Distributed Optimization, Estimation, and Control
of Networked Systems through Event-triggered Message Passing

1 Introduction

This project will investigate *distributed optimization, estimation, and control* of *networked systems* through the use of *event-triggered message passing*. Networked systems consist of several interconnected subsystems or *agents*. Examples of networked systems include the national power grid, traffic networks, the Internet, and water/gas distribution networks. All of these examples are components of our national civil infrastructure, so that optimally managing such networks is in the national interest. These systems are often managed in a centralized manner, where decisions are made by a single computer. This creates a single point of failure that can be addressed by distributing decision making throughout the entire networked system.

Communication issues have a great impact on the performance of distributed decision systems. Individual agents coordinate their actions through message exchanges. This message passing is done over a digital communication network where messages are broadcast at discrete time instants. Communication is done using wireless radios where message collisions in a shared channel reduce throughput and increase message latency. These considerations degrade the ability of agents to successfully coordinate their actions, unless one increases the cost and complexity of the supporting communication infrastructure. The design of distributed decision making systems, therefore, must find ways of reducing overall message passing complexity without sacrificing too much of the overall system's performance.

This project proposes a novel way of reducing the amount of communication required in the distributed optimization, control, and estimation of networked systems. The project uses an *event-triggered* approach to message passing. Under event triggering, each agent broadcasts its state information to its neighbors when an internal error signal exceeds a state-dependent threshold. Prior work on feedback [3, 70, 85] and networked control systems [89, 82, 87] has demonstrated that event triggering can greatly reduce communication traffic while still maintaining acceptable levels of control system performance. More recent results [79, 78, 80] have empirically demonstrated that event-triggered distributed optimization can reduce a system's overall message passing complexity by *several orders of magnitude*. These results suggest that event-triggered message passing may have a transformational impact on the control, estimation, and optimization of sampled-data systems. The objective of this project is to study the advantages and limitations of event-triggering from both a theoretical and practical perspective. Theoretical analyses will characterize the fundamental limitations of event-triggering in control, estimation, and optimization. The project will apply the results of these studies to at least three applications that include 1) distributed control of wastewater networks, 2) distributed receding horizon control of autonomous multi-robot groups, and 3) distributed control of mesh microgrids.

Intellectual Impacts: Event-triggered message passing generates discrete-abstractions of dynamical systems that interact through *sporadic* rather than *periodic* message streams. This project, therefore, is developing algorithms that drop the traditional periodic message passing model that is almost universally assumed in large-scale engineering systems. Dropping the requirement for synchrony and periodicity will have a transformational effect on how we build networked embedded systems, thereby profoundly impacting a variety of engineering disciplines that include real-time, cvil, and mechanical systems engineering.

Broader Impacts: The impact of this project will be broadened through interactions with industrial partners, EmNet LLC and Odyssian LLC. EmNet LLC is interested in using event-triggered algorithms on its CSOnet system [53]. CSOnet is a wireless sensor-actuator network that EmNET is building in a handful of U.S. cities to address environmental problems arising from combined-sewer overflows (CSO). Odyssian LLC has already used an event-triggered control approach in an experimental electrical microgrid as part of a phase I STTR. They are interested in seeing the results of this project applied to larger scale microgrids. The project's impact will also be broadened through curriculum and educational outreach activities. In particular, the project will develop a series of on-line lectures on event-triggered dynamical systems theory using the applications developed under this project to illustrate the design principles. Additional outreach will be made to European colleagues who are also working on networked control systems. The principal investigator is lecturing on event-triggering at a European Ph.D. summer school at the University of Siena, Italy. These types of outreach activities will be continued under this project. Finally, the principal investigator will build upon earlier interactions with a local middle school to develop a presentation on multi-robotic systems that can dovetail with after-school activities on autonomous robotics.

2 Event-Triggered Abstractions of Dynamical Systems

Event-triggered message passing refers to systems in which subsystems broadcast information when an internal error signal exceeds a state-dependent threshold. The error signal measures the ”novelty” of the information embedded within that subsystem's current state. ”Novelty”, in this case, loosely refers to how important that information is in helping other subsystems optimize their behavior. Event-triggering therefore provides a framework for system coordination that is fundamentally cooperative in nature. Subsystems broadcast their state information when they believe that information is needed by their neighbors. Event-triggered data streams tend to be asynchronous and sporadic in a way that is significantly more flexible than conventional synchronous and periodic real-time systems. What this means is that the discrete abstractions traditionally used to analyze such systems are significantly different than what have been used in the past. There is, therefore, a great need to better understand the fundamental system theoretic properties of these systems so we can reliably design event-triggered systems with predictable and scalable properties. This section discusses the research challenges that must be addressed in developing a systems science for event-triggered systems.

Problem Formulation: In networked dynamical systems, individual subsystems (agents) coordinate their actions by passing messages between each other. A fundamental challenge concerns the frequency with which these messages must be passed. The cost of the communication infrastructure will be overwhelming if messages are passed too frequently and system coordination cannot be achieved if messages are not passed often enough. This issue may be addressed by only having agents broadcast their information when absolutely necessary. Under event-triggering an agent decides to broadcast its local state when some internal error signal exceeds a state dependent threshold. Prior work [89, 82, 87] in event-triggered feedback systems has shown that this approach can dramatically reduce the number of messages that each agent might broadcast.

