar{\theta} \end{bmatrix}}_{:=a_2} \mathbf{x} &\geq 0 \end{aligned}$$

2. $\theta \in [\pi/2, \pi]$

   If $\underline{\theta}$ belongs to this region it will act as a lower bound when $x_2 \leq \tan\underline{\theta} x_1$:

$$\begin{aligned} x_2 &\leq \frac{\sin\underline{\theta}}{\cos\underline{\theta}} x_1 \\ \cos\underline{\theta} x_2 &\geq \sin\underline{\theta} x_1 \\ a_1\mathbf{x} &\geq 0 \end{aligned}$$

   If $\bar{\theta}$ belongs to this region it will act as an upper bound when $x_2 \geq \tan\bar{\theta} x_1$:

$$\begin{aligned} x_2 &\geq \frac{\sin\bar{\theta}}{\cos\bar{\theta}} x_1 \\ \cos\bar{\theta} x_2 &\leq \sin\bar{\theta} x_1 \\ a_2\mathbf{x} &\geq 0 \end{aligned}$$

Despite the change in the sign of cosine at $\pi/2$, we have the same inequalities for both angular intervals that describe an intersection between two half-planes that forms a cone:

$$a_1\mathbf{x} \geq 0 \quad \wedge \quad a_2\mathbf{x} \geq 0.$$

To get a quadratic symmetrical form representing the cone, one has to follow a few steps:

$$\begin{aligned} \mathbf{x}^T a_1^T \geq 0 \quad \wedge \quad a_2\mathbf{x} \geq 0 \quad &\Longrightarrow \quad \mathbf{x}^T a_1^T a_2 \mathbf{x} \geq 0 \\ \mathbf{x}^T a_1^T a_2 \mathbf{x} &= \mathbf{x}^T \frac{a_1^T a_2 + a_2^T a_1}{2} \mathbf{x} \\ \mathbf{x}^T \left(\frac{a_1^T a_2 + a_2^T a_1}{2} \right) \mathbf{x} &\geq 0 \\ \mathbf{x}^T \underbrace{(a_1^T a_2 + a_2^T a_1)}_{:=Q_\pi} \mathbf{x} &\geq 0 \end{aligned}$$

After multiplying and adding vectors a_1 and a_2, and using the fact that $\sin(\alpha + \beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$, one obtains the following formula:

$$Q_\pi(\underline{\theta}, \bar{\theta}) = \begin{bmatrix} -2\sin\underline{\theta}\sin\bar{\theta} & \sin(\underline{\theta} + \bar{\theta}) \\ \sin(\underline{\theta} + \bar{\theta}) & -2\cos\underline{\theta}\cos\bar{\theta} \end{bmatrix} \tag{3-7}$$

Following [28], for a point $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ we define a projection on its $i - j$ coordinates as $(x)_{(i,j)} \coloneqq (x_i, x_j) = P_i\mathbf{x}$. P_i is a projection matrix of size $2 \times n$, that contains almost all 0, except for entries $(1, i)$ and $(2, j)$ which are equal to 1. Angular ranges $[\underline{\theta}_i, \bar{\theta}_i]$ for every projection on plane $i \in \{1, 2, ..., n-1\}$ have to satisfy:

$$x^T Q_k x \leq 0 \quad \Longrightarrow \quad (x)_{(i,i+1)}^T Q_\pi(\underline{\theta}_i, \bar{\theta}_i)(x)_{(i,i+1)} \geq 0. \tag{3-8}$$

In order to find $\underline{\theta}_i^* := \max \underline{\theta}_i$ and $\bar{\theta}_i^* := \min \bar{\theta}_i$ that satisfy (3-8) we will use the S-procedure [29]. For every i we initialize the search by setting $\underline{\theta}_i = 0$ and $\bar{\theta}_i = \pi$. Implication (3-8) is then always true. We first fix the upper bound $\bar{\theta}_i$ and increase the lower bound $\underline{\theta}_i$ by a constant step $d\theta$. The feasibility problem that has to be solved offline using a semi-definite programming (SDP) solver is:

$$\text{if} \quad \exists \tau \in \mathbb{R}_0^+ \quad P_i^T Q_\pi((\underline{\theta}_i, \bar{\theta}_i) P_i + \tau Q_k \geq 0. \tag{3-9}$$

We increase the lower bound until problem (3-9) is not feasible any more. The last value of $\underline{\theta}_i$ that yields a feasible solution is $\underline{\theta}_i^*$. We then fix the lower bound at $\underline{\theta}_i^*$ and repeat the same process, this time decreasing the value of $\bar{\theta}_i$ by step $d\theta$. The iterations continue until the problem is not feasible and the value of $\bar{\theta}_i^*$ is found.

Remark 3.1. *When $\underline{\theta}_i \neq 0$ or $\bar{\theta}_i \neq 0$, $P_i^T Q_\pi P_i$ from (3-9) has rank 2 and one positive eigenvalue (it is non-definite). Therefore, for $P_i^T Q_\pi P_i \geq -\tau Q_k$, $-Q_k$ must have at most 1 positive eigenvalue. Thus, Q_k can have at most 1 negative eigenvalue. If it has more negative eigenvalues, problem (3-9) is only feasible in the first iteration and the resulting angular range will be $[0, \pi]$.*

As a result, for every Q_k there is an associated set of angular ranges $[\underline{\theta}_{i,k}^*, \bar{\theta}_{i,k}^*]$, $i \in \{1, 2, ..., n-1\}$, that can be used for more easy locating the current sample $x(t_i)$ in one of the regions $x^T Q_k x \leq 0$. The first thing that has to be computed online after getting a new sample is the set of current angles θ_i. This is done with a short formula:

$$\theta_i = \begin{cases} \arccos\left(\frac{x_i}{r}\right) & \text{if } x_{i+1} \geq 0 \\ \pi - \arccos\left(\frac{x_i}{r}\right) & \text{if } x_{i+1} < 0 \end{cases}, \quad r = |(x)_{(i,i+1)}|, \quad i \in \{1, 2, ..., n-1\}.$$

The dependency on the sign of x_{i+1} is due to the fact that (x) is not a one-to-one relation and there are multiple possible values for one argument x. In the Python implementation of the above formula function `math.atan2` is used instead:

$$\theta_i = \begin{cases} \texttt{math.atan2}(x_{i+1}, x_i) & \text{if } x_{i+1} \geq 0 \\ \texttt{math.atan2}(x_{i+1}, x_i) + \pi & \text{if } x_{i+1} < 0 \end{cases}, \quad i \in \{1, 2, ..., n-1\}. \tag{3-10}$$

Function `math.atan2(y,x)` computes the angle formed by a vector from the origin to point (x, y) with the positive axis X. The output is between $-\pi$ and π and therefore if $y < 0$ we add π to the result in order to get a value between $[0, \pi]$. This implementation reduces significantly the time that is needed to perform the angular coordinates computation.

Once the scheduler has the current angles computed, it can use them to speed up the search of the last occurring event in the search horizon. If there is at least one angle θ_i that does not belong to earlier computed range $[\underline{\theta}_{i,k}^*, \bar{\theta}_{i,k}^*]$, then multiplication $x(t_i)^T Q_k x(t_i)$ can be skipped. If all the current angles θ_i belong to the ranges, then the scheduler has to check if $x(t_i)^T Q_k x(t_i) \leq 0$.

Overapproximating the regions with projections may not always reduce the number of computations drastically. The reason behind that is the unusual shape of the cones $x^T Q_k x \leq 0$. They can be non-convex, very flat but broad, shaped like a curved disc, etc. If they are big, then the projection is usually the whole projection plane. Angular ranges $[\underline{\theta}_{i,k}^*, \bar{\theta}_{i,k}^*]$ smaller

than $[0, \pi]$ can only be found for smaller cones $x^T Q_k x \leq 0$, what goes along with Remark 3.1. The cones, in turn, get smaller when the triggering condition is checked for further time instants in the future. This is easily explained - the longer the time in which the control action is not updated, the more likely that the Lyapunov function will exceed the bound. Therefore, projecting is particularly beneficial if the prediction horizon is longer.

3-2-3 Predecessors

Because all the matrices Q_k are computed for the same system and they describe its behaviour in the future time instances, there exist fundamentals to believe that the matrices are related to each other. Knowing if $x(t_i)^T Q_j x(t_i) > 0$ implies $x(t_i)^T Q_i x(t_i) > 0$, $i < j$, would be particularly useful in our backward search. Once the scheduler performs multiplication $x(t_i)^T Q_j x(t_i)$ and gets a positive value, then it knows that also $x(t_i)^T Q_i x(t_i) > 0$. Therefore, the scheduler can skip the quadratic multiplication with matrix Q_i, because we are only looking for the multiplications where the outcome is negative. This could reduce the number of quadratic multiplications that the scheduler has to perform. All the matrices Q_i that satisfy the above explained relation we call 'predecessors' and the formal definition can be found below.

Definition 3-2.1 (Predecessors)**.** *Matrix Q_i is called a predecessor of matrix Q_j if and only if*

$$x^T Q_j x \geq 0 \quad \Longrightarrow \quad x^T Q_i x \geq 0, \quad i < j, \quad i, j \in \mathbb{N} \tag{3-11}$$

for all $x \in \mathbb{R}^n$.

Each matrix Q_k can have several predecessors, that can be computed offline using the S-procedure again. For every matrix Q_j, $j \in \{1, 2, ..., k_{max}\}$ and Q_i, $i < j$, $i \in \mathbb{N}$, we have to solve the following feasibility problem:

$$\text{if} \quad \exists \tau \in \mathbb{R}_0^+ \quad Q_i \geq \tau Q_j. \tag{3-12}$$

If there exists a feasible solution, then Q_i is a predecessor of Q_j. Solving this problem offline using a semi-definite programming (SDP) solver allows us to associate a set of indexes i to every matrix Q_j, $j \in \{1, 2, ..., k_{max}\}$, such that Q_i is a predecessor of Q_j. This knowledge can be then used to reduce the number of computations online.

From the experimental results it can be noticed that all the predecessors i that appear in the computed sets, correspond to matrices Q_i for which all of the angular ranges of the projections are empty, which means $\underline{\theta}^*_{i,k} = \bar{\theta}^*_{i,k}$ (see Appendix B-1 and B-2). If none of the states satisfy $x^T Q_i x \leq 0$ as the projections suggest, then $\forall x \in \mathbb{R}^n \; x^T Q_i x \geq 0$. Then implication (3-12) is always true. Because predecessors seem to coincide with matrices for which the projections are empty, they are unlikely to reduce the computations, when used together with projections. Moreover, it is possible that using predecessors can enlarge the computation time because some unnecessary operations have to be performed. In other words the scheduler would have to perform a check of the list of predecessors at every iteration, while the same information could be obtained from checking the projections only. Therefore, the final algorithm will not use predecessors.

3-3 The algorithm

In order to maximize the benefits coming from the optimization methods presented in the previous section, the final algorithm (Algorithm 1), that solves Problem 3.1, uses 2 out of 3 previously described methods: sub-sampling and projections.

3-3-1 Offline computations

Before the scheduler can perform any computations, the user has to provide the model of the system (matrices A^p, B^p and K), the Lyapunov function P and its desired decay rate λ. There are also three additional parameters that have to be chosen: discretization (sampling) time Δ, prediction horizon k_{max} and the sub-sampling factor p. With this information the scheduler can compute offline the following data structures:

- matrices Q_k according to (3-6), that are already a sub-sampled sequence,
- $n-1$ angular ranges $[\underline{\theta}^*_{i,k}, \bar{\theta}^*_{i,k}]$ for every matrix Q_k by solving another SDP problem (3-9),
- list *useful* of size k_{max} that has all elements *True* except at indexes j that correspond to matrices Q_j that have all the projection ranges equal to $[0, \pi]$. By checking *useful*$[k]$ the scheduler knows that there is no point in checking if all the angles θ_i belong to associated ranges $[\underline{\theta}^*_{i,k}, \bar{\theta}^*_{i,k}]$, because they do. Instead of performing the check $n-1$ times for every θ_i, it can just check the value of *useful*$[k]$.

3-3-2 Online computations

The search for the maximum k where $x(t_i)^T Q_k x(t_i) \leq 0$ starts from the last precomputed matrix Q_j, so once the scheduler gets a new measurement $x(t_i)$, it sets $j = k_{max}$. It first checks the value of *useful*$[j]$ to see if computing angles can be useful. If the value is *False*, then it can proceed to the quadratic multiplication straight away, because the projection for matrix Q_j is not useful. If it is *True*, the scheduler computes the angles θ_i using formula (3-10). Then, it checks if the angles θ_i belong to projected regions $[\underline{\theta}^*_{i,j}, \bar{\theta}^*_{i,j}]$ associated with matrix Q_j. If they do not belong there then we are sure that $x(t_i)^T Q_j x(t_i) > 0$ and the quadratic vector-matrix multiplication can be skipped and we can proceed to matrix Q_{j-1}.

However, if all the angles θ_i belong to their regions (or the value of *useful*$[j]$ is *False*), the scheduler has to perform more computations with matrix Q_j. It is because the membership to projected regions is a necessary but not a sufficient condition. The scheduler has to perform a quadratic multiplication $x(t_i)^T Q_j x(t_i)$. If $x(t_i)^T Q_j x(t_i) \leq 0$, the scheduler can stop the search because $k_{trig} = j$. However, if the result is greater than 0, the scheduler has to continue the algorithm because j is not the value that we are looking for. The whole procedure can be repeated, this time for matrix Q_{j-1}. This loop continues, decreasing the index of matrix Q_j by one at every iteration, until k_{trig} is found. The corresponding time, when the access to the network should be granted is $t_{trig} = k_{trig} p \Delta$.

The computational complexity of finding k_{trig} with algorithm 1 is reduced compared to the original idea of checking every matrix Q_k as described in section 3-2. First of all it decreased

Algorithm 1 Finding k_{trig}

$j \leftarrow k_{max}$
$k_{trig} \leftarrow 0$
$x_0 \leftarrow$ new sample
compute θ_i according to (3-10)
while $j \geq 1$ **do**
    $belongs \leftarrow True$
    if $useful[j]$ **then**:
        $i \leftarrow 1$
        for $i \geq n-1$ **do**
            if not $\theta_i \in [\underline{\theta}^*_{i,j}, \bar{\theta}^*_{i,j}]$ **then**
                $belongs \leftarrow False$
                break
    if $belongs$ **then**
        if $x_0^T Q_j x_0 \leq 0$ **then**
            $k_{trig} \leftarrow j$
            break
        $j \leftarrow j-1$
return k_{trig}

from $O(qpn^2)$ to $O(qn^2)$ due to sub-sampling. Depending on the value of p this can have a substantial effect on the computations. The use of projections in the algorithm does not reduce the complexity of the worst case scenario, because it may happen that all the projections will contain the whole projection plane. However, in reality it happens very rarely and projections often allow to skip several quadratic multiplications, what is especially beneficial in higher order systems. The effect of using projections on computation time are presented in the next section.

