# Event-triggered and Self-triggered Control

W.P.M.H. Heemels K.H. Johansson P. Tabuada

***Abstract*— Recent developments in computer and communication technologies have led to a new type of large-scale resource-constrained wireless embedded control systems. It is desirable in these systems to limit the sensor and control computation and/or communication to instances when the system needs attention. However, classical sampled-data control is based on performing sensing and actuation periodically rather than when the system needs attention. This article discusses event- and self-triggered control systems where sensing and actuation is performed when needed. Event-triggered control is reactive and generates sensor sampling and control actuation when, for instance, the plant state deviates more than a certain threshold from a desired value. Self-triggered control, on the other hand, is proactive and computes the next sampling or actuation instance ahead of time. The basics of these control strategies are introduced together with references for further reading.**

***Index Terms*— Real-time control, event-triggered control, self-triggered control, resource-constrained embedded control, hybrid systems, sampled-data systems.**

## I. Introduction

In standard control textbooks, *e.g.* [Aström & Wittenmark, 1997], [Franklin et al., 2010], periodic control is presented as the only choice for implementing feedback control laws on digital platforms. Although this time-triggered control paradigm has proven to be extremely successful in many digital control applications, recent developments in computer and communication technologies have led to a new type of large-scale resource-constrained wireless embedded control systems that call for a reconsideration of this traditional paradigm. In particular, the increasing popularity of (shared) wired and wireless networked control systems raises the importance of explicitly addressing energy, computation, and communication constraints when designing feedback control loops. Aperiodic control strategies that allow the inter-execution times of control tasks to be varying in time offer potential advantages with respect to periodic control when handling these constraints but they also introduce many new interesting theoretical and practical challenges.

Although the discussions regarding periodic vs aperiodic implementation of feedback control loops date back to the beginning of computer-controlled systems, *e.g.* [Gupta, 1963], in the late 90s two influential papers [Aström & Bernhardsson, 1999], [Arzén, 1999] highlighted the advantages of *event-based* feedback control. These two papers spurred the development of the first *systematic designs* of event-based implementations of stabilizing feedback control laws, *e.g.* [Yook et al., 2002], [Tabuada, 2007], [Heemels et al., 2008], [Henningsson et al., 2008]. Since then, several researchers have improved and generalized these results and alternative approaches have appeared. In the meantime, also so-called *self-triggered* control [Velasco et al., 2003] emerged. Event-triggered and self-triggered control systems consist of two elements, namely, a feedback controller that computes the control input, and a triggering mechanism that determines when the control input has to be updated again. The difference between event-triggered control and self-triggered control is that the former is reactive, while the latter is proactive. Indeed, in event-triggered control a triggering condition based on current measurements is continuously monitored and when the condition holds, an event is triggered. In self-triggered control the next update time is precomputed at a control update time based on predictions using previously received data and knowledge of the plant dynamics. In some cases, it is advantageous to combine event-triggered and self-triggered control resulting in a control system reactive to unpredictable disturbances and proactive by predicting future use of resources.

## II. Time-triggered, Event-triggered and Self-triggered Control

To indicate the differences between digital implementations of feedback control laws, consider the control of the nonlinear plant

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

with $x \in \mathbb{R}^{n_x}$ the state variable and $u \in \mathbb{R}^{n_u}$ the input variable. The system is controlled by a nonlinear state feedback law

$$u = h(x) \tag{2}$$

where $h : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ is an appropriate mapping, that has to be implemented on a digital platform. Recomputing the control value and updating the actuator signals will occur at times denoted by $t_0, t_1, t_2, \ldots$ with $t_0 = 0$. If we assume

Maurice Heemels is with the Control Systems Technology group, Department of Mechanical Engineering, Eindhoven Univeristy of Technology, the Netherlands; Karl H. Johansson is with ACCESS Linnaeus Center, Royal Institute of Technology, Sweden; Paulo Tabuada is with Department of Electrical Engineering, University of California, Los Angeles, CA, USA. E-mails: m.heemels@tue.nl,kallej@kth.se,tabuada@ee.ucla.edu