To clearly explain how event-triggering works, let's consider a collection of N dynamical agents. A sequence of *broadcast times*, $\{B_i[k]\}_{k=1}^{\infty}$, and *reception times* $\{R_i[k]\}_{k=1}^{\infty}$ are associated with the ith agent. The time instants $B_i[k]$ and $R_i[k]$ represent the kth consecutive times when a message was broadcast or received by agent i. The local state of the ith agent ($i = 1, \ldots, N$) is a function $x_i : \Re \rightarrow \Re^{n_i}$ where n_i is the local state's dimension. We let x_{-i} denote the states of the ith agent's neighbors. At each broadcast time the agent transmits its local state to its neighbors and we let that broadcast state be represented by the function $\hat{x}_i : \Re \rightarrow \Re^{n_i}$ where $\hat{x}_i(t) = x_i(B_i[k])$ for $t \in [B_i[k], B_i[k+1])$. This means that $\hat{x}_i(t)$ is the local state of agent i at the most recent broadcast time. It is a piecewise constant function that only changes value at the broadcast times. The ith agent's local state is assumed to satisfy the following piecewise-continuous differential equation

$$\dot{x}_i(t) \quad = \quad f_i(x_i(t), x_{-i}(t)) + \sum_{j \neq i} g_{ij}(\hat{x}_j(t)) \tag{1}$$

for all $t \in [R_i[k], R_i[k+1])$ and all $k = 1, \ldots, \infty$. The initial condition for equation 1 is the final state value obtained in the preceding time interval $[R_i[k-1], R_i[k]]$. The functions f_i and g_{ij} are suitable vector fields ensuring that equation 1 has a unique solution. The function f_i represents the *physical coupling* between the ith agent and its neighbors. The function g_{ij} represents the communication based coupling between the ith and jth agent, so that that g_{ij} has nonzero support only when agents i and j are neighbors that can communication with each other.

The system equation 1 describes a *sampled-data* system in which the ith agent uses samples of its neighboring agents' local states. We're interested in characterizing a sequence of broadcast times, $\{B_i[k]\}_{k=1}^{\infty}$ such that the sampled-data system is semiglobal asymptotically stable. We obtain this characterization by treating the sampling-data system as a ”discrete” approximation of a continuous system that we already know is input-to-state stable (ISS). This ”continuous” system is described by the system equations

$$\dot{x}_i(t) = f_i(x_i(t), x_{-i}(t)) + \sum_{j \neq i} g_{ij}(x_j(t)) \tag{2}$$

for $t \in [0, \infty)$ and $i = 1, \ldots, N$. Note that the system in equation 2 differs from the sampled system in equation 1 in the argument of g_{ij}. For the ”continuous” system, the argument to g_{ij} is the actual state of the neighboring agent (rather than its sampled version). For the ”discrete” system, the argument to g_{ij} is the sampled state of the agent. Equation 2 may therefore be viewed as a system with

”continuous-broadcasting” between agents, whereas equation 1 is a system with ”discrete-broadcasting” between agents.

To establish the asymptotic stability of the discrete-broadcast system we require that the continuous-broadcast system with suitable input disturbances is input-to-state stable. Let $V : \Re^n \rightarrow \Re$ ($n = \sum_i n_i$ is the state dimension of the entire group) be positive definite and assume there exist class $\mathcal{K}$ functions $\alpha_i : \Re \rightarrow \Re$ and $\beta_i : \Re \rightarrow \Re$ ($i = 1, \ldots, N$) such that

$$\dot{V} = \sum_{i=1}^{N} \frac{\partial V}{\partial x_i} \left(f_i(x_i, x_{-i}) + \sum_{j \neq i} g_{ij}(x_j(t) + \tilde{x}_j(t)) \right) \leq \sum_{i=1}^{N} (-\alpha_i(\|x_i(t)\|_2) + \beta_i(\|\tilde{x}_i\|_2)) \tag{3}$$

Equation 3 essentially means that V is an ISS-Lyapunov function for the overall system [64, 65] with respect to the disturbance $\tilde{x}_j$ that enters through the g_{ij} function.

We use equation 3 to establish the asymptotic stability of the sampled-data system by requiring that

$$-\alpha_i(\|x_i(t)\|_2) + \beta_i(\|\tilde{x}_i\|_2) < 0 \tag{4}$$

for all time t. The inequality in equation 4 provides the basis for selecting broadcast times. In particular, let’s define the *error* between the jth agent’s broadcast state, $\hat{x}_j$, and the actual state, x_j, as the disturbance $\tilde{x}_j(t) = \hat{x}_j(t) - x_j(t)$. Substituting this back into equation 3 shows that V is also a Lyapunov function for the discrete-broadcast system in equation 1. We may therefore conclude that if our ”continuous-broadcast” version of the system is ISS with respect to the error $\tilde{x}$, then its ”discrete-broadcast” version has an asymptotically stable equilibrium. In particular, equation 4 is the *event-trigger* that an agent uses to decide whether or not to broadcast its state. If the locally observed state is about to violate the event-triggering condition, then that agent broadcasts its state to its neighbors. As long as all agents agree to follow this broadcast protocol, then the discrete-broadcast version of the system will have an asymptotically stable equilbilibrium.

Preliminary Work: Event-triggered control has been studied in relay [72] and pulse-width modulated [58] systems since the 1960’s. Recently this idea has been resurrected as a method to reduce the communication complexity in networked control systems. Threshold based feedback in networked systems and its impact on stability was discussed in [76, 91]. More recently event-triggered feedback has been suggested as a way to simplify embedded control systems [3] and has appeared under a variety of names such as interrupt-based feedback [22], Lebesgue sampling [4], asynchronous sampling [75], state-triggered feedback [70], or self-triggered feedback [33, 86, 85, 83]. The analytical view at the heart of the event-triggering condition (equation 4) relies on ISS concepts. The ISS viewpoint of sampled data systems will be found in [55] with extensions to networked systems in [6, 69]. These methods allow us to study the impact that sporadic sampling, delays, and data dropouts [21] have on various stability concepts (input-output and asymptotic). The use of the ISS approach to event-triggering for embedded control was formalized in [70].