3-4 Numerical example

Since the algorithm described in the previous section is used for detecting last event in a given time horizon, a system that experiences multiple crossings of the Lyapunov function with its bound is chosen for the numerical example. Such systems are characterized with an oscillatory behaviour which corresponds to having complex poles. To see how the algorithm performs for different cases, two distinct systems are chosen for the experiment - a 2D and a 5D system.

The 2D system in the form of (1-1) has the following matrices:

$$A^p = \begin{bmatrix} -11.3283 & 12.5283 \\ -12.3283 & 11.4283 \end{bmatrix}, \quad B^p = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad E = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad K = \begin{bmatrix} 11.0712 & -12.4212 \end{bmatrix}. \tag{3-13}$$

The pair of complex poles is $0.05 \pm 5i$. For the PETC implementation the system is discretized with $\Delta = 0.05$. The associated Lyapunov function is

$$P = \begin{bmatrix} 70.7388 & 18.32 \\ 18.32 & 7.1454 \end{bmatrix}, \quad Q = \begin{bmatrix} 82.4416 & 24.3036 \\ 24.3036 & 10.263 \end{bmatrix}, \tag{3-14}$$

which results in $\lambda_0 = 1$. Because we have to choose a smaller decay rate, we set $\lambda = 0.8$. The last parameters to choose are the sub-sampling factor, which in this case will be $p = 4$, and k_{max} that we set to 20. That corresponds to a prediction horizon of $p \cdot \Delta \cdot k_{max} = 4$ seconds. After sub-sampling there are 20 matrices Q_k. Since it is a 2D system, the projection is only made on one plane and for every Q_k we get an associated range for θ_1 by solving problem (3-9) with step $d\theta = \pi/100$.

The result of projections, that can be found in appendix B-1, looks promising. Only for 2 time instants the projection of the safe region is the whole plane ($[0, \pi]$ - half of the plane; the other half is equivalent). 5 of the angular ranges are empty, what can suggest that at these time instants no states are safe to trigger. This gives strong basis to believe that the online computations will be reduced significantly. There are high chances that multiplications can be skipped because the current sample $x(t_i)$ will not belong to the precomputed angular ranges if they are small.

Table 3-1 shows the computation times of finding k_{trig} with and without projections. The computations were run using Python 3.7 on a SAMSUNG NP550P5C with 2.5GHz Intel Core i5 and 8GB RAM. The test was performed for 10 different $x(t_i)$ chosen randomly, where the states could vary between -10 and 10. We measured the time it took to perform the same computation 10000 times in order to obtain some measurable values. In almost all the cases using projections significantly reduced the computation time. On average this method was 1.34 times faster compared to performing only quadratic multiplications $x(t_i)^T Q_k x(t_i)$. Using projections was not beneficial only in one case - when the computed k_{trig} was equal to k_{max}. In such case only one multiplication $x(t_i)^T Q_k x(t_i)$ was needed and using projections required also computing angle θ on top of that.

Table 3-1: Computation time of finding k_{trig} for the 2D system with 20 matrices Q_k, $p\Delta = 0.2$ and prediction horizon of 4 seconds for 10 different cases

Case	1	2	3	4	5	6	7	8	9	10
Comp. time using projections ($10^{-4}s$)	**0.3**	**1.31**	0.22	**1.33**	**0.26**	**1.01**	**1.36**	**0.31**	**0.32**	**0.27**
Comp. time without projections ($10^{-4}s$)	1.06	1.95	**0.11**	2.07	0.39	1.69	2.09	0.47	0.47	0.44
Found t_{trig} (s)	2	0	4	0.2	3.4	0.6	0	3.4	3.4	3.4

The 5D system that is used as the second example has the following model:

$$
\begin{aligned}
A^p &= \begin{bmatrix} 121.804 & -154.715 & -33.9705 & -24.4279 & -112.805 \\ 51.0953 & -64.9082 & -14.6315 & -10.3426 & -47.3816 \\ 21.988 & -27.812 & -6.12563 & -3.83315 & -20.1271 \\ 175.949 & -224.511 & -50.4842 & -35.7558 & -164.186 \\ 16.8231 & -20.7307 & -4.0754 & -3.34716 & -14.7644 \end{bmatrix} \\
E &= \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \\
B^p &= \begin{bmatrix} 3.36454 & 1.41382 & 0.610604 & 4.88482 & 0.452947 \end{bmatrix}^T \\
K &= \begin{bmatrix} -17.962 & 50.6232 & 2.65116 & -4.38992 & 3.94555 \end{bmatrix}
\end{aligned}
\tag{3-15}
$$

The system has two pairs of complex poles and one unstable pole: $0.08 \pm 4i$, $0.02 \pm 2i$ and 0.05. For the PETC implementation the system is discretized with $\Delta = 0.02$. The associated Lyapunov function is

$$P = \begin{bmatrix} 1.72883 & 0.803901 & -0.614227 & -1.14722 & -2.27204 \\ 0.803901 & 0.477912 & -0.302323 & -0.562933 & -1.01183 \\ -0.614227 & -0.302323 & 0.223054 & 0.41259 & 0.804799 \\ -1.14722 & -0.562933 & 0.41259 & 0.769898 & 1.49511 \\ -2.27204 & -1.01183 & 0.804799 & 1.49511 & 3.02528 \end{bmatrix},$$
$$Q = \begin{bmatrix} 1.88436 & 0.841942 & -0.680171 & -1.24789 & -2.53023 \\ 0.841942 & 0.497427 & -0.315428 & -0.588136 & -1.06043 \\ -0.680171 & -0.315428 & 0.255586 & 0.456787 & 0.926282 \\ -1.24789 & -0.588136 & 0.456787 & 0.835975 & 1.66484 \\ -2.53023 & -1.06043 & 0.926282 & 1.66484 & 3.49128 \end{bmatrix}. \tag{3-16}$$

The associated maximum decay rate is $\lambda_0 = 1.035$ and so we can choose $\lambda = 0.83$. For the sub-sampling we choose $p = 10$ and the maximum matrices to check by the scheduler is $k_{max} = 40$. That gives us a prediction horizon of 8 seconds.

For the 5D system, mainly the second half of projections seems to be useful in computing k_{trig} online. The angular ranges found for matrices $Q_1 - Q_{13}$ contain the whole plane for all the projection planes. Therefore for those time steps the scheduler will have to perform quadratic multiplications. Beyond that time interval, i.e. $k > 13$ ($t > 2.6s$) projections are more promising, as almost all of them have at least one angular range different than $[0, \pi]$. The results of the offline computations for this 5D system suggest that the time required for computing k_{trig} can be especially reduced for longer prediction horizons.

A similar speed test, as it was done for the 2D system, was performed also for the 5D case. This time we evaluated the computation time using 40 matrices Q_k (Tab. 3-2) and only 20 (Tab. 3-3). For the case with 40 matrices, using projection was clearly advantageous. In all the cases this method was much faster, on average 1.7 times. For the case with 20 matrices both method yielded very similar results. This can be easily explained by the fact that projections with regions different than the whole plane could be mostly found in the second half of all the matrices Q_k. For the first 20 matrices, projections were not very helpful. Nevertheless, using projections does not impede the computations and are recommended to be used in any case.

Table 3-2: Computation time of finding k_{trig} for the 5D system with 40 matrices Q_k, $p\Delta = 0.2$ and prediction horizon of 8 seconds for 10 different cases

Case	1	2	3	4	5	6	7	8	9	10
Comp. time using projections ($10^{-4}s$)	**2.58**	**2.37**	**2.63**	**2.4**	**0.99**	**2.31**	**2.5**	**2.41**	**2.13**	**2.11**
Comp. time without projections ($10^{-4}s$)	4.43	3.81	3.62	3.99	2.49	3.84	3.8	4.01	3.79	3.85
Found t_{trig} (s)	0.2	0.2	0.8	0.2	3.2	0.2	0.2	0.2	0.8	0.6

This example shows the result of offline computations. In the numerical example of the next chapter the same systems will be used to show how the precomputed data structures

Table 3-3: Computation time of finding k_{trig} for the 5D system with 20 matrices Q_k, $p\Delta = 0.2$ and prediction horizon of 4 seconds for 10 different cases

Case	1	2	3	4	5	6	7	8	9	10
Comp. time using projections ($10^{-4}s$)	2.02	**1.88**	1.65	**1.91**	**0.49**	**1.89**	1.78	2.02	**1.62**	**1.81**
Comp. time without projections ($10^{-4}s$)	**1.96**	2.09	1.65	2.04	0.51	1.92	**1.53**	**1.91**	1.66	1.88
Found t_{trig} (s)	0.2	0.2	0.8	0.2	3.2	0.2	1	0.2	0.8	0.6

can reduce online computations in a loop designed by merging PETC implementation from chapter 2 and the scheduler algorithm from chapter 3. The results so far suggest that for the 2D system, the number of online computations can be reduced in a greater extent than for the 5D system. This hypothesis will be verified in the next chapter.

Chapter 4

Relaxed PETC combined with the scheduler

The last step of the roadmap is to merge concepts introduced in the two previous chapters. In this set-up, a communication request triggered by the relaxed PETC can be suppressed by the scheduler until the time interval when the last event in the time horizon is predicted. Skipping some events results in a smaller overall number of transmissions in the network. One loop in a NCS that uses both the relaxed PETC and the scheduler that predicts future events is shown in Fig. 4-1. Because each loop requires less attention under this control strategy, it could be possible to connect more loops with the same network, effectively resulting in a less expensive realization of a NCS.

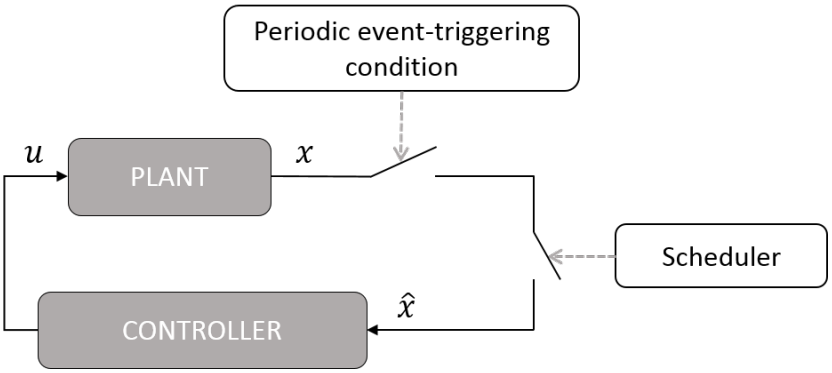


Figure 4-1: Combined design of the control loop.

As far as the performance of the system is concerned, one has to be aware of the possibly much higher amplitude of the system trajectories when using the scheduler. This is due to the fact that the scheduler does not consider the value of the triggering condition before the found k_{trig}. When access to the network is closed, system trajectories are unknown. The scheduler only knows that they will be back in a region safe to trigger at time instant k_{trig}.

Because of that, the recommendation for using this control strategy is primarily for NCSs that are very congested or very scarce in resources. Applying this kind of scheduler can be especially beneficial in case of a conflict in the network. When two loops request access to the network, the scheduler can quickly check if any of them will experience another crossing and can wait until then. Overall, the combined loop presented in this chapter should not be the first choice in controlling a single loop because of its inferior performance, but can be very helpful in NCSs where the chance of communication conflicts is high. Apart from that, when controlling a single loop there are usually no strict requirements on the number of communications and applying relaxed PETC only is sufficient.

4-1 Stability analysis

In order to find performance gains for the control loop combining the relaxed PETC and the scheduler, we will follow similar steps as in section 2-2-1.

4-1-1 Guarantees

Before stating GES and EISS, we start by introducing the function

$$g_s(\tau_{max}) \coloneqq e^{\frac{\omega\mu\tau_{max}}{\mu-\omega}}\left((e^{\frac{\lambda\omega\tau_{max}}{\mu-\omega}}+e^{\frac{\lambda\mu\tau_{max}}{\mu-\omega}})^{\frac{-\omega}{\mu-\omega}}(2e^{\frac{\lambda\mu\tau_{max}}{\mu-\omega}})^{\frac{\mu}{\mu-\omega}}\right) \tag{4-1}$$

with

$$\tau_{max} = pk_{max}\Delta.$$

Theorem 4.1. *If $\lambda < \lambda_0$ and $\Delta < \tau_{min}$, the PETC implementation (2-14) combined with a scheduler running algorithm 1 as shown in Fig. 4-1 renders the closed loop system (1-1) GES with*

$$\sigma = \left(\frac{\lambda_M(P)}{\lambda_m(P)}\right)^{\frac{1}{2}}(g_s(\tau_{max}))^{\frac{1}{2}}, \quad \rho = \frac{1}{2}\lambda. \tag{4-2}$$

Before showing the performance gain in case when bounded disturbance is present, we first recall one function defined earlier in this thesis:

$$\gamma_{P,T}(s) = s\frac{\lambda_M(P)}{\lambda_m^{\frac{1}{2}}(P)}\int_0^T |e^{A^p r}E|dr$$

Theorem 4.2. *If $\lambda < \lambda_0$, $\Delta < \tau_{min}$ and $\delta \in \mathcal{L}_\infty$, the PETC implementation (2-14) combined with a scheduler running algorithm 1 as shown in Fig. 4-1 yields the closed loop system (1-1) EISS with*

$$\gamma(||\delta||_\infty) = \gamma_s(||\delta||_\infty) \coloneqq \lambda_m^{-\frac{1}{2}}(P)\gamma_{P,\tau_{max}}(||\delta||_\infty)\left(\frac{(g_s(\tau_{max}))^{\frac{1}{2}}}{1-e^{-\frac{1}{2}\lambda\Delta}}+1\right).$$

For the proofs the reader is referred to Appendices: A-3 and A-4.

4-1-2 Disturbances in the prediction

The performance guarantees for the control loop with the relaxed PETC triggering mechanism combined with the scheduler are not as good as most of the control engineers would like to have. The performance gains obtained in the previous subsections can grow to large numbers due to suppressing the communication by the scheduler. The gains are dependent on $g_s(\tau_{max})$ which grows exponentially with the increase of $\tau_{max} = pk_{max}\Delta$. Moreover, because of the STC-like limited access to the network, the relaxed PETC has lost its ability of being robust toward disturbances, which was one of the most important advantages. Therefore, to recover that feature, it is possible to include the disturbances in the prediction of the scheduler by adding some conservatism. This idea is inspired by [14] and [30], where a similar approach was presented for the output-feedback STC case.

In order to predict the worst-case value of the triggering condition (3-2) under the unknown disturbances in the consecutive instances of time, one has to perform reachability analysis. Before we introduce the necessary definitions, we first denote a solution to system (1-1) with initial state x_0, control input $\hat{u}$ and disturbance δ with:

$$x_{x_0\hat{u}w} = e^{A^p t}x_0 + \int_0^t e^{A^p(t-s)}(B^p\hat{u}(s) + E\delta(s))ds \tag{4-3}$$

Let $\mathcal{F}_\mathcal{U}$ and $\mathcal{F}_\mathcal{W}$ be the sets of essentailly bounded piecewise continuous functions from $\mathbb{R}^+$ to $\mathcal{U}$ and $\mathcal{W}$, respectively. Assuming that $u \in \mathcal{F}_\mathcal{U}$ and $\delta \in \mathcal{F}_\mathcal{W}$ we would like to find a set of all possible solutions $x_{x_0\hat{u}\delta}$ according to (4-3), that we call a reachable set.