The work of Maurice Heemels was partially supported by the Dutch Science Foundation (STW) and the Dutch Organization for Scientific Research (NWO) under the VICI grant "Wireless controls systems: A new frontier in automation". The work of Karl Johansson was partially supported by the Knut and Alice Wallenberg Foundation and the Swedish Research Council. Maurice Heemels and Karl Johansson were also supported by the European 7th Framework Programme Network of Excellence under grant HYCON2-257462. The work of Paulo Tabuada was partially supported by NSF awards 0834771 and 0953994.

the inputs to be held constant in between the successive re-computations of the control law (referred to as sample-and-hold or zero-order-hold), we have

$$u(t) = u(t_k) = h(x(t_k)) \quad \forall t \in [t_k, t_{k+1}),\ k \in \mathbb{N}. \tag{3}$$

We refer to the instants $\{t_k\}_{k\in\mathbb{N}}$ as the *triggering* times or *execution* times. Based on these times we can easily explain the difference between time-triggered control, event-triggered control and self-triggered control.

In *time-triggered* control we have the equality $t_k = kT_s$ with $T_s > 0$ being the sampling period. Hence, the updates take place equidistantly in time irrespective of how the system behaves. There is no "feedback mechanism" in determining the execution times; they are determined a priori and in "open loop." Another way of writing the triggering mechanism in time-triggered control is

$$t_{k+1} = t_k + T_s,\ k \in \mathbb{N} \tag{4}$$

with $t_0 = 0$.

In *event-triggered* control the next execution time of the controller is determined by an event-triggering mechanism that continuously verifies if a certain condition based on the actual state variable becomes true. This condition includes often information on the state variable $x(t_k)$ at the previous execution time $t_k$ and can be written, for instance, as $C(x(t), x(t_k)) > 0$. Formally, the execution times are then determined by

$$t_{k+1} = \inf\{t > t_k \mid C(x(t), x(t_k)) > 0\} \tag{5}$$

with $t_0 = 0$. Hence, it is clear from (5) that there is a *feedback* mechanism present in the determination of the next execution time as it is based on the measured state variable. In this sense event-triggered control is *reactive*.

Finally, in *self-triggered* control the next execution time is determined *proactively* based on the measured state $x(t_k)$ at the previous execution time. In particular, there is a function $M : \mathbb{R}^{n_x} \to \mathbb{R}_{\geq 0}$ that specifies the next execution time as

$$t_{k+1} = t_k + M(x(t_k)) \tag{6}$$

with $t_0 = 0$. As a consequence, in self-triggered control both the control value $u(t_k)$ as well as the next execution time $t_{k+1}$ are computed at execution time $t_k$. In between $t_k$ and $t_{k+1}$ no further actions are required from the controller. Note that the time-triggered implementation can be seen as a special case of the self-triggered implementation by taking $M(x) = T_s$ for all $x \in \mathbb{R}^{n_x}$.

Clearly, in all the three implementation schemes $T_s$, $C$ and $M$ are chosen together with the feedback law given through $h$ to provide stability and performance guarantees, and to realize a certain utilization of computer and communication resources.

## III. Lyapunov-based analysis

Much work on event-triggered control used one of the following two modeling and analysis frameworks: The perturbation approach and the hybrid system approach.

### A. Perturbation approach

In the perturbation approach one adopts perturbed models that describe how the event-triggered implementation of the control law perturbs the ideal continuous-time implementation $u(t) = h(x(t))$, $t \in \mathbb{R}_{\geq 0}$. In order to do so, consider the error $e$ given by

$$e(t) = x(t_k) - x(t) \quad \text{for } t \in [t_k, t_{k+1}),\ k \in \mathbb{N}. \tag{7}$$