Our work applied these event-triggering ideas to networked systems [84, 82, 87]. In this work we developed ”decentralized” event conditions that are only a function of the agent’s local state. A decentralized event-trigger was determined for networked linear systems with symmetric communication links [82, 84]. Extensions to nonlinear systems with packet dropouts and delays were later derived [88, 87]. These nonlinear generalizations are embodied in the event-triggering conditions described above in equation 4. As in the case of the event-triggered feedback systems studied in [85], it is possible to derive useful bounds on the broadcast periods and tolerance to packet dropouts [87] in the networked case. These results again suggest that we can dramatically reduce the network bandwidth through the use of event-triggering schemes.

Research Challenges: The recent work cited above demonstrated that event-triggering can reduce message-passing complexity. These are preliminary efforts establishing the importance of event-triggering in the design of linear networked control systems. To extend these ideas to highly nonlinear control, estimation, and optimization applications, we need to establish a systems theoretical framework for event-triggered abstractions of dynamical systems. This project seeks to develop just such a systems theory for event-triggered systems. The following paragraphs detail some of the issues that will need to be addressed over the course of the project.

Let’s first turn to the assumptions embodied in equations 1 and 3. It’s reasonable to require that the interaction occurs in an additive manner as appears in equation 1. Such an assumption is valid in many applications including microgrids, water/gas networks, traffic control and multi-robot formation control.

The primary restriction comes from equation 3. This condition requires that x_i and $\tilde{x}_i$ enter the right hand side in an additive manner before being operating upon by the function g_{ij}. This requirements can be extremely restrictive as it is not satisfied by many important classes of nonlinear systems. Such classes are found in mechanical, biological, and chemical networks. Consequently, the generality of the above approach will be limited by the assumption in equation 3. This issue may be addressed by considering more general comparison functions in which α_i and β_i are also dependent on neighboring states. This is one direction that we'd like to address under the project.

Our earlier work quantified the impact that data dropouts and delays have on the performance of the event-triggered system, but again this work is preliminary. Ideally, one would like to determine the functional relationship between delays and sampling period, similar to what was done in [21]. This is part of a broader ambition which would develop a theory that unifies our studies of quantization and sampling period. Such a theory would build upon earlier work in studying the minimum amount of information required for feedback control [38, 18, 71]. In fact the estimates of sampling time found in [55] and [85] are reminiscent of the minimum bit rate formulas for stabilizing quantized feedback systems [38]. It seems that there should be some framework in which all of these aspects of feedback communication (event-triggering, quantization, data dropouts, delays, sampling period) can be treated in a unified manner. We believe the ISS viewpoint may provide the technical tools to build that unified theory.

Much of our prior work has focused on systems with state accessibility. We want to extend event-triggering to output-feedback systems. One way of doing this is to first study estimation problem and then investigate whether some form of certainty-equivalence can be obtained for such systems. This motivates an examination of event-triggered estimation which is discussed below in section 4. This project will see how results in event-triggered estimation can be used to develop event-triggered output controllers.

A grand research challenge concerns the scheduling of sporadic message broadcasts in event-triggered NCS. We hope to develop network protocols that coordinate message passing with sampling time and delay constraints that are required for a specified level of overall control system performance. This invariably requires that we prioritize message transmission in a decentralized manner. One way of doing this is to adopt a stochastic framework for broadcast scheduling that is similar to what was done in [45]. This project hopes to address the scheduling issue in event-triggered systems.

One interesting aspect of event-triggering is that it takes a ”continuous” system and transforms it into a ”discrete” system. In particular, the prior work in [85] essentially constructs a discrete-abstraction of the plant. This abstraction is an input-output model of these system in which the input/output signals are event-streams. An event-stream signal is a sequence of ”events” where each event consists of an ordered pair (t_k, v_k) where t_k is the time when the event occurred and v_k is the ”value” of the event. So the event-triggering formalism adopted in this project is essentially constructing an equivalent event-flow model of the plant. Such event-flow systems fit within the behavioral framework pioneered by J. Willems [59]. Under this framework the system is a set of behaviors and the composition of such systems is the intersection of these behavioral sets. Behavioral models based on event streams preserve a form of ”causality” under system composition [30, 31] that make them particularly well-suited for modeling the hierarchical scheduling of multiple resources [81, 63, 62, 51, 52]. Since these models have such attractive compositional properties, this immediately suggests that we may be able to take event-triggered abstractions of physical processes and directly interface them with discrete-event models of real-time computational processes. Such a hybrid interconnection of physical and computational processes may provide a powerful new way of modeling complex cyber-physical systems. Relatively little is known about such system models, especially with regard to traditional system-theoretic concepts such as controllability and observability. An important goal of this project is to develop a systems theory for event-triggered systems that can provide a useable modeling framework to develop cross-disciplinary approaches for the design of large-scale cyber-physical systems.

3 Event-triggered Optimization of Networked Systems

This project will use event-triggering to develop distributed algorithms that solve constrained static optimization problems over networked systems. The approach is based on recent results [79, 78, 80] that experimentally demonstrate that event-triggered algorithms have a computational complexity that is several orders of magnitude smaller than comparable dual-decomposition algorithms [46] used to solve *network utility maximization* (NUM) problems [27]. The objective of this task is to extend that earlier work to a larger class of constrained optimization problems that are found in other networked system applications involving energy/material transport across networked infrastructure. The specific application to be addressed is the optimal management of combined-sewer overflow events [53] and power dispatch in microgrids [32].

Problem Statement: We use the network utility maximization problem (NUM) to illustrate the potential benefits of event-triggered optimization. The NUM problem tries to maximize the overall utility received by a group of users transmitting over a set of shared capacity constrained communication links. A shadow price formalism for solving this problem [27] was implemented as a distributed algorithm [46] known as the "dual-decomposition" algorithm. Extensions of this algorithm were later applied to cross layer control of ad hoc wireless networks [13, 90, 14].