Definition 4-1.1 (Reachability operator [30, Def. 1]). *Given an initial time t_1, a final time t_2, an initial state set $\mathcal{X}(t_1)$ and the input sets $\mathcal{U}$ and $\mathcal{W}$, the reachability operator* reach$(\cdot)$ *is defined as* reach$(t_1,\ t_2,\ \mathcal{X}(t_1),\ \mathcal{U},\ \mathcal{W}) \coloneqq \{x_{x_0u\delta}(t_2) : x_{x_0u\delta}(t_1) \in \mathcal{X}(t_1), \forall\hat{u} \in \mathcal{F}_\mathcal{U}, \forall\delta \in \mathcal{F}_\mathcal{W}\}$.

In our case the initial state $x(t_i)$ is known, as well as the control input $\hat{u} = Kx(t_i)$. The only unknown signal is the disturbance $\delta(t)$ and due to linearity we can separate its effect on the solution. The resulting reachable set can be written with modified [30, Eq. (19)]:

$$\mathcal{X}(t_i + \Delta k) = \Phi(k)x(t_i) + \Gamma(k)\hat{u} + \mathcal{X}_w(k), \tag{4-4}$$

where

$$\Phi(k) \coloneqq e^{A^p\Delta k} \quad \text{and} \quad \Gamma(k) \coloneqq \int_0^{\Delta k} e^{A^p s}B^p ds$$

and $\mathcal{X}_w(k) \coloneqq \text{reach}(0, \Delta k, 0, 0, \mathcal{W})$. Reachable sets can be of arbitrary complexity, that sometimes are difficult to work with. Therefore, there exist methods of outer-approximating the reachable sets with predefined shapes. One of them are ellipsoids, with the main advantage over spheres, that can more tightly outer-approximate the reachable set.

Definition 4-1.2 (Ellipsoid [31, Chap. 2]). *Let $m \in \mathbb{R}^n$ and $M \succ 0 \in \mathbb{R}^{n\times n}$. An ellipsoid can be described in the inequality form*

$$\mathcal{E}(m, M) = \{x \in \mathbb{R}^n : (x - m)^T M^{-1}(x - m) \leq 1\}.$$

If the matrix W, such that $\mathcal{W} \subseteq \tilde{\mathcal{W}} = \mathcal{E}(0, W)$ is known, the ellipsoidal toolbox for MATLAB [32] can outer-approximate the disturbance response $\mathcal{X}_w(k)$ with $\tilde{\mathcal{X}}_w(k) \coloneqq \mathcal{E}(0, W_k)$. The resulting problem of finding the worst-case triggering function value at time instant k is the following:

$$\begin{aligned} \max_{x,d} \quad & \psi(x, \hat{x}, k) = \begin{bmatrix} x^T & \hat{x}^T \end{bmatrix} \bar{Q}(k) \begin{bmatrix} x \\ \hat{x} \end{bmatrix} \\ \text{subject to} \quad & x = R(k)\hat{x} + d \\ & d^T W^{-1} d \leq 1 \end{aligned} \tag{4-5}$$

with d being the contribution of the unknown disturbances to states at time $r_k = t_i + \Delta k$ and $R(k)$ defined with (3-1). Since $\bar{Q}(k)$ is not a definite matrix, the above written optimization problem is non-convex. Following [30] we instead compute a conservative upper bound for $\psi(x, \hat{x}, k)$. Let us define:

$$R_w(k) = F_w(k) W_k F_w^T(k), \quad F_w(k) = \begin{bmatrix} R^T(k) & I \end{bmatrix} \bar{Q}(k) \begin{bmatrix} I \\ 0 \end{bmatrix}.$$

Corollary 4.1. *Function*

$$\bar{\psi}(\hat{x}, k) \coloneqq \hat{x}^T Q_k \hat{x} + 2\sqrt{\hat{x}^T R_w(k) \hat{x}} + \lambda_M(W_k \bar{Q}(k)) \tag{4-6}$$

provides an upper bound for the solution of (4-5).

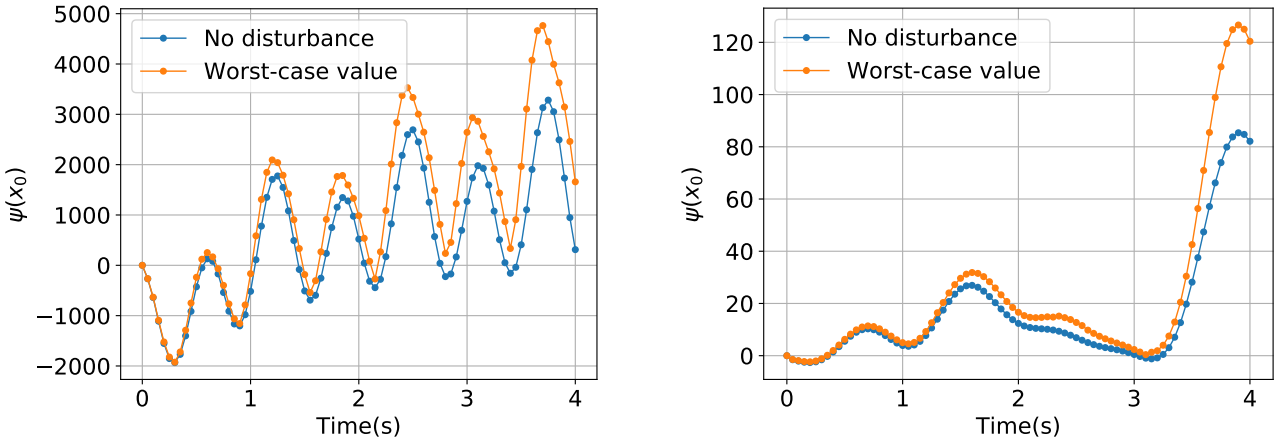
Proof. The proof for a more general case is given in [30]. After simplifying the original problem from output to state feedback, considering a static controller and measurement noise-free case, one obtains formula (4-6). □

Matrices W_k can be computed offline using the aforementioned ellipsoidal toolbox [32]. Other structures that can also be precomputed are $R_w(k)$ and $c \coloneqq \lambda_M(W_k \bar{Q}(k))$.

Figure 4-2 compares the predicted values of the triggering condition in the worst-case scenario (4-5) and without including disturbances (3-3) at discrete time instants Δk if the control action is not updated throughout the simulation. Fig. 4-2(a) was generated for the 2D system introduced in chapter 3 with initial condition $x(0) = \begin{bmatrix} -5 & -3 \end{bmatrix}^T$, while Fig. 4-2(b) for the 5D system with initial condition $x(0) = \begin{bmatrix} 3 & -7 & -3 & -1 & 4 \end{bmatrix}^T$. For both cases, it was assumed that $||\delta||_\infty = 0.1$ and hence $W = 0.1^2$ was set for the ellipsoidal toolbox.

As it can be seen in the plots, for the 2D system the last predicted event in the time horizon of 4 seconds shifts from 3.45 s to 2.15 s, if we decide to use the worst-case value prediction. For the 5D system including the disturbance in the prediction results in discarding the possibility to trigger later. Moreover, the further the prediction the bigger the gap between two values of the triggering condition tends to be. It is something to be expected because at every sampling time the reachable set get bigger.

Ellipsoids based on matrices W_k are outer-approximations of the reachable set $\mathcal{X}_w(k) \coloneqq \text{reach}(0, \Delta k, 0, 0, \mathcal{W})$. It is in our interest to find them as tight as possible to avoid extensive conservatism in the prediction. The ellipsoidal toolbox that is used to compute W_k offers



(a) Triggering function of the 2D system

(b) Triggering function of the 5D system

Figure 4-2: Comparison of the predicted value of the triggering condition in a disturbance free case (3-3) and in the worst-case scenario (4-6) for the 2D (3-13) and 5D systems (3-15).

different options that can be used to initialize the toolbox. Changing parameters, such as tolerances or differential equations solver, significantly affects the resulting matrices W_k. Some settings raise warnings of ellipses resulting from reachability not having their base matrices positive (semi-)definite. After trying several settings for different systems, it can be reasoned that on average the best results are obtained for ode solver using Runge-Kutta method with adaptive step size and second and third order formulae ('RK23'). It is, however, not guaranteed that this solver is the best for a particular system. It is also worth noting that including disturbances in the prediction does not always yield a smooth trajectory of the triggering condition. It is dependant on the direction of minimization of the ellipsoid, as well as the chosen optimization method. Therefore, if there appear jumps in the prediction, it should not be considered as a bad result, as there are multiple ways one can outer-approximate the reachable set with an ellipsoid.

For the case with estimating the disturbances the projections method described in section 3-2-2 cannot be used to speed up online computations. Including disturbances in the prediction changes the shape of the region from a cone to a hyperboloid. A projection of a hyperboloid is not a 2D cone as in Fig. 3-3, which is a requirement for this method. As a result, only sub-sampling can be used to decrease the number of computations online. However, it is still a powerful method that can reduce computations severalfold. The scheduler's algorithm in this case just performs computations $x(t_i)^T Q_k x(t_i) + 2\sqrt{x(t_i)^T R_w(k) x(t_i)} + c$ one by one starting from $k = k_{max}$ until it finds k_{trig}, for which the triggering condition is smaller than 0.

4-2 Numerical example

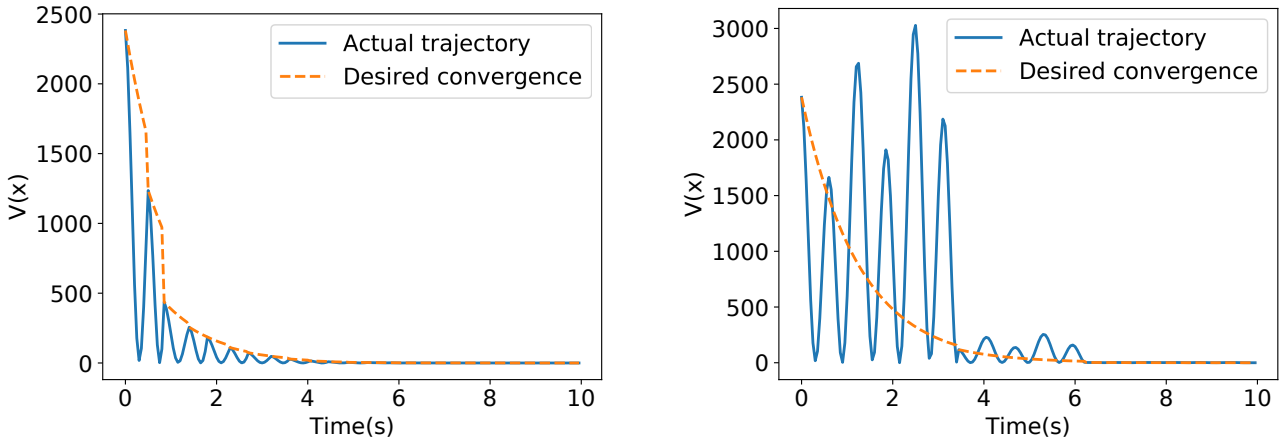
To show how systems perform in a combined loop with the PETC triggering mechanism and the scheduler, we choose the same examples as in the numerical example of chapter 3. We

show it in a comparison to the implementation with the relaxed PETC only.

4-2-1 Disturbance-free case

A simulation in a disturbance-free case of the 2D system (3-13) with the Lyapunov function (3-14) is presented in Fig. 4-3 and 4-4. The initial condition in this case was $x(0) = \begin{bmatrix} -5 & -3 \end{bmatrix}^T$. When the scheduler was included in the loop, the number of events in the first 10 seconds of the simulation was 4, in comparison to 23 when it was not used. Moreover, the number of online computations performed by the scheduler was small. For the first 3 events it only had to multiply $x(t_i)^T Q_k x(t_i)$ once and for the 4th one the number of these multiplications was 4. The maximum number of multiplications encountered if the simulation was allowed to continue for longer was 8 out of 20 matrices Q_k after sub-sampling. The reason behind efficient computations online are the data structures precomputed offline presented in the previous chapter. For the 2D system the angular ranges from the projections were most of the cases smaller than the whole plane. Together with vector *useful*, they allowed to skip most of the quadratic multiplications $x(t_i)^T Q_k x(t_i)$.

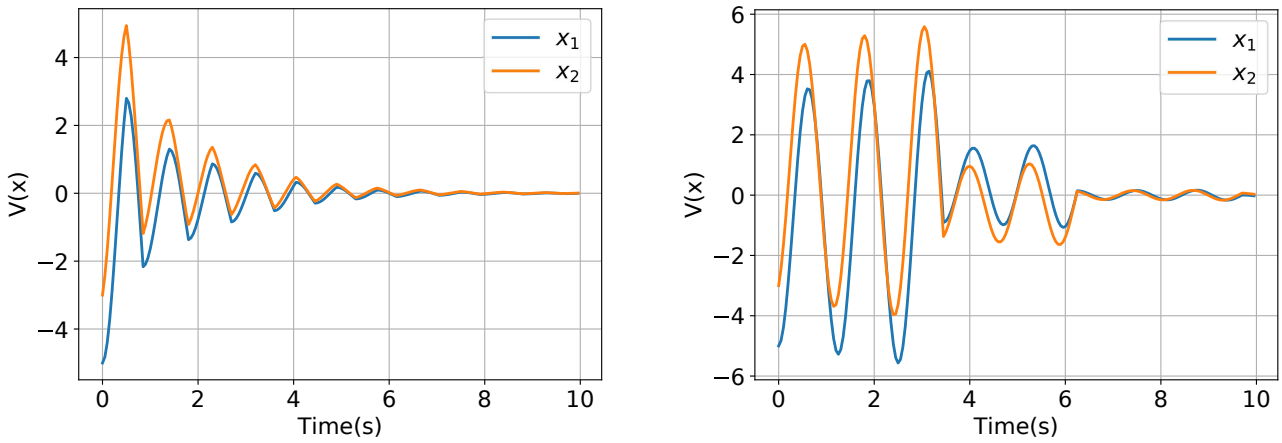
The substantial reduction of communication between the sensor and the controller was at the expense of a much greater magnitude of the Lyapunov function, what was also reflected in the behaviour of the states. Although the settling time when using the scheduler was not much longer than without it, the amplitude of the oscillations of the states was significantly bigger. One positive aspect of having less events was that the trajectory of the states was smoother. When using the relaxed PETC only, we can see sharp spikes that in some applications can be more dangerous than the higher magnitude of the system in a combined loop. What is also worth noticing is the substantial reduction of the Lyapunov function's magnitude after the second trigger in the case with the scheduler. This behaviour shows that the reduced number of triggering times is still very effective in stabilizing the system.