Using this error variable we can write the closed-loop system based on (1) and (3) as

$$\dot{x} = f(x, h(x + e)). \tag{8}$$

Essentially, the three implementations discussed above have their own way of indicating when an execution takes place and $e$ is reset to zero. The equation (8) clearly shows how the ideal closed-loop system is *perturbed* by using a time-triggered, event-triggered or self-triggered implementation of the feedback law (2). Indeed, when $e = 0$ we obtain the ideal closed loop

$$\dot{x} = f(x, h(x)). \tag{9}$$

The control law (2) is chosen so as to guarantee that (9) has certain global asymptotic stability (GAS) properties for event-triggered implementations. In particular, it is often assumed that there exists a Lyapunov function $V : \mathbb{R}_{n_x} \to \mathbb{R}_{\geq 0}$ in the sense that $V$ is positive definite and for all $x \in \mathbb{R}^{n_x}$ we have

$$\frac{\partial V}{\partial x} f(x, h(x)) \leq -\|x\|^2. \tag{10}$$

Note that this inequality is stronger than strictly needed (at least for nonlinear systems), but for pedagogical reasons we choose this simpler formulation. For the perturbed model, the inequality in (10) can in certain cases (including linear systems) be modified to

$$\frac{\partial V}{\partial x} f(x, h(x)) \leq -\|x\|^2 + \beta\|e\|^2 \tag{11}$$

in which $\beta > 0$ is a constant used to indicate how the presence of the implementation error $e$ affects the decrease of the Lyapunov function. Based on (10) one can now choose the function $C$ in (5) to preserve GAS of the event-triggered implementation. For instance, $C(x(t), x(t_k)) = \|x(t_k) - x(t)\| - \sigma\|x(t\|$, *i.e.*,

$$t_{k+1} = \inf\{t > t_k \mid \|e(t)\| > \sigma\|x(t)\|\}, \tag{12}$$

assures that

$$\|e\| \leq \sigma\|x\| \tag{13}$$

holds. When $\sigma < 1/\beta$, we obtain from (11) and (13) that GAS properties are preserved for the event-triggered implementation. Besides, under certain conditions provided in [Tabuada, 2007], a global positive lower bound exists on the inter-execution times, *i.e.*, there exists a $\tau_{\min} > 0$ such that $t_{k+1} - t_k > \tau_{\min}$ for all $k \in \mathbb{N}$ and all initial states $x_0$.

Also self-triggered controllers can be derived using the perturbation approach. In this case, stability properties can be guaranteed by choosing $M$ in (6) ensuring that $C(x(t), x(t_k)) \leq 0$ holds for all times $t \in [t_k, t_{k+1})$ and all $k \in \mathbb{N}$.

### B. Hybrid system approach

By taking as a state variable $\xi = (x, e)$ one can write the closed-loop event-triggered control system given by (1), (3), and (5) as the hybrid impulsive system [Goebel et al., 2009]

$$\dot{\xi} = \begin{pmatrix} f(x, h(x+e)) \\ -f(x, h(x+e)) \end{pmatrix} \text{ when } C(x, x+e) \geq 0 \quad (14a)$$

$$\xi^+ = \begin{pmatrix} x \\ 0 \end{pmatrix} \text{ when } C(x, x+e) \leq 0. \quad (14b)$$

This observation was made in [Donkers & Heemels, 2010], [Postoyan et al., 2011], [Donkers & Heemels, 2012]. Tools from hybrid system theory can be used to analyze this model, which is more accurate as it includes the error dynamics of the event-triggered closed-loop system. In fact, the stability bounds obtained via the hybrid system approach can be proven to be never worse than ones obtained using the perturbation approach in many cases, *e.g.*, [Donkers & Heemels, 2012], and typically the hybrid system approach provides (strictly) better results in practice. However, in general an analysis via the hybrid system approach is more complicated to perform than using a perturbation approach.

Note that by including a time variable $\tau$, one can also write the closed-loop system corresponding to self-triggered control (1), (3), and (6) as a hybrid system using the state variable $\chi = (x, e, \tau)$. This leads to the model

$$\dot{\chi} = \begin{pmatrix} f(x, h(x+e)) \\ -f(x, h(x+e)) \\ 1 \end{pmatrix} \text{ when } 0 \leq \tau \leq M(x+e) \quad (15a)$$

$$\chi^+ = \begin{pmatrix} x \\ 0 \\ 0 \end{pmatrix} \text{ when } \tau = M(x+e), \quad (15b)$$

which can be used for analysis based on hybrid tools as well.