Formally, the NUM problem has a group of N users where the ith user transmits data at a positive rate x_i over a set of M links. The relationship between users and links is given by an incidence matrix $A \in \Re^{M\times N}$ whose ijth element is 1 if user i uses link j and is zero otherwise. The vector of user rates is denoted as $x = \left[\begin{array}{ccc} x_1 & \cdots & x_N \end{array} \right]^T$. Each user receive a utilty $U_i(x_i)$ for transmitting at rate x_i. The NUM problem tries to find a distributed algorithm that selects user rates to solve the following optimization problem,

$$\begin{array}{ll} \text{maximize:} & \sum_{i=1}^N U_i(x_i) \\ \text{subject to:} & Ax \leq c, \quad x \geq 0 \end{array} \tag{5}$$

The constraint is a linear constraint that requires the total flow rate through the link j be less than or equal to a capacity limit c_j. The vector c therefore is a vector of the link capacities. This problem seeks to maximize a collective additive measure of the total utility subject to the specified link capacity limits.

A distributed solution for this problem may be obtained from the dual problem,

$$\begin{array}{ll} \text{minimize:} & \max_{x\geq 0} \left(\sum_{i=1}^N U_i(x_i) - p^T(Ax - c) \right) \\ \text{subject to:} & p \geq 0 \end{array} \tag{6}$$

where $p = \left[\begin{array}{ccc} p_1 & \cdots & p_M \end{array} \right]^T$ is the vector of Lagrange multipliers associated with the link capacity constraints. In particular, the vector's jth component, p_j, can be viewed as a "shadow price" [27] that the link charges the user. If x^* and p^* are vectors solving the above dual problem, then it is well known that x^* also solves the original NUM problem.

The dual formulation given in equation 6 suggests a distributed solution in which users adjust their rates assuming a fixed link price, and links adjust their prices assuming fixed user rates. This interaction sets up a feedback loop between the group of users and the group of links which was formalized as the "dual-decomposition" algorithm [46]. This algorithm generates a sequence of user rates $\{x[k]\}_{k=1}^\infty$ and link prices $\{p[k]\}_{k=1}^\infty$ that asymptotically converges to the solution of the dual problem. If we let $U_i(x_i) = \log(x_i)$, then the recursion used in generating this sequence is a dynamical system of the form,

$$x_i[k+1] = \arg\max_{x\geq 0} \left\{ U_i(x_i[k]) - x_i[k] \sum_{j\in L_i} p_j[k] \right\} = \frac{1}{\sum_{j\in L_i} p_j[k]} \tag{7}$$

$$p_j[k+1] = \max \left\{ 0, p_j[k] + \gamma \left(\sum_{i\in S_j} x_i[k] - c_j \right) \right\} \tag{8}$$

for $i = 1, \ldots, N$, $j = 1, \ldots, M$, S_j is the set of users using link j, L_i is the set of links used by user i, and γ is a step size that is chosen small to assure the recursion is convergent.

Preliminary Work: Equations 7 and 8 represent a set of coupled dynamical equations in which the link broadcasts its price to the relevant users at each time step. An event-triggered implementation of this algorithm would have a sequence of broadcast times $\{B_j[k]\}_{k=1}^\infty$ when the jth link broadcasts its current price to its users. The broadcast price is denoted as $\hat{p}_j(t) = p_j(B_j[k])$ for $t \in [B_j[k], B_j[k+1])$. The link's price, $p_j(t)$, is updated in a continuous fashion according to the differential equation

$$\frac{dp_j(t)}{dt} = \begin{cases} \sum_{i\in S_j} x_i(t) - c_j & \text{if } p_j(t) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$x_i(t) = \frac{1}{\sum_{j\in L_i} \hat{p}_j(t)} \tag{10}$$

for $t \in [B_j[k], B_j[k+1])$ for all j and k. Note that we've modified the link-update equation 8 to be a continuous-time process, rather than a discrete-time process. This is done with the expectation that computational cycles are relatively inexpensive, whereas communication is expensive. So we can approximate the discrete update in equation 8 with a continuous-process.

As we did in section 2, we introduce a Lyapunov function for the continuous-broadcast system. The function, V, is the same one that was used in [27] for the dual algorithm.

$$V(p) = -\sum_{i=1}^{N} \log\left(\sum_{j\in L_i} p_j\right) + \sum_{j=1}^{M} c_j p_j$$

The directional derivative of V is

$$\dot{V} = \sum_{j=1}^{M} \frac{\partial V(p)}{\partial p_j}\dot{p}_j(t) = -\sum_{j=1}^{M}\left(\sum_{i\in S_j} \frac{1}{\sum_{k\in L_i} p_k(t)} - c_j\right)^2$$

We now replace the neighbor's p_j by $\hat{p}_j$ to represent the discrete broadcast system. Let $\tilde{p}_j(t) = \hat{p}_j(t) - p_j(t)$ denote an error term which represents the difference between the link's current price and the price it last broadcast to its users. We now replace $\hat{p}_j(t)$ with $p_j(t) + \tilde{p}_j(t)$ to model the impact that the error $\tilde{p}$ has on the behavior of the Lyapunov function. In this case, we can use the inequality $-\frac{1}{x+y} \leq x + y - 1$ to determine class $\mathcal{K}$ functions satisfying the equation 3. This analysis suggests an event trigger of the form

$$|p_j(t) - \hat{p}_j(t)| \leq \rho\, |p_j(t)| \tag{11}$$

where ρ is a suitably chosen constant between 0 and 1.