(a) Relaxed PETC

(b) Relaxed PETC with a scheduler

Figure 4-3: Evolution of the Lyapunov function (3-14) of the 2D system (3-13).



(a) Relaxed PETC

(b) Relaxed PETC with a scheduler

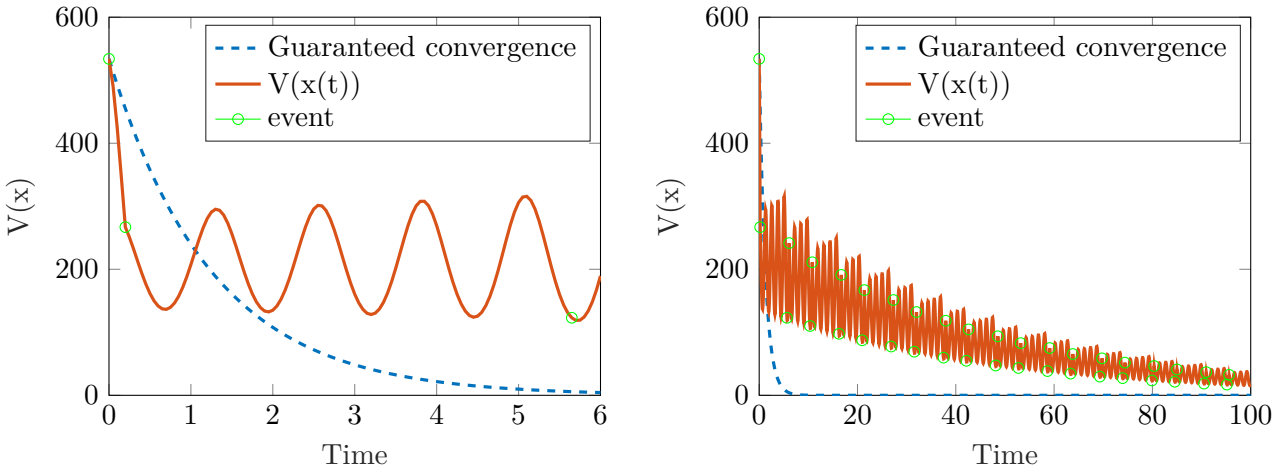
Figure 4-4: Evolution of the states of the 2D system (3-13).

Since this 2D system used in this chapter has an oscillatory behaviour, that is substantially different from the system (2-18) from chapter 2, it is worth checking how other PETC approaches cope with it. In order to find parameter σ for PETC (7) from [1] and dynamic PETC from [2], as well as parameter β from [1], LMIs were solved with the same values as in this numerical example: $\Delta = 0.05$, $\lambda = 0.8$ and $\rho = \lambda/2 = 0.4$. Finding β ran without any problems and the biggest possible value (to minimize the number of transmissions) was found to be $\beta = 0.99$. Solving LMIs in search of both parameters σ, would often result in 'Inaccurate/Solved' status of the problem. For the dynamic PETC obtained this way value $\sigma = 0.004$ gave reasonable simulation results. However, for PETC (7) from [1] (with state error as triggering condition), an 'Innacurate/Solved' value of $\sigma = 0.24$ resulted in stable, but unexpected simulations that are shown in Fig. 4-5. For all the checked initial conditions, the trajectories of the Lyapunov function were similar, that differed only in the magnitude and shape of oscillations.

A possible explanation for such behaviour can be that since the states oscillate, the norm $||x - \hat{x}||$ also oscillates, but rarely exceeds $\sigma||x||$. The system remains stable and with the right choice of gain σ from GES definition 1-2.1, the convergence rate is $\rho = \lambda/2 = 0.4$. The number of transmissions is small, but the settling time is undesirably long.

Having settling time over 100 seconds does not seem like a fair comparison with other PETC strategies, that stabilize in time below 6 seconds. Therefore, for the PETC (7) with the triggering condition based on the state error, the search for parameter σ continued until the status of the solved problem was accurate. The obtained value was $\sigma = 0.005$, which in turn resulted in transmissions occurring at every sampling instant.

Comparison of the number of transmissions based on 15 different initial conditions for all the considered PETC strategies are illustrated in Fig. 4-6. For $\lambda = 0.8$ the relaxed PETC with the scheduler had the smallest median of the number of transmissions. However, the standard deviation for both relaxed PETCs was very big. The number of triggering instants depended on the initial conditions to a large extent. What was observed is that at times the



(a) First 6 seconds

(b) First 100 seconds

Figure 4-5: Evolution of the Lyapunov function for the 2D oscillatory system (3-13) under the PETC (7) from [1] with $\Delta = 0.05$, $\sigma = 0.24$, $\lambda = 0.8$.

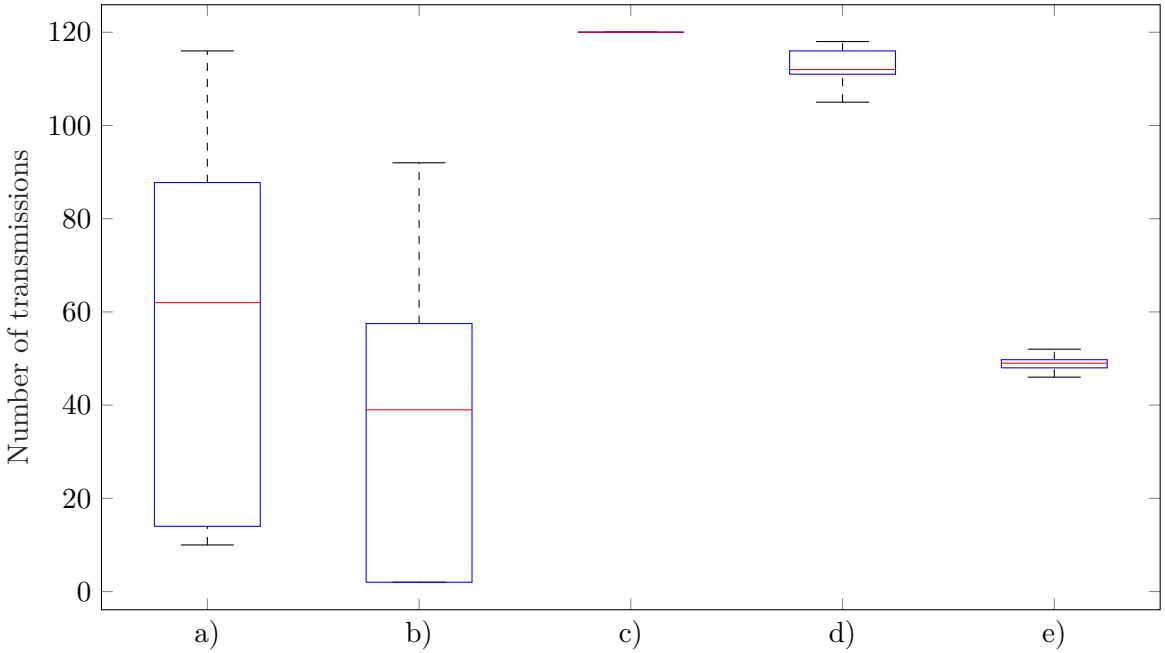


Figure 4-6: Number of transmissions for system (3-13) with $\lambda = 0.8$ under: a) Relaxed PETC, b) Relaxed PETC with scheduler, c) PETC (7) from [1], d) PETC (14) from [1] and d) dynamic PETC from [2].

systems was 'trapped' in a state that required periodic updates of control actions. In this case, the relaxed PETC would behave similarly to other static PETCs. On the other hand, the dynamic PETC has a buffer mechanism that prevents from triggering in two consecutive time instants. Hence, dynamic PETC repetitively achieved the number of transmissions that was slightly smaller than half of all the sampling instants.

While solving LMIs, one often encounters numerical problems. Although the dynamic PETC achieved very good results in the previous experiment, it might not always satisfy the guarantees, because the status of the SDP problem was 'Inaccurate/Solved'. Therefore another comparison is shown in Fig. 4-7, where the result of the LMI solution was accurate. To obtain such result the convergence rate had to be decreased to $\lambda = 0.4$.

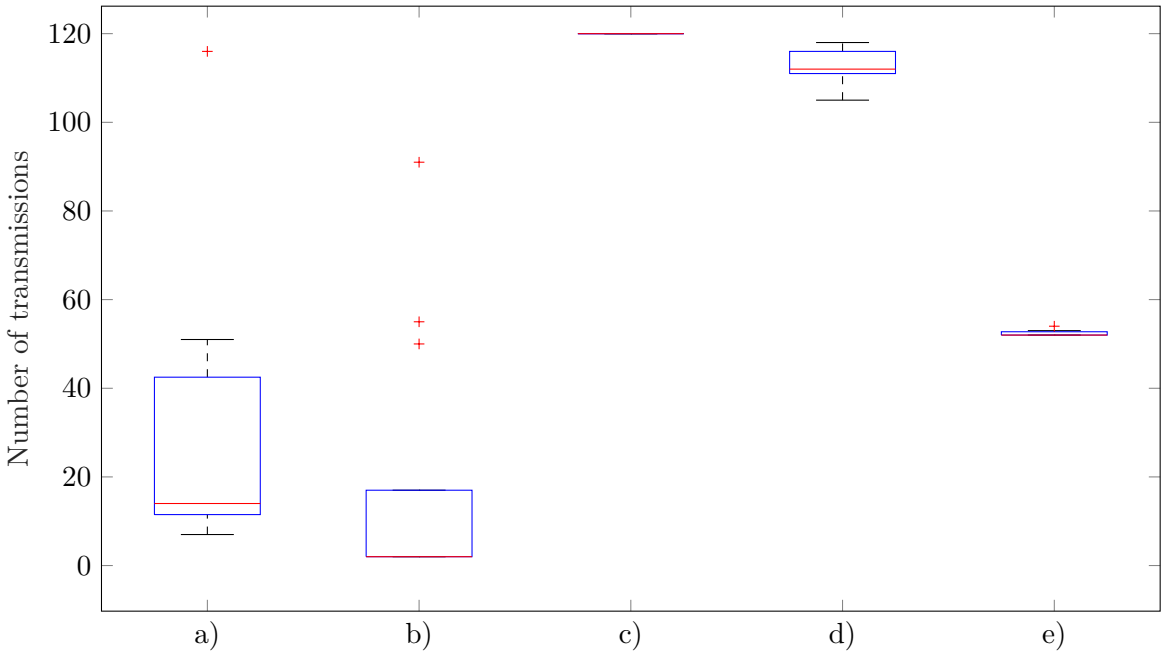


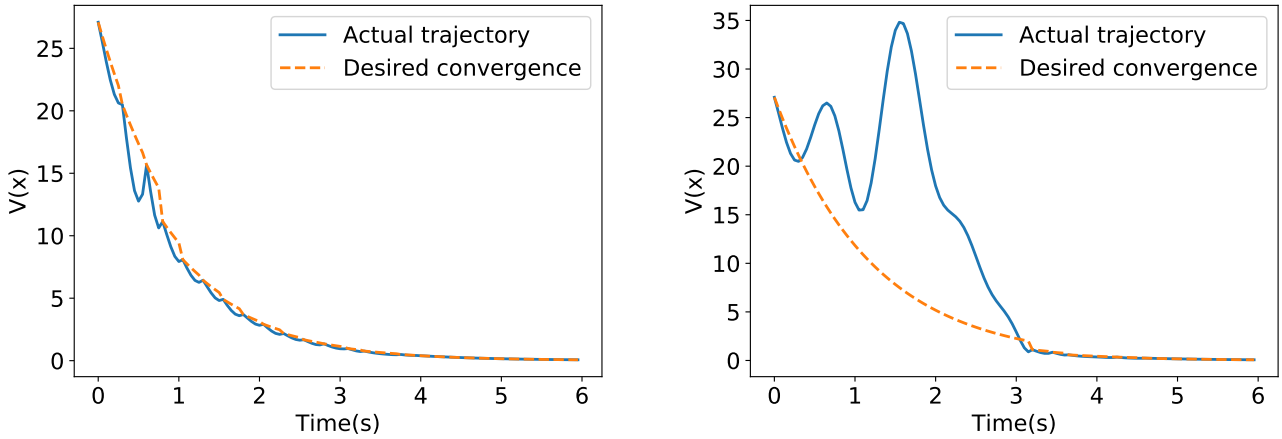
Figure 4-7: Number of transmissions for system (3-13) with $\lambda = 0.4$ under: a) Relaxed PETC, b) Relaxed PETC with scheduler, c) PETC (7) from [1], d) PETC (14) from [1] and d) dynamic PETC from [2].

Smaller convergence rate did not change much for static PETCs and dynamic PETC, but significantly reduced the standard deviation of both relaxed PETCs. The bound put on the Lyapunov function was less strict and the system was not getting stuck in difficult states often.

Overall, we can conclude that oscillatory systems are more difficult to control with little transmissions. Static and dynamic PETCs resulted in practically periodic control, with distinction in the period. Static PETCs transmitted almost every sampling instant, while dynamic PETC every second sampling instant. On the other hand, both relaxed PETCs only sometimes required periodic control with period Δ.

In order to test the relaxed PETCs for systems of higher orders, the second simulation was done for the 5D system introduced in chapter 3 ((3-15) and (3-16)). The initial condition was taken as $x(0) = \begin{bmatrix} 3 & -7 & -3 & -1 & 4 \end{bmatrix}^T$. Fig. 4-8 shows the trajectory of the Lyapunov function, while Fig. 4-9, of the states for both cases - relaxed PETC only and the combined control loop with the scheduler. The former case resulted in 39 events, while the latter in 27. Similarly to the 2D system, the simulation with the scheduler experienced much higher magnitude, but also much smoother trajectories. Here, not only the oscillations are much sharper when using relaxed PETC only, but they are also much more frequent.