## IV. Alternative event-triggering mechanisms

There are various alternative event-triggering mechanisms. A few of them are described in this section.

### A. Relative, absolute and mixed triggering conditions

Above we discussed a very basic event-triggering condition in the form (12), which is sometimes called *relative* triggering as the next control task is executed at the instant when the ratio of the norms of the error $\|e\|$ and the measured state $\|x\|$ is larger than or equal to $\sigma$. Also *absolute* triggering of the form

$$t_{k+1} = \inf\{t > t_k \mid \|e(t)\| \geq \delta\}, \quad (16)$$

can be considered. Here $\delta > 0$ is an absolute threshold, which has given this scheme the name send-on-delta [Miskowicz, 2006]. Recently, a *mixed* triggering mechanism of the form

$$t_{k+1} = \inf\{t > t_k \mid \|e(t)\| \geq \sigma\|x(t)\| + \delta\}, \quad (17)$$

combining an absolute and a relative threshold, was proposed [Donkers & Heemels, 2012]. It is particularly effective in the context of output-based control.

### B. Model-based triggering

In the triggering conditions discussed so far essentially the current control value $u(t)$ is based on a *held* value $x(t_k)$ of the state variable, as specificed in (3). However, if good model-based information regarding the plant is available, one can use better model-based predictions of the actuator signal. For instance, in the linear context, [Lunze & Lehmann, 2010] proposed to use a *control input generator* instead of a plain zero-order hold function. In fact, the plant model was described by

$$\dot{x} = Ax + Bu + Ew \quad (18)$$

with $x \in \mathbb{R}^{n_x}$ the state variable, $u \in \mathbb{R}^{n_u}$ the input variable and $w \in \mathbb{R}^{n_w}$ a bounded disturbance input. It was assumed that a well functioning state feedback controller $u = Kx$ was available. The control input generator was then based on the model-based predictions on the time interval $[t_k, t_{k+1})$ given by

$$\dot{x}_s = (A + BK)x_s + E\hat{w}(t_k) \text{ with } x_s(t_k) = x(t_k) \quad (19)$$

and $\hat{w}(t_k)$ is an estimate for the (average) disturbance value, which is determined at execution time $t_k$, $k \in \mathbb{N}$. The applied input to the actuator is then given by $u(t) = Kx_s(t)$ for $t \in [t_k, t_{k+1})$, $k \in \mathbb{N}$. Note that (19) is a prediction of the closed-loop state evolution using the latest received value of the state $x(t_k)$ and the estimate $\hat{w}(t_k)$ of the disturbances. Also the event-triggering condition is based on this model-based prediction of the state as it is given by

$$t_{k+1} = \inf\{t > t_k \mid \|x_s(t) - x(t)\| \geq \delta\}. \quad (20)$$

Hence, when the prediction $x_s(t)$ diverts to far from the measured state $x(t)$, the next event is triggered so that updates of the state are sent to the actuator. These model-based triggering schemes can enhance the communication savings as they reduce the number of events by using model-based knowledge.

Other model-based event-triggered control schemes are proposed, for instance, in [Yook et al., 2002], [Garcia & Antsaklis, 2013], [Heemels & Donkers, 2013].

### C. Triggering with time-regularization

Time-regularization was proposed for *output-based* triggering to avoid the occurrence of accumulations in the execution times (Zeno behavior) that would obstruct the existence of a positive lower bound on the inter-execution times $t_{k+1} - t_k$, $k \in \mathbb{N}$. In [Tallapragada & Chopra, 2012a], [Tallapragada & Chopra, 2012b], the triggering update

$$t_{k+1} = \inf\{t > t_k + T \mid \|e(t)\| \geq \sigma\|x(t)\|\} \quad (21)$$

was proposed, where $T > 0$ is a built-in lower bound on the minimal inter-execution times. The authors discussed how $T$ and $\sigma$ can be designed to guarantee closed-loop stability. In [Heemels et al., 2008] a similar triggering was proposed using an absolute-type of triggering.