We can test this event-trigger by simulating the event-triggered dual-decomposiition algorithm on the network shown to the left of figure 1. In this simulation a link broadcasts using the event-trigger in equation 11. For this particular example, we arbitrarily chose ρ to be 0.1. The simulation considers a network with three users and five links. The link capacities were chosen to switch halfway through the simulation. The middle plot in figure 1 has three plots. The top plot shows that the system converges to the desired user rates and the middle plot shows the link prices. The bottom plot is the most interesting. This plot uses blue dots to mark the number of agents that are broadcasting at each time instant in the simulation. What we see is that as the user rates and prices reach their equilibrium, the number of broadcasting agents begins to drop. The solid blue line in the bottom plot shows a smoothed plot of the number of broadcasting agents, averaged over a finite window. This blue line more clearly shows that the "density" of broadcasting states significantly reduces as the system approaches its equilibrium point.

Note that when the initial error $\tilde{p}$ is large, that the number of broadcasts is large, but quickly decreases to a very small level, depending on how many of the constraints are actually active. In contrast, a synchronous implementation of the dual decomposition algorithm would have all 5 links broadcast their updates every time step. This is an interesting result because it shows that event-triggering can dramatically reduce the total number of broadcasts that are required. In particular, for the second constraint c_1, we see that the average number of broadcasts reduces to just a single agent, as soon as the prices stabilize. So as suggested by our earlier work with event-triggered feedback, the use of event-triggering in distributed optimization algorithms such as the NUM problem can dramatically reduce the algorithm's communication complexity.

The preceding event-triggered implementation of dual-decomposition can be extended to other distributed algorithms solving the NUM problem. More detailed studies of such algorithms were done for an augmented Lagrangian algorithm [78] and a barrier method algorithm [79] solving the NUM problem. In this case, a more detailed study was done that directly compared the total number of messages that were passed under the event-triggered algorithm versus the dual-decomposition algorithm where the stabilizing step size was chosen according to the guidelines provided in [46]. The results from this study are shown on the right hand side of figure 1. This plot graphs the total number of iterations used by traditional dual decomposition and an event-triggered NUM algorithm as a function of the network's diameter. The remarkable thing to be seen is that the total number of messages used by the event-triggered algorithm is nearly two orders of magnitude lower than what is found under dual-decomposition. The second observation is that the message passing complexity under event-triggering appears to be *scale-free*. In other words,

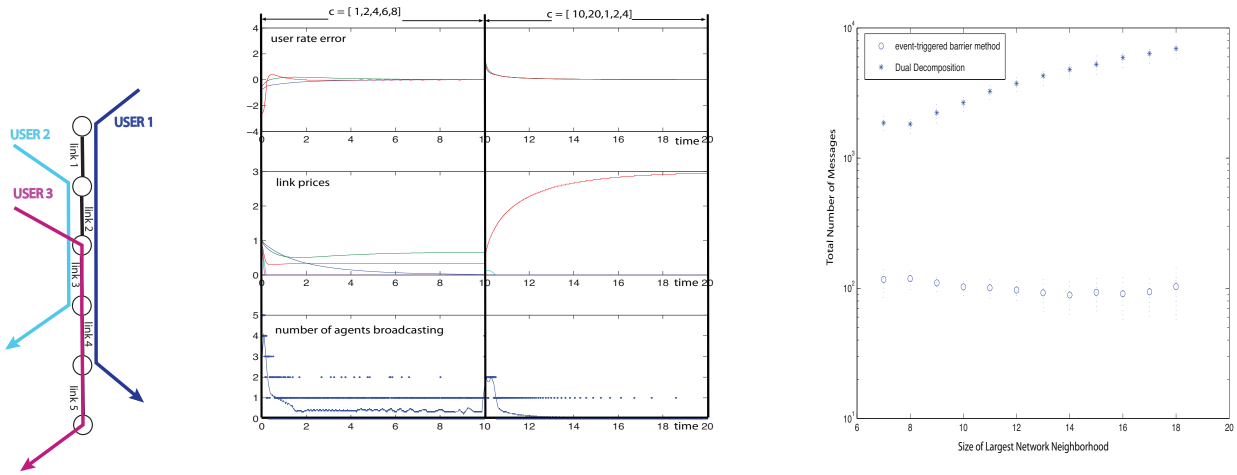


Figure 1: (left) user-link geometry, (middle) simulation results for sample problem, (right) scaling results for event-triggered optimization [78]

again as suggested by earlier work , event triggering appears to significantly reduce the communication complexity of these distributed optimization algorithms.

Research Challenges: The preliminary work reviewed above suggests we can greatly reduce the communication complexity of distributed NUM algorithms through event-triggered communication. These findings, however, are primarily empirical in nature. These simulation results suggest that event-triggered optimization has highly attractive scaling properties. What is still missing is the formal characterization of such algorithms with regard to accuracy and convergence rate.

Analyzing the scaling, convergence, and sensitivity properties of event-triggered optimization have so far proven to be very difficult. The sporadic nature of the algorithm's message passing greatly complicates the analysis and has prevented us from obtaining deterministic guarantees on the algorithm's behavior. An important objective of this project is therefore to obtain an analytical characterization of the event-triggered algorithm's sensitivity, convergence, and accuracy. While "deterministic" guarantees have proven elusive, we believe it is possible to formally characterize the algorithm's properties using a probabilistic characterization. Similar probabilistic approaches have been used to characterize the complexity of PAC learning algorithms [73] [20]. We believe that such an approach may be useful in studying the convergence properties of the event-triggered optimization algorithm.

The triggering event in equation 12 generates very satisfactory performance. However, we must point out that this choice of the event only guarantees the convergence to some neighborhood of the equilibrium. This is because the system has a nonzero equilibrium point p^*. By equation 12, the triggering stops as long as the the difference between $p_j(t)$ and $\hat{p}_j(t)$ is no larger than $\rho|p_j^*|$. We believe that by using a time-varying coefficient ρ, we may be able to ensure the convergence of the algorithm to the exact solution. This approach was taken in some of our earlier work [78, 79].