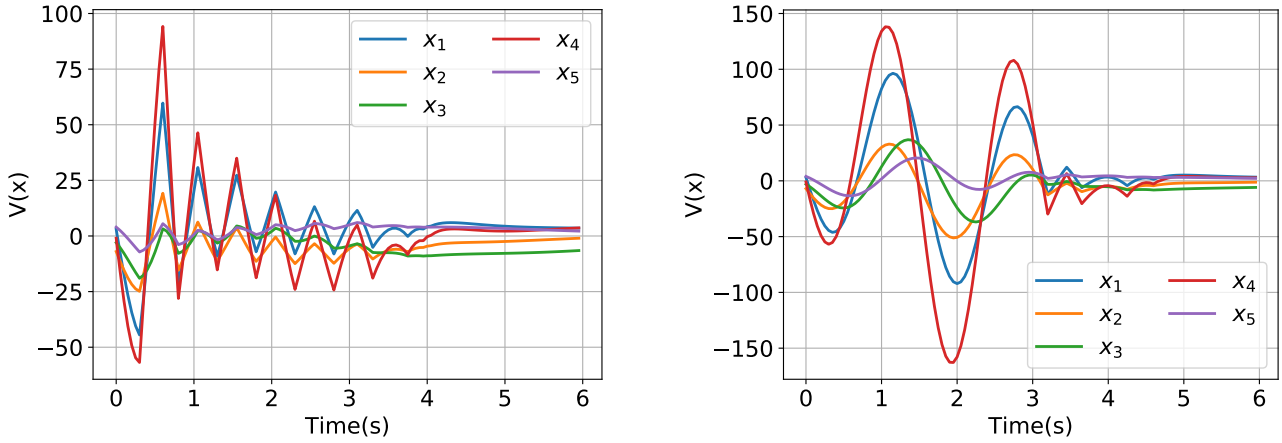
Out of 40 possible quadratic multiplications after sub-sampling, the scheduler had to perform, on average, 18.19 per prediction. The maximum number of multiplications encountered in a long simulation was 19. Overall, the online computations were reduced more than twofold. However, it had to do with the fact that the prediction horizon was long (8 seconds) and mostly the second half of matrices Q_k had useful angular ranges associated with them (see Appendix B-2). If the prediction horizon was 4 seconds, only few matrices would be possible to skip due to the projections. Therefore, it might be beneficial to set longer prediction horizons, since the number of computations that have to be performed online is comparable to shorter horizons but the number of events can possibly be reduced even further.



(a) Relaxed PETC

(b) Relaxed PETC with a scheduler

Figure 4-8: Evolution of the Lyapunov function (3-16) of the 5D system (3-15).



(a) Relaxed PETC

(b) Relaxed PETC with a scheduler

Figure 4-9: Evolution of the states of the 5D system (3-15).

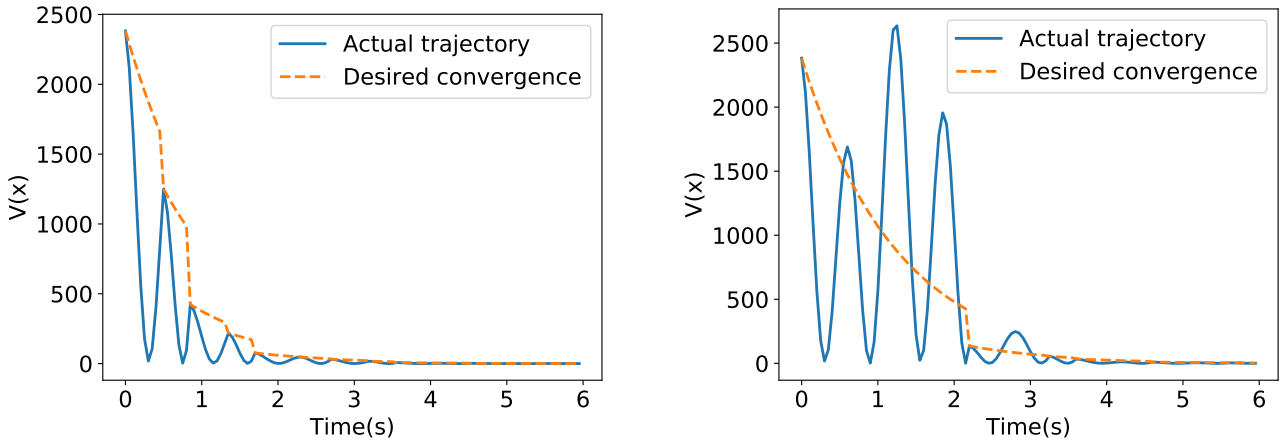
Concluding, the number of computations performed online can be reduced significantly by

using precomputed offline data structures. The hypothesis from chapter 3, that it is more beneficial for systems of smaller dimensions, is confirmed. The 5D system experienced slightly more than 2 times reduction of quadratic multiplications and it was mostly because the prediction horizon was sufficiently long. For a shorter horizon the data structures would be less useful and the reduction would also be smaller. In comparison, for the 2D system only a quarter of all possible quadratic multiplications were performed on average when searching for the last event. This would not change much if the prediction horizon was shorter because, for all the matrices Q_k, the precomputed data structures were useful.

4-2-2 Perturbed case

In the presence of disturbances, one can apply the same algorithm for the scheduler and accept possible high gains from disturbance to states or modify the algorithm according to section 4-1-2. In this section, the performance of a control loop with a modified scheduler is presented. Again, we use the same 2D and 5D systems as before and compare the evolution of their Lyapunov functions under the relaxed PETC and under the relaxed PETC combined with the scheduler.

Fig. 4-10 shows the Lyapunov function of the 2D system perturbed with $\delta(t) = 0.1\sin(t)$. In the time horizon of 4 seconds, the relaxed PETC triggered 11 times, while adding the scheduler to the loop reduced this number to 6. The small disturbance that was applied to the system did not have a big effect on the system trajectories but it changed the scheduler's prediction. Instead of predicting 3.4 seconds in the beginning of the simulation, as it was the case in the disturbance-free case, this time the predicted last event was supposed to occur at 2.2. This was already suggested by Fig. 4-2(a), which used the same initial condition.



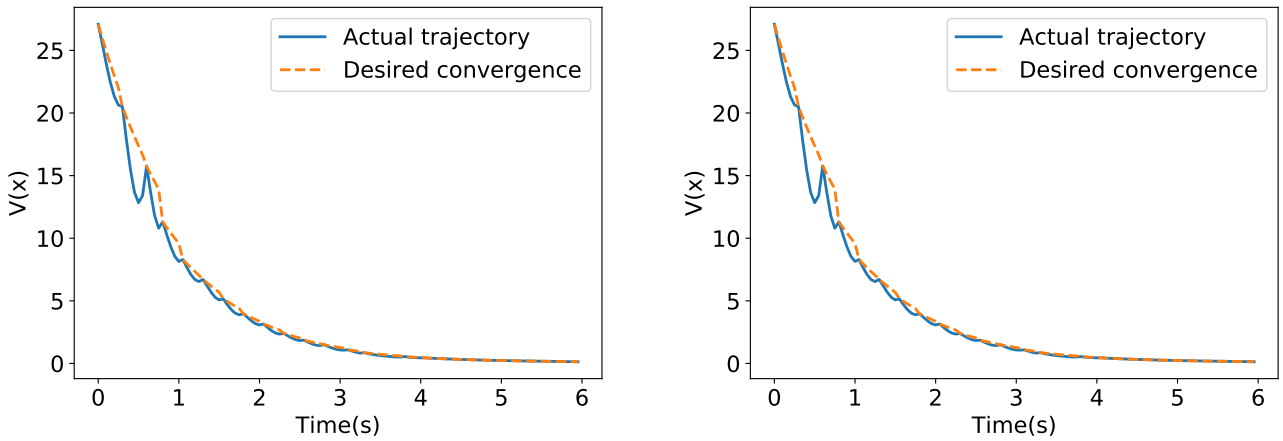
(a) Relaxed PETC

(b) Relaxed PETC with a scheduler

Figure 4-10: Evolution of the Lyapunov function (3-14) of the 2D system (3-13) with including disturbance in the scheduler's prediction and $\delta(t) = 0.1\sin(t)$.

For the 5D system including disturbance in the prediction resulted in almost identical evolution of the Lyapunov function regardless of the use of the scheduler, as can be seen in

Fig. 4-11. One possible cause for that can be the fact that the worst-case value of the triggering condition seems more conservative in comparison to e.g. the 2D system. In Fig. 4-2(b) the gap between predictions made in two ways was significant compared to the magnitude of the reference triggering condition. Moreover, the chosen 5D system does not experience as big oscillations as e.g. the 2D system. This two aspects combined resulted in the removal of the future events in the worst-case scenario prediction.



(a) Relaxed PETC

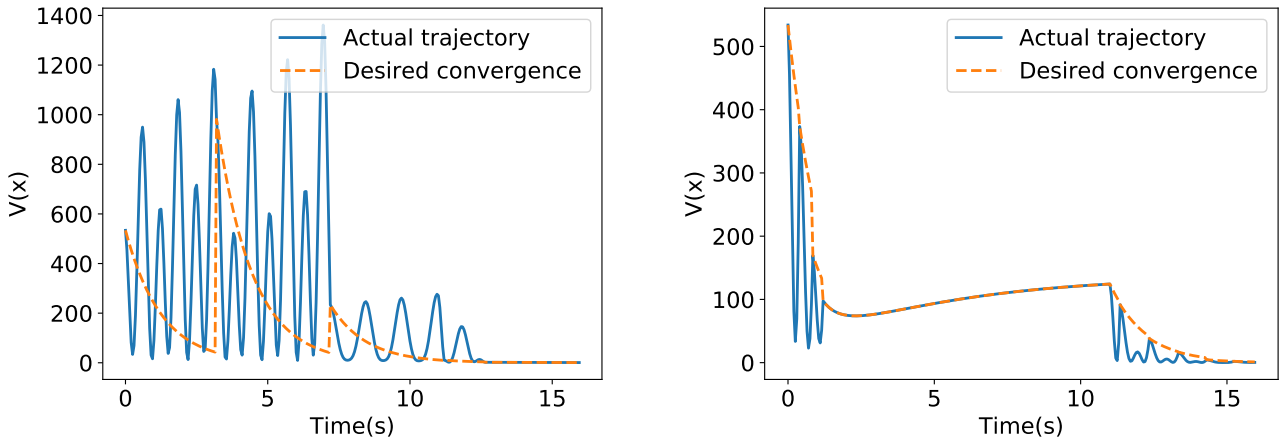
(b) Relaxed PETC with a scheduler

Figure 4-11: Evolution of the Lyapunov function (3-16) of the 5D system (3-15) with including disturbance in the scheduler's prediction and $\delta(t) = 0.1\sin(t)$.

Lastly, it is reasonable to see how the control loop with a scheduler performs under disturbances of higher magnitude that significantly changes system trajectories. It should be investigated how including the disturbances in scheduler's prediction compares to the algorithm 1 that neglects them.

For the 2D system, we chose a constant disturbance signal $\delta(t) = -3$ for $t \leq 11$ and $\delta(t) = 0$ for $t > 11$. Fig. 4-12 illustrates the trajectory of the Lyapunov function in cases when disturbances were neglected or included in the scheduler's prediction for initial condition $x(0) = \begin{bmatrix} 3 & -1 \end{bmatrix}^T$. Worst-case prediction for disturbances of higher magnitude resulted in an identical trajectory as using relaxed PETC without a scheduler. The conservativeness of the prediction did not allow the Lyapunov function to exceed the bound and the predicted t_{trig} was always associated with the first upcoming event (Fig. 4-12(b)). For this case the overall number of transmissions was 205. On the other hand, neglecting disturbances resulted in only 71 events (Fig. 4-12(a)). However, it can be seen that the magnitude of the Lyapunov function is significantly higher, what directly translates to state trajectories. Moreover, the exponential bound had to be reset to a much higher value at every triggering instant. This is related to high performance gains obtained for this implementation in section 4-1 on stability analysis.

A similar disturbance was applied to the 5D system. Here, $\delta(t) = -2$ for $t \leq 11$ and $\delta(t) = 0$ for $t > 11$. The initial condition was $x(0) = \begin{bmatrix} 3 & -7 & -3 & -1 & 4 \end{bmatrix}^T$. The number of

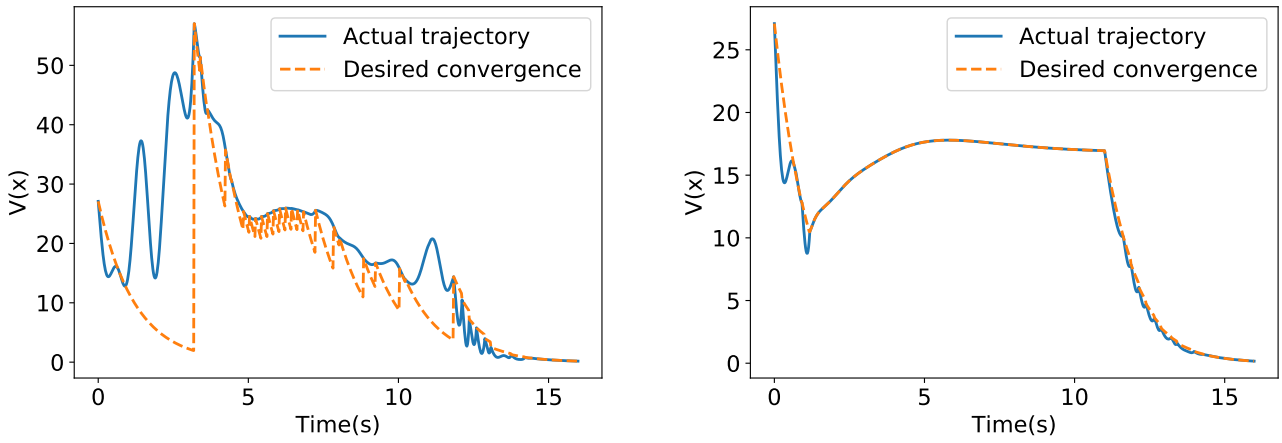


(a) Neglecting disturbances

(b) Disturbance included in the prediction

Figure 4-12: Evolution of the Lyapunov function (3-14) of the 2D system (3-13) with different algorithms for scheduler's prediction and $\delta(t) = -3$ for $t \leq 11$ and $\delta(t) = 0$ for $t > 11$.

transmissions for the case when the disturbances were neglected was 53 (Fig. 4-13(a)), while including the disturbances in the prediction (Fig. 4-13(a)) resulted in 532 events out of all 800 sampling times. For the 5D system, the relaxed PETC without the scheduler yielded more transmissions - 556. The observations for the 5D system are similar to the 2D system, but here they are even more visible. Neglecting the disturbances caused the exponential bound to be reset to a higher value every time until the disturbances vanished. Moreover, the trajectory of the Lyapunov function was very oscillatory, as opposite to fairly smooth trajectory when the scheduler included the disturbances in the prediction.



(a) Neglecting disturbances

(b) Disturbance included in the prediction

Figure 4-13: Evolution of the Lyapunov function (3-16) of the 5D system (3-15) with different algorithms for scheduler's prediction and $\delta(t) = -2$ for $t \leq 11$ and $\delta(t) = 0$ for $t > 11$.

For both systems, persistent disturbance up to time $t = 11$ prevented the system to settle in the origin. Therefore, also the Lyapunov function could not converge to 0 what is clearly visible on Fig. 4-12(b) and Fig. 4-13(b). In case of including the disturbance in the prediction, this situation resulted in triggering at every sampling instant until the disturbance vanished. This is also the main reason of having a significantly greater number of communication instants, when the scheduler includes the disturbances in computing time for granting access to the network.

The conclusion from the last experiment is that, when choosing the algorithm for the scheduler, one has to decide on the main objective of the implementation. Regardless of the algorithm used inside the scheduler, stability of the system is guaranteed. However, if one's goal is to minimize the number of transmissions, a better option might be to neglect the disturbances. On the other hand, the magnitude of states is then likely to be high and the performance gains from disturbance to states as well. Therefore, it is important to choose the prediction horizon wisely, depending on what gains one can agree on. Including the disturbances in the prediction keeps the magnitude of system trajectories smaller and the robustness against disturbances, which is characteristic to ETC implementations, in maintained. Nevertheless, when the magnitude of the disturbances is known to be high, it could make more sense to use the relaxed PETC without the scheduler instead. The conservativeness of the prediction often results in almost identical behaviour of the system, but the computational effort is higher due to the search for t_{trig}.