An alternative to exploiting a built-in lower bound $T$ is combining ideas from time-triggered control and

event-triggering control. Essentially, the idea is to only verify a specific event-triggering condition at certain equidistant time instants $kT_s$, $k \in \mathbb{N}$, where $T_s > 0$ is the sampling period. Such proposals were mentioned in, for instance, [Arzén, 1999], [Yook et al., 2002], [Henningsson et al., 2008], [Heemels et al., 2008], [Heemels et al., 2013]. In this case the execution times are given by

$$t_{k+1} = \inf\{t > t_k \mid t = kT_s,\ k \in \mathbb{N},\ \text{and}\ \|e(t)\| \geq \sigma\|x(t)\|\} \tag{22}$$

in case a relative triggering is used. In [Heemels et al., 2013] the term *periodic event-triggered control* was coined for this type of control.

### D. Decentralized triggering conditions

Another important extension of the mentioned event-triggered controllers, especially in large-scale networked systems, is the decentralization of the event-triggered control. Indeed, if one focusses on any of the above mentioned event-triggering conditions (take for example (5)), it is obvious that the full state variable $x(t)$ has to be continuously available in a central coordinator to determine if an event is triggered or not. If the sensors that measure the state are physically distributed over a wide area, this assumption is prohibitive for its implementation. In such cases, it is of high practical importance that the event-triggering mechanism can be decentralized and the execution of control tasks can be executed based on local information. One first idea could be to use *local* event-triggering mechanisms for the $i$-th sensor that measures $x_i$. One could "decentralize" the condition (5), into

$$t^i_{k^i+1} = \inf\{t > t^i_{k^i} \mid \|e_i(t)\| \geq \sigma\|x_i(t)\|\} \tag{23}$$

in which $e_i(t) = x_i(t^i_{k^i}) - x_i(t)$ for $t \in [t^i_{k^i}, t^i_{k^i+1})$, $k^i \in \mathbb{N}$. Note that each sensor now as its own execution times $t^i_{k^i}$, $k^i \in \mathbb{N}$ at which the information $x_i(t)$ is transmitted. More importantly, the triggering condition (23) is based on local data only and does not need a central coordinator having access to the complete state information. Besides since (23) still guarantees that (13) holds, stability properties can still be guaranteed, see [Mazo Jr. & Tabuada, 2011].

Several other proposals for decentralized event-triggered control schemes were made, *e.g.*, [Persis et al., 2013], [Wang & Lemmon, 2011], [Garcia & Antsaklis, 2013], [Yook et al., 2002], [Donkers & Heemels, 2012].

### E. Triggering for multi-agent systems

Event-triggered control strategies are suitable for cooperative control of multi-agent systems. In multi-agent systems, local control actions of individual agents should lead to a desirable global behavior of the overall system. A prototype problem for control of multi-agent systems is the agreement problem (also called the consensus or rendezvous problem), where the state of all agents should converge to a common value (sometimes the average of the agents' initial conditions). The agreement problem has been shown to be solvable for certain low-order dynamical agents in both continuous and discrete time, *e.g.*, [Olfati-Saber et al., 2007]. It was recently shown in [Dimarogonas et al., 2012], [Shi & Johansson, 2011], [Seyboth et al., 2013] that the agreement problem can be solved using event-triggered control. In [Seyboth et al., 2013] the triggering times for agent $i$ are determined by

$$t^i_{k^i+1} = \inf\{t > t^i_{k^i} \mid C_i(x_i(t), x_i(t^i_{k^i})) > 0\}, \tag{24}$$

which should be compared to the triggering times (5). The triggering condition compares the current state value with the one previously communicated, similarly to the previously discussed decentralized event-triggered control (see (23)), but now the communication is *only* to the agent's neighbors. Using such event-triggered communication, the convergence rate to agreement (*i.e.*, $\|x_i(t) - x_j(t)\| \to 0$ as $t \to \infty$ for all $i, j$) can be maintained with a much lower communication rate than for time-triggered communication.