Another issue concerns the way in which event-triggered optimization would be implemented in a real-life system. Computation of the shadow price must be done by an "agent" that can actually observe the inequality constraint associated with that price. For example, in applying utility maximization concepts to wireless ad hoc networks, Xue notes [90] that no single node is well-positioned to directly observe whether or not the network's capacity is being exceeded. Xue and his colleagues had to introduce an algorithm that used a virtual link contention graph to estimate network capacity limits. This type of issue appears in other applications as well. The power dispatch problem used in managing electrical power grids [5] has a similar issue, where constraints are on the total power generated within a given service area. The problem here is that there is no single agent that can observe how close that service area is to exceeding its capacity. This information must be inferred from an agent that indirectly observes or manages the entire service area. What these examples suggest is that while distributed optimization over networks may be formally stated as a NUM problem (or something similar), the actual application and our ability to observe important constraints within that application may necessitate a restructuring of the problem. What impact that restructuring may have on the scalability of the proposed event-triggered algorithm must be studied to determine which applications are best suited to the methods proposed in this project. To address this issue we will apply event-triggered optimization to the evaluation testbeds discussed in section 6.

4 Event-triggered Estimation of Networked Systems

This project will develop distributed event-triggered algorithms for estimation in networked systems. We will consider two distributed estimation problems that often appear in networked systems. The first problem is data fusion. Many wireless sensor network require sensors to forward their data over a multi-hop network to a central data fusion center. We propose developing event-triggered broadcast protocols for the sensors. The second problem is the distributed estimation problem as seen in consensus filtering [2]. This problem requires all agents to reach "consensus" on a noisy observation of a commonly observed process. We propose developing event-triggered broadcast mechanisms for such consensus filters.

Preliminary Work: The *consensus filtering problem* assumes that the local state of the ith agent satisfies the differential equation

$$\dot{x}_i(t) = \sum_{j \in N_i} A_{ij}(x_j - x_i) + \sum_{j \in N_i} A_{ij}(u_j - x_i) \tag{12}$$

where A_{ij} is one if agents i and j communicate and is zero otherwise. $u_i = \overline{u} + w_i$ is the ith agent's observation of an environmental variable, $\overline{u}$. We assume that this observation is corrupted by noise, w_i, which is statistically independent over all agents in the group. The agent's local state, x_i, represents that agent's estimate of the environmental variable $\overline{u}$.

We now consider an event-triggered version of the consensus filter in which the local state $x_i(t)$ satisfies the following differential equation

$$\dot{x}_i(t) = \sum_{j \in N_i} A_{ij}(\hat{x}_j(t) - x_i(t)) + \sum_{j \in N_i} A_{ij}(\hat{u}_j(t) - x_i(t)) \tag{13}$$

for $t \in [R_i[k], R_i[k+1])$. Recall that $R_i[k]$ denotes the kth time when agent i receives an update from its neighbor and $B_i[k]$ denotes the time when the agent broadcasts an update to it neighbors. The information broadcast by the agent i is $(\hat{x}_i(t), \hat{u}_i(t))$ where $\hat{x}_i(t) = x_i(B_i[k])$ and $\hat{u}_i(t) = u_i(B_i[k])$. Since the consensus filter is linear, it is easy to use methods identified in [84] to construct the comparison functions α_i and β_i. In this way we show that $\dot{V}$ satisfies the inequality in equation 4. For this system, the event-triggering condition is the inequality $\|\hat{x}_i(t) - x_i(t)\|_2 \leqslant \rho \|\hat{x}_i(t)\|_2$. With a suitable choice of ρ, V will be a Lyapunov function thereby proving (see discussion in section 2) that the event-triggered system in equation 13 is asymptotically stable to the desired equilibrium.

A Matlab simulation of the event-triggered consensus filter was used to examine the potential reduction in message passing when compared against the traditional consensus filter. The left side of figure 2 shows results for a group of 30 agents observing a single noisy scalar. A step change in $\overline{u}$ occurs halfway through the simulation. This is done to see if the filter can track the step change. The top graph shows that the estimation errors for all agents converges asymptotically to zero. The bottom graph plots the average number of agents that are broadcasting over a fixed interval of time. These plots show that as the state estimation error converges to zero, the total number of broadcasting agents decreases. After the second disturbance occurs, we see agents begin broadcasting again until the network has converged.

The results shown on the left side figure 2 are significant. In traditional consensus filters, information would continue to be exchanged even after the filter has converged. In our case, message passing essentially stops after the filter has converged. These results are consistent with the improvements seen in figure 1 for the NUM problem. These results therefore suggest that event-triggered broadcasting may be a very effective way of reducing the message passing complexity of distributed estimation algorithms.

The preliminary results in figure 2 suggest that similar advantages may be gained for other estimation problems found in networked systems. One of the most important such problems is the *data fusion problem* seen in wireless sensor networks. In this case, we assume we have a set of sensors that are observing a *dynamical process* whose global state, $x(t)$, satisfies a linear differential equation of the form, $\dot{x}(t) = Ax(t) + w(t)$ and $y(t) = Cx(t) + v(t)$ for $t \in [0, \infty)$ where A and C are appropriately dimensioned system matrices. We assume that the process is driven by a white noise process, $w(t)$. The ith sensor observes the component