Chapter 5

Conclusions and future work

5-1 Conclusions

This thesis presented a relaxed triggering condition for PETC, that reduces the number of communication instants compared to existing PETC implementations. The novelty lies in allowing the Lyapunov function of the continuous-time closed-loop system to increase up to an exponentially decaying bound. As a result the controlled system stays globally exponentially stable and exponentially input to state stable. The relaxed PETC is a good solution for NCSs where the communication resources are scarce. Because the plants require less transmissions, it allows more loops to be connected by the same network.

For highly congested NCSs, we introduced a 'last resort' approach where the number of communications is reduced further by skipping some of the events. To achieve that, a special algorithm for a scheduler of NCS was presented that predicts future events if the control action is not updated. To keep the needed online computations as cheap as possible, some useful data structures can be computed offline, to speed up scheduler's tasks in real time. The algorithm also allows an easier scheduling of event-based PETC because it can estimate the time when the next event will happen.

One application of this kind of scheduler could be inspired by [33], where the NCS is modelled as a set of timed game automata (TGAs) and the scheduler can choose if the next update time should be based on a chosen ETM or occur earlier. Similarly, triggering at the last detected event could be another possibility of our scheduler. Depending on other control loops in the network and their expected triggering times, the scheduler could decide that, if there is a chance for communication conflict, one loop can wait until the next event. This way, a communication conflict could be avoided and all the loops would maintain stability without the necessity of triggering earlier.

For both versions of the relaxed PETC, several modifications were presented that aim at improving the performance of the closed loop system, by introducing trade-offs. Depending on the desired objective, one can reduce the oscillations of the system trajectories or include the disturbances in the scheduler's prediction at the expense of more transmissions. Finally,

we gave suggestions on when and how to apply the introduced modification and what types of systems might benefit from this kind of PETC implementations the most.

5-2 Recommendations for future work

As far as recommended future research is concerned, there are several aspects of the relaxed PETC that are open for improvement. First of all, better bounds could be found for GES and EISS guarantees of the relaxed PETC. Experimental results suggest that currently used bounds are conservative. Furthermore, in order to reduce the oscillations in system trajectories, other classes of Lyapunov functions could be studied.

Although the oscillations of state trajectories tend to be reduced for greater values of the imposed decay rate λ, the number of transmissions for the relaxed PETC with the scheduler seems to be larger for bigger λ. This poses another question, how to mitigate the contrary effects of increasing the decay rate. It might be a good idea to look into dynamic PETC and incorporate the idea of the buffer to the relaxed PETC. This could potentially prevent from triggering in consecutive sampling instants and resolve the issue of getting stuck in 'difficult states' (as in chapter 4) or in case of disturbances (as in chapter 2).

Scheduler's algorithm for predicting future events also has room for future work. When computing projections, the angular ranges could be found using a smarter approach to line search than fixed step. In addition, currently no way of computing projections of shapes different than cones has been presented and therefore this is another direction for further research.

Scheduler's predictions neglect the behaviour of the system until the last predicted event in a given time horizon. This poses yet another topic for future work, that focuses on incorporating the values of the triggering condition throughout the whole time horizon, without increasing the number of computations online. Lastly, including disturbances in the prediction results in numerical issues at times, what requires further investigation on the source of the problem and possible solutions.

Appendix A

Proofs

A-1 Proof of Theorem 2.1

Proof. Let us start by defining the monitored Lyapunov function $V(t) := x(t)^T P x(t)$, the exponentially decaying continuous-time bound $S(t_i + \tau) := V(t_i)e^{-\lambda\tau}$ and sampling times within one inter-event time $r_n = t_i + n\Delta$, $n \in \mathbb{N}_0$, $r_n \in [t_i, t_{i+1})$. At discrete sampling instants the bound given by our PETC implementation is equal to $S(t)$, namely $\eta(t_k)^T \eta(r_n) = S(r_n)$. Therefore, in the absence of disturbances, the monitored Lypaunov function satisfies $V(r_n) \leq S(r_n)$. The behavior is thus the same as the STC's from [25], and so is this proof, except for a small modification. To bound the evolution of the Lyapunov function in between the samples, let us start with finding the derivative $\dot{V}$ for the PETC system $\dot{x} = A^p x(t) + B^p K x(t_i)$. For $t \in [t_i, t_{i+1}]$, we have:

$$\dot{V}(t) = v(t)^T G v(t), \tag{A-1}$$

$$v(t) = \begin{bmatrix} \left(P^{\frac{1}{2}} x(t)\right)^T & \left(P^{\frac{1}{2}} x(t_i)\right)^T \end{bmatrix}^T,$$

$$G = \begin{bmatrix} P^{\frac{1}{2}} A^p P^{-\frac{1}{2}} + (P^{\frac{1}{2}} A^p P^{-\frac{1}{2}})^T & P^{\frac{1}{2}} B^p K P^{-\frac{1}{2}} \\ (P^{\frac{1}{2}} B^p K P^{-\frac{1}{2}})^T & 0 \end{bmatrix}. \tag{A-2}$$

Matrix G is symmetric and hence orthogonally diagonalizable. Furthermore, it holds that $\mu = \lambda_m(G) < 0$ and $\omega = \lambda_M(G) > 0$. To prove it, let us write $G = \begin{bmatrix} H & D \\ D^T & 0 \end{bmatrix}$, with $D \neq 0$. According to the Schur complement condition, $G \succ 0$ if and only if $H \succ 0$ and $G/H = -D^T H^{-1} D \succ 0$. If we assume that $H \succ 0$, then also $H^{-1} \succ 0$. If so, there exists a non-singular matrix M such that $H^{-1} = MM^T$. We can rewrite $D^T H^{-1} D = D^T M M^T D = (M^T D)^T M^T D = N^T N$, which implies (semi-)positive definiteness for any $N = M^T D$. That gives us $H \succ 0$ and $G/H \preceq 0$, so G is not (semi-)positive definite. Similarly, we can show that G is not (semi-)negative definite (by applying the same reasoning for $-G$). Hence, G must have at least one positive and at least one negative eigenvalue.

Since $v(t)^T v(t) = V(t) + V(t_i)$, (A-1) can be used to lower and upper bound the derivative of the monitored Lyapunov function.

$$\mu(V(t) + V(t_i)) \leq \dot{V}(t) \leq \omega(V(t) + V(t_i)), \quad t \in [t_i, t_{i+1}). \tag{A-3}$$

By the example of the right hand side of inequality (A-3), it will be shown how to integrate it in order to obtain a bound on $V(t)$. Based on the exact solution for LTI systems, for $t + s \in [t_i, t_{i+1})$:

$$V(t+s) \leq e^{\omega s} V(t) + \int_t^{t+s} e^{\omega(t+s-\eta)} d\eta \, \omega V(t_i)$$

With the change of variables and limits as follows:

$$\begin{aligned} r &= t + s - \eta, \\ dr &= -d\eta, \\ \eta &= t \rightarrow r = s, \\ \eta &= t + s \rightarrow r = 0 \end{aligned}$$

the solution is

$$\begin{aligned} V(t+s) \leq e^{\omega s} V(t) - \int_s^0 e^{\omega r} dr \omega V(t_i) = e^{\omega s} V(t) + \int_0^s e^{\omega r} dr \omega V(t_i) = \\ e^{\omega s} V(t) + \frac{1}{\omega}(e^{\omega s} - e^{\omega 0}) \omega V(t_i) = e^{\omega s} V(t) + (e^{\omega s} - 1) V(t_i). \end{aligned}$$

Following the same steps for the left hand side of (A-3), we can bound $V(t)$ itself by:

$$V(t+s) \leq e^{\omega s} V(t) + V(t_i)(e^{\omega s} - 1), \tag{A-4a}$$

$$V(t+s) \geq e^{\mu s} V(t) + V(t_i)(e^{\mu s} - 1), \tag{A-4b}$$

for $t + s \in [t_i, t_{i+1})$. We know the values of the Lypaunov function at sampling instants so we use them to obtain the bounds on $V(r_n + s)$, when $s \in [0, \Delta)$. From the upper bound (A-4a), we have

$$V(r_n + s) \leq e^{\omega s} V(r_n) + V(t_i)(e^{\omega s} - 1).$$

The lower bound (A-4b) can be transformed as follows:

$$\begin{aligned} V(r_n + s) &\geq e^{\mu s} V(r_n) + V(t_i)(e^{\mu s} - 1) \quad / e^{-\mu s} \\ V(r_n + s) e^{-\mu s} &\geq V(r_n) + V(t_i)(1 - e^{-\mu s}) \\ V(r_n) &\leq e^{-\mu s} V(r_n + s) + V(t_i)(e^{-\mu s} - 1) \\ V(r_n + h) &\leq e^{-\mu s} V(r_n + h + s) + V(t_i)(e^{-\mu s} - 1) \quad / s = \Delta - h \\ V(r_n + h) &\leq e^{\mu(h-\Delta)} V(r_n + \Delta) + V(t_i)(e^{\mu(h-\Delta)} - 1) \end{aligned}$$

To have the same variable, we substitute back $h = s$:

$$V(r_n + s) \leq e^{\mu(s-\Delta)} V(r_{n+1}) + V(t_i)(e^{\mu(s-\Delta)} - 1)$$

Putting the two bounds together, one obtains

$$V(r_n + s) \leq \begin{cases} e^{\omega s} V(r_n) + V(t_i)(e^{\omega s} - 1), & s \in [0, s^*], \\ e^{\mu(s-\Delta)} V(r_{n+1}) + V(t_i)(e^{\mu(s-\Delta)} - 1), & s \in [s^*, \Delta). \end{cases} \tag{A-5}$$

where s^* is the point where the two branches meet. Since the first bound is increasing and the second decreasing in s, $V(r_n + s^*)$ is the maximum of the Lyapunov function between r_n and r_{n+1}. We can find s^* by equating the two bounds, which results in:

$$s^* = \frac{1}{\omega - \mu} \log \left(\frac{V(r_{n+1}) + V(t_i)}{V(r_n) + V(t_i)} \right) + \frac{\mu \Delta}{\mu - \omega}$$

We substitute back the obtained expression for s^* to one of the bounds on $V(r_n + s)$ to obtain

$$V(r_n + s^*) \leq -V(t_i) + e^{\frac{\omega\mu\Delta}{\mu-\omega}} \left(\left(V(r_n) + V(t_i)\right)^{\frac{\mu}{\mu-\omega}} \left(V(r_{n+1}) + V(t_i)\right)^{\frac{-\omega}{\mu-\omega}} \right).$$

Using the fact that $V(r_n) \leq S(r_n)$ for all $r_n \in \mathbb{N}_0$ and dropping the first term, we have

$$V(r_n + s^*) \leq e^{\frac{\omega\mu\Delta}{\mu-\omega}} \left(\left(S(r_n) + S(t_i)\right)^{\frac{\mu}{\mu-\omega}} \left(S(r_{n+1}) + S(t_i)\right)^{\frac{-\omega}{\mu-\omega}} \right).$$

Because $S(r_n) = e^{\lambda s^*} S(r_n + s^*)$, $S(r_{n+1}) = e^{\lambda(s^* - \Delta)} S(r_n + s^*)$ and $S(t_i) = e^{\lambda(n\Delta + s^*)} S(r_n + s^*)$, and due to the fact that $\frac{\mu}{\mu-\omega} + \frac{-\omega}{\mu-\omega} = 1$, we can factor out $S(r_n + s*)$ to obtain $V(r_n + s^*) \leq \tilde{g}(\Delta, n) S(r_n + s^*)$ with

$$\tilde{g}(\Delta, n) = e^{\frac{\omega\mu\Delta}{\mu-\omega}} \left(e^{\lambda s^*} + e^{\lambda(n\Delta + s^*)}\right)^{\frac{\mu}{\mu-\omega}} \cdot \left(e^{\lambda(s^* - \Delta)} + e^{\lambda(n\Delta + s^*)}\right)^{\frac{-\omega}{\mu-\omega}}.$$

To make the above formula independent of time, we bound $s^* \leq \Delta$ and $n = N_{max} - 1$, since the formula is increasing in both parameters:

$$\tilde{g}(\Delta, N_{max} - 1) \leq e^{\frac{\omega\mu\Delta}{\mu-\omega}} \left(e^{\lambda\Delta} + e^{\lambda N_{max}\Delta}\right)^{\frac{\mu}{\mu-\omega}} \cdot \left(1 + e^{\lambda N_{max}\Delta}\right)^{\frac{-\omega}{\mu-\omega}} = g(\Delta, N_{max}). \tag{A-6}$$

$V(r_n + s^*) \leq g(\Delta, N_{max}) S(r_n + s^*)$ holds for all $n \in [0, N_{max}]$, so we can write

$$V(t_i + \tau) \leq g(\Delta, N_{max}) S(t_i + \tau),$$

$$V(t_i + \tau) \leq g(\Delta, N_{max}) V(t_i) e^{-\lambda\tau}.$$

Now we can use the fact that our PETC implementation yields $V(t_{i+1}) \leq V(t_i) e^{-\lambda\tau_i}$ to obtain

$$V(t_i + \tau) \leq g(\Delta, N_{max}) V(t_{i-1}) e^{-\lambda\tau_{i-1}} e^{-\lambda\tau} \leq g(\Delta, N_{max}) V(t_0) e^{-\lambda(\tau_0 + .. + \tau_{i-1}) + \tau} = g(\Delta, N_{max}) V(t_0) e^{-\lambda(t_i + \tau)} V(t) \leq g(\Delta, N_{max}) V(t_0) e^{-\lambda t},$$

where we substituted $t = t_i + \tau$. Finally, using bounds

$$\lambda_m(P) x(t)^T x(t) \leq V(t) \leq \lambda_M(P) x(t)^T x(t) \tag{A-7}$$

we arrive at

$$|x(t)| \leq \left(\frac{\lambda_M(P)}{\lambda_m(P)} \right)^{\frac{1}{2}} \left(g(\Delta, N_{max})\right)^{\frac{1}{2}} |x(t_0)| e^{-\frac{1}{2}\lambda t}. \tag{A-8}$$

Without loss of generality, we choose $t_0 = 0$. Then (A-8) proves GES with

$$\sigma = \left(\frac{\lambda_M(P)}{\lambda_m(P)} \right)^{\frac{1}{2}} \left(g(\Delta, N_{max})\right)^{\frac{1}{2}}, \quad \rho = \frac{1}{2}\lambda. \tag{A-9}$$

□

A-2 Proof of Theorem 2.2

Before showing the proof of Theorem 2.2, we introduce two lemmas.