## V. Outlook

Many simulation and experimental results show that event-triggered and self-triggered control strategies are capable of reducing the number of control task executions, while retaining a satisfactory closed-loop performance. In spite of these results, the actual deployment of these novel control paradigms in relevant applications is still rather marginal. Some exceptions include recent event-triggered control applications in underwater vehicles [Teixeira et al., 2010], process control [Lehmann et al., 2012], and control over wireless networks [Araujo et al., 2013]. To foster the further development of event-triggered and self-triggered controllers in the future, it is therefore important to validate these strategies in practice, next to building up a complete system theory. Regarding the latter, it is fair to say that, even though many interesting results are currently available, the system theory for event-triggered and self-triggered control is far from being mature, certainly compared to the vast literature on time-triggered (periodic) sampled-data control. As such, many theoretical and practical challenges are ahead of us in this appealing research field.

## References


[Araujo et al., 2013] Araujo, J., M. Mazo Jr., A. Anta, P. Tabuada, & K. H. Johansson 2013. System architectures, protocols, and algorithms for aperiodic wireless control systems. IEEE Transactions on Industrial Informatics. To appear.

[Arzén, 1999] Arzén, K.-E. 1999. A Simple Event-Based PID Controller. In Preprints IFAC World Conf., volume 18, pages 423–428.

[Aström & Bernhardsson, 1999] Aström, K.J., & B.M. Bernhardsson 1999. Comparison of periodic and event based sampling for first order stochastic systems. In Proc. IFAC World Conf., pages 301–306.

[Aström & Wittenmark, 1997] Aström, K. J., & B. Wittenmark 1997. Computer Controlled Systems. Prentice Hall.

[Dimarogonas et al., 2012] Dimarogonas, D. V., E. Frazzoli, & K. H. Johansson 2012. Distributed Event-Triggered Control for Multi-Agent Systems. IEEE Transactions on Automatic Control, 57(5):1291–1297.

[Donkers & Heemels, 2010] Donkers, M.C.F., & W.P.M.H. Heemels 2010. Output-Based Event-Triggered Control with Guaranteed $\mathcal{L}_\infty$-gain and Improved Event-Triggering. In Proc. IEEE Conf. Decision & Control, pages 3246 – 3251.

[Donkers & Heemels, 2012] Donkers, M.C.F., & W.P.M.H. Heemels 2012. Output-Based Event-Triggered Control with Guaranteed $\mathcal{L}_\infty$-gain and Improved and Decentralised Event-Triggering. IEEE Trans. Autom. Control, 57(6):1362–1376.

[Franklin et al., 2010] Franklin, G.F., J.D. Powel, & A. Emami-Naeini 2010. Feedback Control of Dynamical Systems. Prentice Hall.

[Garcia & Antsaklis, 2013] Garcia, E., & P. J. Antsaklis 2013. Model-Based Event-Triggered Control for Systems With Quantization and Time-Varying Network Delays. Automatic Control, IEEE Transactions on, 58(2):422–434.

[Goebel et al., 2009] Goebel, R., R. Sanfelice, & A.R. Teel 2009. Hybrid dynamical systems. IEEE Control Syst. Mag., 29:28–93.

[Gupta, 1963] Gupta, S. 1963. Increasing the sampling efficiency for a control system. Automatic Control, IEEE Transactions on, 8(3):263 – 264.

[Heemels et al., 2013] Heemels, W.P.M.H., M.C.F. Donkers, & A.R. Teel 2013. Periodic event-triggered control for linear systems. IEEE Trans. Autom. Control, 58(4):847–861.

[Heemels et al., 2008] Heemels, W.P.M.H., J.H. Sandee, & P.P.J. van den Bosch 2008. Analysis of event-driven controllers for linear systems. Int. J. Control, 81:571–590.