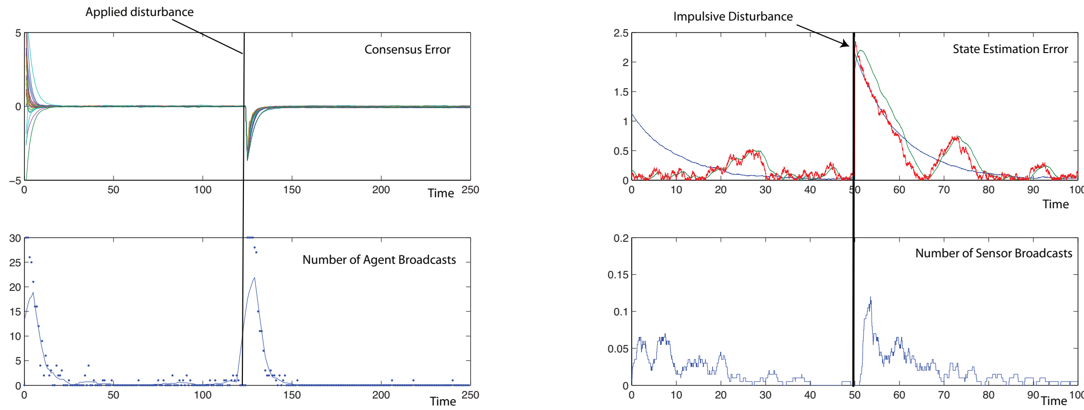


Figure 2: (left) Event-Triggered Consensus Filtering - (right) Event-Triggered Data Fusion

$y_i \in \Re$ of the output vector. In particular, we can see that $y_i(t) = c_i x(t) + v_i(t)$ where c_i is the ith row of C and v_i is a scalar noise process that is statistically independent from the other sensor noise. The data fusion problem has each sensor forward its local measurement y_i to a centralized data fusion center which combines these measurements using a Luenberger observer to obtain an estimate of the process state.

This problem is very straightforward if all sensors can continuously stream their data to the data fusion center. Such streaming, however, is impossible to realize over ad hoc wireless communication networks and so we consider the use of an event-triggering scheme in which each sensor autonomously decides when to broadcast its sensor data to the data fusion center.

Event-triggered broadcasts from each sensor are generated on the basis of a local estimate of the process' state. Of course, it is highly unlikely that the pair (A, c_i) will be observable, so rather than estimating the full process state, we have the ith sensor construct an observer for a "partial" state, $z^{(i)}$, that lies in the orthogonal complement of the unobservable subspace of (A, c_i). The state equations for the partial state are determined by constructing the standard form , $\left(\begin{bmatrix} A_{11}^{(i)} & 0 \\ A_{21}^{(i)} & A_{22}^{(i)} \end{bmatrix}, \begin{bmatrix} c_1^{(i)} & c_2^{(i)} \end{bmatrix} \right)$ for the partially observable system (A, c_i). We let $\hat{z}^{(i)}(t)$ denote the ith sensor's estimate of the partial state $z^{(i)}$ and we construct a Luenberger observer

$$\frac{d}{dt}\hat{z}^{(i)}(t) = A_{11}^{(i)}\hat{z}^{(i)}(t) + \ell_i \left(y_i(t) - c_1^{(i)}\hat{z}^{(i)}(t) \right) \tag{14}$$

Under this realization, the observable modes are decoupled from the unobservable modes. Letting $e_i(t) = y_i(t) - c_1^{(i)}\hat{z}^{(i)}(t)$ denote the observation error at time t, we use an event trigger of the form $\alpha(\|e_i(t)\|) \leq \beta(\|\hat{e}_i\|)$ where $\hat{e}_i$ is the observation error at the last broadcast time and α, β are class $\mathcal{K}$ functions. The violation of this inequlaity is used to determine when the ith sensor should broadcast its local measurement, y_i, to the data fusion center.

We simulated such a distributed data fusion sensor in Matlab for a process that had three states and two sensors. The results are shown on the right side of figure 2. The original process was unobservable under either of these sensors. An impulse disturbance was injected into the state halfway through the simulation to see if the system could reject the disturbance. The top graph shows that the estimated states asymptotically converge to the true state. The bottom plot shows the average number of agents that were broadcasting in a fixed time interval. Note that again as the system's estimation error goes to zero, the broadcast frequency begins to decrease to zero. In looking at the total number of broadcasts between the sensors and the data fusion center, this example again shows that event-triggering reduces the number of broadcasts by at least one order of magnitude over continuous streaming of the sensor data.

Research Challenges: The preceding discussion suggests that event-triggering greatly reduces the communication complexity associated with distributed estimation in networked systems. These results are preliminary and have yet to be published by our group. These results, however, seem promising and leave open a number of research issues that can be addressed in this project.

The event triggers used above were suggested by equation 4. We have yet to formally prove that this is the "optimal" event to be used. To study this type of event-triggering, we are currently using the $\mathcal{K}L$ function approach originally suggested in [56]. In this case, it was shown that a sampled data system would be input-to-state stable provided we could guarantee two conditions. The first condition is that the estimation error's rate of growth between consecutive broadcasts is bounded by a suitable class $\mathcal{K}$ function. The second condition is that the discrete-time system formed by only considering the broadcasts is bounded by a class $\mathcal{K}L$ function. The results in [56] originally considered these concepts in the context of control. We're working to apply these to the estimation problem. Using this approach, we believe it should be possible to determine an event-trigger that is *optimal* in the sense of minimizing broadcast frequency subject to a constraint on the estimation error's convergence rate.

A number of interesting research directions come out of the preceding work in consensus filtering. We have, as yet, no formal results on the scaling properties of event-triggered consensus filtering or data fusion. While the simulation results given above suggest that event-triggered estimation may scale well, it would be better to determine the actual scaling exponents for various measures of network size.

The preceding results assumed a fixed communication network topology. Obviously it would be valuable to study the impact that real-life communication uncertainties have on the performance of the event-triggered estimator. In particular, we would like to study the impact of delays, data dropouts, and quantization on estimator performance.

In our prior discussion, we left unclear the precise nature of the centralized algorithm used for data fusion. Our initial work assumed that all broadcasts were fused into a state estimate at a central fusion center. This approach, however, would create a communication bottleneck at the fusion center. Another approach would use a distributed fusion approach in which sensors or fusion centers "detect" which sensor streams have the most information and then only forward those high-quality sensor streams. This approach is similar to decentralized detection schemes used in sensor networks [8]. This idea could also be extended to a hierarchical scheme in which sensors automatically cluster themselves to "clusterheads" which then selectively broadcast their estimates to the data fusion center. A key issue of future concern would be how such clusters would automatically form given the volatile nature of the wireless communication graph.