Lemma A.1 ([25, Lemma A.1])**.** *Consider system (1-1) and a positive definite function* $\tilde{V}(x) = (x^T P x)^{\frac{1}{2}}$, $P \succ 0$. *For any given* $0 \leq T < \infty$ *the following bound holds:*

$$\tilde{V}(x_{x_0 u\delta}(t)) \leq \tilde{V}(x_{x_0 u0}(t)) + \gamma_{P,T}(||\delta||_\infty), \quad \forall t \in [0, T],$$
$$\gamma_{P,T}(s) \coloneqq s \frac{\lambda_M(P)}{\lambda_m^{\frac{1}{2}}(P)} \int_0^T |e^{A^p r} E| dr.$$

Lemma A.2. *The Lyapunov function* $\tilde{V}(x) = (x^T P x)^{\frac{1}{2}}$, $P \succ 0$ *of system (1-1) under the proposed PETC implementation, with* $\lambda < \lambda_0$, $\Delta < \tau_{min}$ *and* $\delta \in \mathcal{L}_\infty$, *satisfies*

$$\tilde{V}(t_{i+1}) \leq \tilde{V}(t_i) e^{-\frac{1}{2}\lambda \tau_i} + \gamma_{P,\Delta}(||\delta||_\infty).$$

Proof. From our PETC implementation, if the inter-event time is $\tau_i = t_{i+1} - t_i = n_i \Delta$, then for the case if $\delta = 0$ it holds that

$$V(x(r_n)) \leq V(x(t_i)) e^{-\lambda n \Delta},$$
$$\tilde{V}(x(r_n)) \leq \tilde{V}(x(t_i)) e^{-\frac{1}{2}\lambda n \Delta}, \tag{A-10}$$

for all $n = 0, ..., n_i$. When bounded disturbances are present (A-10) holds only for $n = 0, ..., n_{i-1}$, due to a possible worst-case scenario when $\zeta(n_i \Delta)^T Q_1 \zeta(n_i \Delta) > 0$. It corresponds to the situation when at time $r_{n_{i-1}}$ one-step-ahead prediction of the Lyapunov function would still be below the bound, but due to disturbances at r_{n_i} it exceeded the bound. Using Lemma A.1 we can write it as

$$\tilde{V}(x_{x(r_{n_{i-1}})u\delta}(r_{n_i})) \leq \tilde{V}(x_{x(r_{n_{i-1}})u0}(r_{n_i})) + \gamma_{P,\Delta}(||\delta||_\infty).$$

The left-hand side becomes $\tilde{V}(t_{i+1})$ and on the right-hand side we can use (A-10) to bound the prediction made without taking into account the disturbances. That concludes the proof. □

With that we can proceed to the proof of Theorem 2.2.

Proof. We start by iterating Lemma A.2.

$$\begin{aligned}
\tilde{V}(t_i) &\leq e^{-\frac{1}{2}\lambda(t_i - t_0)} \tilde{V}(t_0) + \gamma_{P,\Delta}(||\delta||_\infty) \sum_{k=0}^{i-1} e^{-\frac{1}{2}\lambda \tau_{min} k} \\
&\leq e^{-\frac{1}{2}\lambda(t_i - t_0)} \tilde{V}(t_0) + \gamma_{P,\Delta}(||\delta||_\infty) \sum_{k=0}^{\infty} e^{-\frac{1}{2}\lambda \Delta k} \\
&\leq e^{-\frac{1}{2}\lambda(t_i - t_0)} \tilde{V}(t_0) + \gamma_{P,\Delta}(||\delta||_\infty) \frac{1}{1 - e^{-\frac{1}{2}\lambda \Delta}},
\end{aligned}$$

where we used the fact that $\tau_{min} > \Delta$. Without loss of generality, we choose $t_0 = 0$:

$$\tilde{V}(t_i) \leq \tilde{V}(0) e^{-\frac{1}{2}\lambda t_i} + \frac{\gamma_{P,\Delta}(||\delta||_\infty)}{1 - e^{-\frac{1}{2}\lambda \Delta}}. \tag{A-11}$$

From GES of the unperturbed case, we have

$$\tilde{V}(x_{x(r_n)u0}(r_n+s)) \leq g(\Delta, N_{max})^{\frac{1}{2}} \tilde{V}(t_i) e^{-\frac{1}{2}\lambda(n\Delta+s)}. \tag{A-12}$$

for all $s \in [0, \Delta]$. In PETC we make only a one-step-ahead prediction and use the true value of the Lyapunov function at r_n to predict r_{n+1}. After the triggering condition is evaluated the prediction is discarded. Using Lemma A.1 and choosing $x(r_n)$ as the initial condition for predicting the state at $r_n + s$, we can write:

$$\tilde{V}(x_{x(r_n)u\delta}(r_n+s)) \leq \tilde{V}(x_{x(r_n)u0}(r_n+s)) + \gamma_{P,\Delta}(||\delta||_\infty), \tag{A-13}$$

for all $s \in [0, \Delta]$. After substituting (A-12) in (A-13) we obtain

$$\tilde{V}(x_{x(r_n)u\delta}(r_n+s)) \leq g(\Delta, N_{max})^{\frac{1}{2}} \tilde{V}(t_i) e^{-\frac{1}{2}\lambda(n\Delta+s)} + \gamma_{P,\Delta}(||\delta||_\infty), \quad \forall s \in [0, \Delta].$$

The above holds for all $n \in [0, ..., N_{max}]$ which, with $\tau = n\Delta + s$, leads to

$$\tilde{V}(x(t_i+\tau)) \leq g(\Delta, N_{max})^{\frac{1}{2}} \tilde{V}(t_i) e^{-\frac{1}{2}\lambda\tau} + \gamma_{P,\Delta}(||\delta||_\infty), \tag{A-14}$$

where the extensive subscripts are dropped, since everything involves disturbance. Finally, we substitute (A-11) to (A-14), arriving at

$$\tilde{V}(x(t_i+\tau)) \leq g(\Delta, N_{max})^{\frac{1}{2}} \tilde{V}(t_0) e^{-\frac{1}{2}\lambda(t_i+\tau)} + \gamma_{P,\Delta}(||\delta||_\infty)\left(\frac{g(\Delta, N_{max})^{\frac{1}{2}}}{1-e^{-\frac{1}{2}\lambda\Delta}} e^{-\frac{1}{2}\lambda\tau} + 1\right).$$

Replacing $t_i + \tau = t$ and bounding $e^{-\frac{1}{2}\lambda\tau} \leq 1$, we get

$$\tilde{V}(x(t)) \leq g(\Delta, N_{max})^{\frac{1}{2}} \tilde{V}(t_0) e^{-\frac{1}{2}\lambda t} + \gamma_{P,\Delta}(||\delta||_\infty)\left(\frac{g(\Delta, N_{max})^{\frac{1}{2}}}{1-e^{-\frac{1}{2}\lambda\Delta}} + 1\right).$$

Using the following bound:

$$\lambda_m^{\frac{1}{2}}(P)|x| \leq \tilde{V}(x) \leq \lambda_M^{\frac{1}{2}}(P)|x|,$$

one can bound the evolution of states as follows:

$$|x(t)| \leq \sigma|x(0)|e^{-\frac{1}{2}\lambda t} + \lambda_m^{-\frac{1}{2}}(P)\gamma_{P,\Delta}(||\delta||_\infty)\left(\frac{g(\Delta, N_{max})^{\frac{1}{2}}}{1-e^{-\frac{1}{2}\lambda\Delta}} + 1\right),$$

which proves that the system is EISS. □

A-3 Proof of Theorem 4.1

Proof. We want to bound the Lyapunov function $V(t)$ by an exponentially decaying function, for which we can use derived before bounds (A-4a) and (A-4b). As a result, in the interval $t_i + s \in [t_i, t_{i+1})$, we can bound

$$V(t_i+s) \leq \begin{cases} (2e^{\omega s} - 1)V(t_i), & s \in [0, s^*], \\ e^{\mu(s-\tau_i)}V(t_{n+1}) + V(t_i)(e^{\mu(s-\tau_i)} - 1), & s \in [s^*, \tau_i), \end{cases} \tag{A-15}$$

where s^* is the point where the two bounds meet. It is also the maximum for the bound of $V(t_i + s)$ for $s \in [0, \tau_i)$. Intuitively, we can think of it as having a control loop discretized with sampling time τ_i, because the scheduler does not know anything about the evolution of the Lyapunov function in between update times t_i and t_{i+1}. Therefore in (A-5) we substitute $r_n \leftarrow t_i$, $r_{n+1} \leftarrow t_{i+1}$ and $\Delta \leftarrow \tau_i$ to get (A-15). To obtain the worst-case scenario and bounds that work for any case, we notice that $\tau_i \leq pk_{max}\Delta = \tau_{max}$. After equating bounds (A-15) and applying a few logarithm properties, we find that

$$s^* = \frac{1}{\omega - \mu}\left(\log\left(\frac{V(t_{i+1}) + V(t_i)}{2V(t_i)}\right)\right) + \frac{\mu\tau_{max}}{\mu - \omega}. \tag{A-16}$$

We can now substitute (A-16) into one of the bounds (A-15):

$$V(t_i + s^*) \leq e^{\frac{\omega\mu\tau_{max}}{\mu-\omega}}\left(\left(V(t_{i+1}) + V(t_i)\right)^{\frac{-\omega}{\mu-\omega}}\left(2V(t_i)\right)^{\frac{\mu}{\mu-\omega}}\right) - V(t_i)$$

By dropping the last term, we obtain an expression that is monotonically increasing on $V(t_{i+1})$ and $V(t_i)$. Having in mind that for a properly chosen $\Delta \leq \tau_{min}$, the PETC implementation in a disturbance-free case yields $V(t_{i+1}) \leq S(t_i + \tau_i)$ even with the use of the scheduler, we can write:

$$V(t_i + s^*) \leq e^{\frac{\omega\mu\tau_{max}}{\mu-\omega}}\left(\left(S(t_{i+1}) + S(t_i)\right)^{\frac{-\omega}{\mu-\omega}}\left(2S(t_i)\right)^{\frac{\mu}{\mu-\omega}}\right).$$

Using the fact that $S(t_i) = e^{\lambda s^*}S(t_i + s^*)$ and $S(t_{i+1}) = e^{\lambda(s^* - \tau_{max})}S(t_i + s^*)$, the bound can be transformed to

$$V(t_i + s^*) \leq e^{\frac{\omega\mu\tau_{max}}{\mu-\omega}}\left(\left(e^{\lambda(s^*-\tau_{max})} + e^{\lambda s^*}\right)^{\frac{-\omega}{\mu-\omega}}\left(2e^{\lambda s^*}\right)^{\frac{\mu}{\mu-\omega}}\right)S(t_i + s^*). \tag{A-17}$$

Because $V(t_{i+1})$ is always smaller than $V(t_i)$ when no disturbance is present, $\log\left(\frac{V(t_{i+1})+V(t_i)}{2V(t_i)}\right) \leq 0$ and (A-16) can be ultimately bounded with:

$$s^* \leq \frac{\mu\tau_{max}}{\mu - \omega}.$$

With that, the bound (A-17) can be written in the form $V(t_i + s^*) \leq g_s(\tau_{max})S(t_i + s^*)$ with

$$g_s(\tau_{max}) = e^{\frac{\omega\mu\tau_{max}}{\mu-\omega}}\left(\left(e^{\frac{\lambda\omega\tau_{max}}{\mu-\omega}} + e^{\frac{\lambda\mu\tau_{max}}{\mu-\omega}}\right)^{\frac{-\omega}{\mu-\omega}}\left(2e^{\frac{\lambda\mu\tau_{max}}{\mu-\omega}}\right)^{\frac{\mu}{\mu-\omega}}\right). \tag{A-18}$$

Since $S(t_i + s^*) = V(t_i)e^{-\lambda s^*}$, we have

$$V(t_i + s^*) \leq g_s(\tau_{max})V(t_i)e^{-\lambda s^*}.$$

Substituting iteratively $V(t_{i+1}) \leq V(t_i)e^{-\lambda\tau_i}$ to the above inequality, the bound becomes

$$V(t) \leq g_s(\tau_{max})V(0)e^{-\lambda t}$$

where we substituted $t = t_i + s^*$ and $t_0 = 0$. Finally, using (A-7), we arrive at

$$|x(t)| \leq \left(\frac{\lambda_M(P)}{\lambda_m(P)}\right)^{\frac{1}{2}}(g_s(\tau_{max}))^{\frac{1}{2}}|x(0)|e^{-\frac{1}{2}\lambda t},$$

which concludes the proof.

□

A-4 Proof of Theorem 4.2

Proof. Since the end result of implementing a scheduler is suppressing communication between the plant and the controller similarly to STC, the proof is analogous to the one presented in [25]. From Lemma A.1, we have

$$\tilde{V}(x_{x_0u\delta}(t_{i+1})) \leq \tilde{V}(x_{x_0u0}(t_{i+1})) + \gamma_{P,\tau_i}(||\delta||_\infty),$$

and from the PETC implementation with the scheduler:

$$V(x_{x_0u0}(t_{i+1})) \leq V(x_{x_0u0}(t_i))e^{-\lambda\tau_i},$$
$$\tilde{V}(x_{x_0u0}(t_{i+1})) \leq \tilde{V}(x_{x_0u0}(t_i))e^{-\frac{1}{2}\lambda\tau_i},$$

for $i \in \mathbb{N}_0$, which can be together combined into:

$$\tilde{V}(t_{i+1}) \leq \tilde{V}(t_i)e^{-\frac{1}{2}\lambda\tau_i} + \gamma_{P,\tau_i}(||\delta||_\infty).$$

We iterate this inequality, similarly to the beginning of the proof of Theorem 2.2, to arrive at

$$\tilde{V}(t_i) \leq \tilde{V}(0)e^{-\frac{1}{2}\lambda t_i} + \frac{\gamma_{P,\tau_{max}}(||\delta||_\infty)}{1 - e^{-\frac{1}{2}\lambda\Delta}} \tag{A-19}$$

where we used the fact that $\gamma_{P,\tau_i} \leq \gamma_{P,\tau_{max}}$. To find a bound that is valid not only at discrete time instants t_i, we use GES property from Theorem 4.1:

$$\tilde{V}(t_i + \tau) \leq (g_s(\tau_{max}))^{\frac{1}{2}}\tilde{V}(t_i)e^{-\frac{1}{2}\lambda\tau}, \quad \forall\tau \in [0, \tau_{max}].$$

This expression can replace the disturbance-free term in Lemma A.1 written for time instant $t_i + \tau$ as follows:

$$\tilde{V}(x_{x_0u\delta}(t_i + \tau)) \leq (g_s(\tau_{max}))^{\frac{1}{2}}\tilde{V}(t_i)e^{-\frac{1}{2}\lambda\tau} + \gamma_{P,\tau_{max}}(||\delta||_\infty)$$

The final steps to obtain a bound on $\tilde{V}$ in a perturbed case is to substitute (A-19) into the previous inequality, set $t_i + \tau = t$ and use the fact that $e^{-\frac{1}{2}\lambda\tau} \leq 1$:

$$\tilde{V}(t) \leq (g_s(\tau_{max}))^{\frac{1}{2}}\tilde{V}(0)e^{-\frac{1}{2}\lambda t} + \gamma_{P,\tau_{max}}(||\delta||_\infty)\left(\frac{(g_s(\tau_{max}))^{\frac{1}{2}}}{1 - e^{-\frac{1}{2}\lambda\Delta}} + 1\right).$$

Lastly, we use bounds (A-7) to bound the norm of system's trajectories

$$|x(t)| \leq \left(\frac{\lambda_M(P)}{\lambda_m(P)}\right)^{\frac{1}{2}}(g_s(\tau_{max}))^{\frac{1}{2}}|x_0|e^{-\frac{1}{2}\lambda t} + \gamma_s(||\delta||_\infty)$$

with $\gamma_s(||\delta||_\infty)$ as in Theorem 4.2. □

Appendix B

Predecessors and projections

This appendix contains sets of predecessors and angular regions of projections computed for systems used in chapter 3 and 4.