[Heemels & Donkers, 2013] Heemels, W. P. M. H., & M. C. F. Donkers 2013. Model-based periodic event-triggered control for linear systems. Automatica, 49(3):698–711.

[Henningsson et al., 2008] Henningsson, T., E. Johannesson, & A. Cervin 2008. Sporadic Event-Based Control of First-Order Linear Stochastic Systems. Automatica, 44:2890–2895.

[Lehmann et al., 2012] Lehmann, D., G. A. Kiener, & K. H. Johansson 2012. Event-triggered PI control: saturating actuators and anti-windup compensation. In IEEE Conference on Decision and Control, Maui, HI, USA.

[Lunze & Lehmann, 2010] Lunze, J., & D. Lehmann 2010. A state-feedback approach to event-based control. Automatica, 46:211–215.

[Mazo Jr. & Tabuada, 2011] Mazo Jr., M., & P. Tabuada 2011. Decentralized event-triggered control over wireless sensor/actuator networks. IEEE Transactions on Automatic Control, Special Issue on Wireless Sensor and Actuator Networks, 56(10):2456–2461.

[Miskowicz, 2006] Miskowicz, M. 2006. Send-on-delta concept: An event-based data-reporting strategy. Sensors, 6:49–63.

[Olfati-Saber et al., 2007] Olfati-Saber, R., J. A. Fax, & R. M. Murray 2007. Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE, 95(1):215–233.

[Persis et al., 2013] Persis, C. De, R. Sailer, & F. Wirth 2013. Parsimonious event-triggered distributed control: A Zeno free approach. Automatica, 49(7):2116–2124.

[Postoyan et al., 2011] Postoyan, R., A. Anta, D. Nešić, & P. Tabuada 2011. A unifying Lyapunov-based framework for the event-triggered control of nonlinear systems. In Proc. Joint IEEE Conf. Decision and Control and European Control Confernce, pages 2559–2564.

[Seyboth et al., 2013] Seyboth, G. S., D. V. Dimarogonas, & K. H. Johansson 2013. Event-based broadcasting for multi-agent average consensus. Automatica, 49(1):245–252.

[Shi & Johansson, 2011] Shi, G., & K. H. Johansson 2011. Multi-agent robust consensus—Part II: application to event-triggered coordination. In IEEE Conference on Decision and Control, Orlando, FL.

[Tabuada, 2007] Tabuada, P. 2007. Event-Triggered Real-Time Scheduling of Stabilizing Control Tasks. Automatic Control, IEEE Transactions on, 52(9):1680 –1685.

[Tallapragada & Chopra, 2012a] Tallapragada, P., & N. Chopra 2012a. Event-triggered decentralized dynamic output feedback control for LTI systems. In IFAC Workshop on Distributed Estimation and Control in Networked Systems, pages 31–36.

[Tallapragada & Chopra, 2012b] Tallapragada, P., & N. Chopra 2012b. Event-triggered dynamic output feedback control for LTI systems. In Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, pages 6597–6602.

[Teixeira et al., 2010] Teixeira, P. V., D. V. Dimarogonas, K. H. Johansson, & J. Borges de Sousa 2010. Event-based motion coordination of multiple underwater vehicles under disturbances. In IEEE OCEANS, Sydney, Australia.

[Velasco et al., 2003] Velasco, M., P. Marti, & J. M. Fuertes 2003. The Self Triggered Task Model for Real-Time Control Systems. In Proceedings of 24th IEEE Real-Time Systems Symposium, Work-in-Progress Session.

[Wang & Lemmon, 2011] Wang, X., & M. D. Lemmon 2011. Event-Triggering in Distributed Networked Systems with Data Dropouts and Delays. IEEE Trans. Autom. Control, pages 586–601.

[Yook et al., 2002] Yook, J.K., D.M. Tilbury, & N.R. Soparkar 2002. Trading Computation for Bandwidth: Reducing Communication in Distributed Control Systems Using State Estimators. IEEE Trans. Control Systems Technology, 10(4):503–518.