B-1 2D system

B-1-1 Predecessors

A set of predecessors for every Q_k in a form $k : set$. {} denotes an empty set. The sets were computed as described in section 3-2-3.

1-6: {},
7: {6},
8: {6},
9: {6},
10: {6},
11: {6},
12: {6},
13: {6, 12},
14: {6, 12, 13},
15: {6, 12, 13},
16: {6, 12, 13},
17: {6, 12, 13},
18: {6, 12, 13},
19: {6, 12, 13, 18},
20: {6, 12, 13, 18, 19}.

B-1-2 Ranges

Angular regions in radians for θ_1 resulting from the projections method described in section 3-2-2 for every matrix Q_k. The notation is $k : [\underline{\theta}^*_{1,k}, \bar{\theta}^*_{1,k}]$.

1: [1.0367255756846323, 2.419026343264145],
2: [0.5026548245743668, 2.4818581963359403],
3: [0.8796459430051425, 2.5761059759436336],
4: [1.539380400259, 3.141592653589793],
5: [0, 3.141592653589793],
6: [0, 4.385380947269368e−15],
7: [1.38230076757951, 1.6650441064025963],
8: [0.12566370614359174, 2.6075219024795313],
9: [0.31415926535897926, 2.7017696820872246],
10: [0, 3.141592653589793],
11: [0.15707963267948966, 1.2880529879718206],
12: [0, 4.385380947269368e−15],
13: [0, 4.385380947269368e−15],
14: [0.031415926535897934, 1.2252211349000246],
15: [2.70176968208722, 3.1101767270538954],
16: [2.733185608623118, 3.1101767270538954],
17: [0.06283185307179587, 0.8168140899333509],
18: [0, 4.385380947269368e−15],
19: [0, 4.385380947269368e−15],
20: [1.0995574287564283, 1.4451326206513104].

B-2 5D system

B-2-1 Predecessors

For this 5D system most of the matrices Q_k do not have any predecessors. The only non-empty sets resulting from computations described in section 3-2-3 are:

1-33: {},
34: {33},
34: {33, 34},
35: {33, 34},
36: {33, 34},
37: {33, 34},
38: {33, 34},
39: {33, 34}.

B-2-2 Ranges

Angular regions in radians for $\theta_1 - \theta_5$ resulting from the projections method described in section 3-2-2. The notation is $k : \{[\underline{\theta}^*_{i,k}, \bar{\theta}^*_{i,k}], i \in \{1, .., 4\}\}$.

```
1-13: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
14: {[0, 3.141592653589793],
    [0.2199114857512855, 1.8221237390820861],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
15: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
16: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.0787608005179976]},
17: {[0.43982297150257094, 0.5654866776461671],
    [0, 3.141592653589793],
    [1.288052987971816, 1.6650441064025963],
    [0, 3.141592653589793]},
18: {[0.47123889803846886, 0.691150383789759],
    [0, 3.141592653589793],
    [1.2566370614359181, 1.8535396656179841],
    [3.0473448739820954, 3.141592653589793]},
19: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
20: {[0.2199114857512855, 0.5969026041820651],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
21: {[0.5026548245743668, 0.5969026041820651],
    [0, 3.141592653589793],
    [1.0053096491487343, 1.6964600329384942],
    [0, 3.141592653589793]},
22: {[0.5026548245743668, 0.6283185307179631],
    [0, 3.141592653589793],
    [1.2566370614359181, 1.7592918860102902],
    [0, 3.141592653589793]},
23: {[0.5026548245743668, 0.6283185307179631],
    [0, 3.141592653589793],
    [1.350884841043612, 1.8221237390820861],
    [2.9845130209103, 3.141592653589793]},
24: {[0.43982297150257094, 0.5654866776461671],
    [0, 3.141592653589793],
    [1.445132620651306, 1.91637151868978],
    [2.953097094374402, 3.1101767270538954]},
25: {[0.06283185307179587, 0.5026548245743712],
    [0, 3.141592653589793],
    [1.507964473723102, 2.0420352248333717],
    [2.8902652413026066, 3.0787608005179976]},
```

```
26: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [1.476548547187204, 2.3561944901923493],
    [2.7646015351590156, 3.0787608005179976]},
27: {[0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
28: {[0.3769911184307751, 0.691150383789759],
    [0, 3.141592653589793],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
29: {[0.5654866776461628, 0.6283185307179631],
    [0.8796459430051425, 1.1309733552923307],
    [0.408407044966673, 0.9424777960769428],
    [0, 3.141592653589793]},
30: {[0.5654866776461628, 0.659734457253861],
    [0.7539822368615506, 1.1938052083641266],
    [0.2199114857512855, 1.0681415022205347],
    [0, 0.31415926535898364]},
31: {[0.5969026041820608, 0.8168140899333509],
    [0.7225663103256527, 1.2880529879718206],
    [0, 3.141592653589793],
    [0, 3.141592653589793]},
32: {[0.3455751918948772, 0.4398229715025753],
    [2.1362830044410606, 2.9530970943744066],
    [1.6022122533307959, 1.8535396656179841],
    [0, 3.141592653589793]},
33: {[0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15]},
34: {[0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15]},
35: {[0, 3.141592653589793],
    [1.2566370614359181, 1.6336281798666983],
    [0, 3.141592653589793],
    [0.31415926535897926, 2.8588493147667133]},
36: {[0.43982297150257094, 0.5026548245743712],
    [0.47123889803846886, 0.8482300164692489],
    [1.1623892818282242, 1.4137166941154125],
    [0, 3.141592653589793]},
37: {[0.47123889803846886, 0.5340707511102691],
    [0.0942477796076938, 0.4712388980384732 4],
    [1.413716694115408, 1.5393804002590044],
    [3.078760800517993, 3.141592653589793]},
38: {[0.5026548245743668, 0.5654866776461671],
    [0, 3.141592653589793],
    [1.507964473723102, 1.6336281798666983],
    [3.0473448739820954, 3.1101767270538954]},
39: {[0.5026548245743668, 0.5654866776461671],
```

```
    [2.6703537555513224, 2.890265241302611],
    [1.6336281798666938, 1.6964600329384942],
    [3.0159289474461977, 3.0473448739821]},
40: {[0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15],
    [0, 4.385380947269368e-15]}.
```

Bibliography

[1] W. P. M. H. Heemels, M. C. F. Donkers, and A. R. Teel, "Periodic event-triggered control for linear systems," in *IEEE Transactions on Automatic Control*, vol. 58, pp. 847–861, Apr. 2013.

[2] D. P. Borgers, V. S. Dolk, and W. P. M. H. Heemels, "Dynamic periodic event-triggered control for linear systems," in *HSCC*, 2017.

[3] C. Fiter, L. Hetel, W. Perruquetti, and J.-P. Richard, "A state dependent sampling for linear state feedback," in *Automatica, Elsevier*, vol. 48, pp. 1860–1867, 2012.

[4] P. Seiler and R. Sengupta, "Analysis of communication losses in vehicle control problems," in *Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148)*, vol. 2, pp. 1491–1496 vol.2, June 2001.

[5] Cai Meng, Tianmiao Wang, Wusheng Chou, Sheng Luan, Yuru Zhang, and Zengmin Tian, "Remote surgery case: robot-assisted teleneurosurgery," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, pp. 819–823 Vol.1, April 2004.

[6] J. Liu, A. Gusrialdi, D. Obradovic, and S. Hirche, "Study on the effect of time delay on the performance of distributed power grids with networked cooperative control*," *IFAC Proceedings Volumes*, vol. 42, no. 20, pp. 168 – 173, 2009. 1st IFAC Workshop on Estimation and Control of Networked Systems.

[7] D. Lehmann and J. Lunze, "Extension and experimental evaluation of an event-based state-feedback approach," *Control Engineering Practice*, vol. 19, no. 2, pp. 101 – 112, 2011.

[8] W. P. M. H. Heemels, M. C. F. Donkers, and A. R. Teel, "Problems in event based engine control," in *Proc. Amer. Control Conf.*, vol. 2, pp. 1585–1587, 1994.

[9] K. E. Arzen, "A simple event-based pid controller," in *Preprints IFAC World Conf.*, vol. 18, pp. 423–428, 1999.

[10] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *IEEE Conference on Decision and Control*, pp. 3270–3285, Maui, Hawaii, USA, Dec. 2012.

[11] W. P. M. H. Heemels, M. C. F. Donkers, and A. R. Teel, "Periodic event-triggered control based on state feedback," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pp. 2571–2576, Dec. 2011.

[12] V. S. Dolk, D. P. Borgers, and W. P. M. H. Heemels, "Output-based and decentralized dynamic event-triggered control with guaranteed $\mathcal{L}_p$-gain performance and Zeno-freeness," in *IEEE Transactions on Automatic Control*, vol. 62, Jan. 2017.

[13] M. Velasco, P. Marti, and J. M. Fuertes, "The self triggered task model for real-time control systems," in *Proc. of 24th IEEE Real-Time Systems Symposium, Work-in-Progress Session*, 2003.

[14] G. de A. Gleizer and M. Mazo Jr., "Self-triggered output feedback control for perturbed linear systems," in *Preprints, 7th IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pp. 248–253, Aug. 2018.

[15] X. Wang and M. D. Lemmon, "Self-triggered feedback control systems with finite-gain $\mathcal{L}_\infty$ stability," in *IEEE Transactions on Automatic Control*, vol. 54, pp. 452–467, 2009.

[16] M. Lemmon, T. Chantem, X. S. Hu, and M. Zyskowski, "On self-triggered full-information h-infinity controllers," in *Hybrid Systems: Computation and Control* (A. Bemporad, A. Bicchi, and G. Buttazzo, eds.), (Berlin, Heidelberg), pp. 371–384, Springer Berlin Heidelberg, 2007.

[17] A. Eqtami, D. V. Dimarogonas, and K. J. Kyriakopoulos, "Event-triggered control for discrete-time systems," in *Proc. Amer. Control Conf.*, pp. 4719–4724, 2010.

[18] M. Velasco, P. Marti, and E. Bini, "On Lyapunov sampling for event-driven controllers," in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pp. 6238–6243, Dec. 2009.

[19] P. Tabuda, "Event-triggered real-time scheduling of stabilizing control tasks," in *IEEE Transactions on Automatic Control*, vol. 52, pp. 1680–1685, Sep. 2007.

[20] M. C. F. Donkers and W. P. M. H. Heemels, "Output-based event-triggered control with guaranteed $\mathcal{L}_\infty$-gain and improved and decentralized event-triggering," in *IEEE Transactions on Automatic Control*, vol. 57, pp. 1363–1376, Jun. 2012.

[21] A. Szymanek, *Literature Survey: Periodic event-triggered control: towards less conservative triggering condition.* Delft University of Technology, 2018.

[22] A. Szymanek, G. de A. Gleizer, and M. Mazo Jr., "Periodic event-triggered control with a relaxed triggering condition," *58th IEEE Conference on Decision and Control*, in press.

[23] X. Wang and M. D. Lemmon, "Event design in event-triggered feedback control systems," in *Proc. of the 47th IEEE Conference on Decision and Control*, pp. 2105–2110, Dec. 2008.

[24] A. Girard, "Dynamic triggering mechanisms for event-triggered control," in *IEEE Transactions on Automatic Control*, vol. 60, pp. 1992–1997, July 2015.

[25] M. Mazo Jr., A. Anta, and P. Tabuada, "An ISS self-triggered implementation of linear controllers," in *Automatica*, vol. 46, pp. 1310–1314, 2010.

[26] A. S. Kolarijani and M. Mazo Jr., "Formal traffic characterization of lti event-triggered control systems," in *IEEE Transactions on Control of Networked Systems*, vol. 5, pp. 274–283, Mar. 2018.

[27] M. Mazo Jr. and P. Tabuada, "Input-to-state stability of self-triggered control systems," in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pp. 928–933, Dec. 2009.

[28] M. Mazo Jr., A. Sharifi-Kolarijani, D. Adzkiya, and C. Hop, *Abstracted Models for Scheduling of Event-Triggered Control Data Traffic*, pp. 197–217. Cham: Springer International Publishing, 2018.

[29] F. G. M.A. Aizerman, *Absolute Stability of Regulator Systems*. Holden-Day, 1964.

[30] G. de A. Gleizer and M. Mazo Jr., "Self-triggered output-feedback control of LTI systems subject to disturbances and noise," *Automatica*, in press.

[31] A. Kurzhanski and I. Valyi, *Ellipsoidal Calculus for Estimation and Control*. Birkhäuser, 1997.

[32] A. Kurzhanski and I. Valyi, *Ellipsoidal Toolbox, Technical Report*. 2006.

[33] D. Adzkiya and M. Mazo Jr., "Scheduling of event-triggered networked control systems using timed game automata," *CoRR*, vol. abs/1610.03729, 2016.

Glossary

List of Acronyms

CETC	continuous event-triggered control
EISS	exponential input-to-state stability
ETC	event-triggered control
ETM	event-triggered mechanism
LMI	linear matrix inequality
LTI	linear time-invariant
NCS	networked control system
PETC	periodic event-triggered control
STC	self-triggered control
SDP	semi-definite programming
TGA	timed game automaton

List of Symbols

Δ	Sampling interval
δ	Disturbance signal
$\hat{u}$	Control input applied to the plant
$\hat{x}$	State measurement available to the controller
λ	Desired decay rate of the Lyapunov function
λ_0	Decay rate of the Lyapunov function for the continuous-time closed-loop system
$\mathcal{C}$	Triggering condition

$\mathcal{R}_s$	Conic region
$\mathcal{U}$	Set of possible control inputs
$\mathcal{W}$	Set of possible disturbances
ρ	Convergence rate of the system
τ_i	Inter-event time
τ_{min}	Minimum inter-event time
θ_i	Angular coordinate
$\{t_i\}_{i\in\mathbb{N}}$	Triggering times
$\{t_k\}_{k\in\mathbb{N}}$	Periodic sampling times
p	Sub-sampling factor
x	Current state

Index