5 Event-triggered Optimal Control of Networked Systems

This project will develop event-triggered implementations of receding horizon controllers for nonlinear networked systems. Our earlier work [84, 82, 87] has already developed event-triggered mechanisms for networked linear systems. For linear systems, that earlier work showed that we could dramatically extend the sampling period of the networked control system. While that earlier work also applies to a restricted class of nonlinear systems, it has not addressed the optimality of event-triggered feedback. We want to extend these methods to study optimal event-triggered control using receding horizon control of networked nonlinear systems as a starting point.

Preliminary Work: This subsection demonstrates that event-triggering can reduce the communication complexity of a distributed receding horizon controller (RHC) that was developed in [16] for a walking robot. The example shows that our earlier work may be extended to nonlinear optimal control systems. The robot's configuration is shown to the right of figure 3, where θ_1 is the angle between the thighs, θ_2 is the angle of the right knee and θ_3 is the angle of the left knee. The joint angles satisfy the following differential equations.

$$\begin{aligned}
\ddot{\theta}_1 &= 0.1\left[1-5.25\theta_1^2\right]\dot{\theta}_1 + u_1(t) \\
\ddot{\theta}_2 &= 0.01\left[1-p_2(\theta_2-\theta_{2e})^2\right]\dot{\theta}_2 - 4(\theta_2-\theta_{2e}) - 0.057\theta_1\dot{\theta}_1 + 0.1(\dot{\theta}_2-\dot{\theta}_3) + u_2 \\
\ddot{\theta}_3 &= 0.01\left[1-p_3(\theta_3-\theta_{3e})^2\right]\dot{\theta}_3 - 4(\theta_3-\theta_{3e}) - 0.057\theta_1\dot{\theta}_1 + 0.1(\dot{\theta}_3-\dot{\theta}_2) + u_3
\end{aligned}$$

where u_i is the applied torque at the ith joint, p_1, p_2, θ_{2e} and θ_{3e} are constants establishing the stable limit cycle of this system.

For appropriately chosen parameters (p_1, p_2, θ_{2e}, and θ_{3e}), the system exhibits a stable limit cycle when $u_2 = u_3 = 0$. By switching these parameters, we can force the robot to walk. The control problem is to transition the robot's configuration from a regular walk to a standing position where $\theta_1 = 0$. This will be

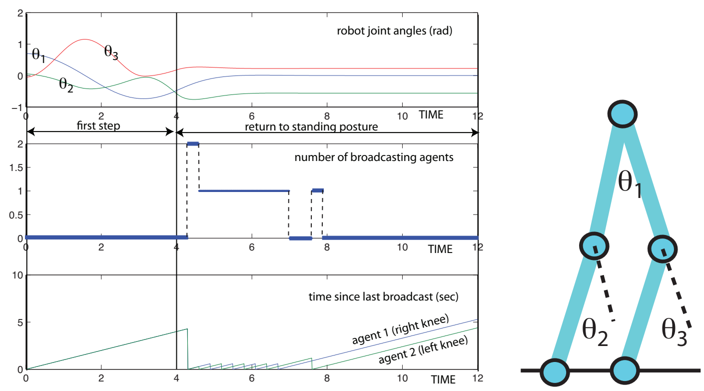


Figure 3: Walking Robot

accomplished by introducing controls at the knee joints (u_2 and u_3) which move the system's state to the standing position. Let each joint's local state be denoted as $x_i = \left[\begin{array}{cc} \dot{\theta}_i & \theta_i \end{array} \right]^T$. We'd like to accomplish this maneuver in an optimal manner by selecting u to minimize the performance functional

$$J(u) = \sum_{i=1}^{3} \int_0^\infty \left(x_i^T(\tau) Q_i x_i(\tau) + u_i^T(\tau) u_i(\tau) \right) d\tau \tag{15}$$

where Q_i is an appropriately chosen symmetric positive definite matrix. One well known heuristic for solving this problem is receding horizon control (RHC) [54]. Receding horizon control addresses the problem in equation 15 by solving a sequence of finite-horizon problems. The finite horizon problem of interest seeks to minimize

$$J(u;T) = \sum_{i=1}^{N} \int_{B_i[k]}^{B_i[k]+T} \left(x_i^T Q_i x_i + u_i^T u_i \right) d\tau + x_i^T(B_i[k]) M_i x_i(B_i[k]) \tag{16}$$

where the optimization is subject to the control u being admissible and stabilizing. The sequences $B_i[k]$ represent a set of times when we update the control on the ith joint and T represents the horizon over which the optimal control is computed. In this case, M_i, is a positive definite matrix that is chosen to satisfy a certain algebraic Riccati equation. Under these conditions the terminal constraint in equation 16 becomes a reasonable approximation to the value function of the original infinite horizon problem [25] guaranteeing that the RHC solution is indeed stabilizing.

We're interested in implementing the RHC in a distributed manner. Earlier work with distributed RHC will be found in [1, 26, 16]. This example follows [16] where we use a linearization of the original system equations as the basis for solving the RHC problem. The RHC controller for this linearized system can be obtained by determining the appropriate linear quadratic regulator (LQR) for the linearized system. Note that the LQR treats the broadcast measurement as a constant disturbance whose impact on the local state is handled by introducing an integrator into the controller. Under this approach, we end up with a controlled system which is identical to the structure suggested in our original paper [84] and the event trigger takes the form $\|x_i - \hat{x}_i\|_2 \leqslant \rho \|\hat{x}_i\|_2$ where ρ is an appropriately chosen constant. The right knee joint would broadcast its local state to the left knee when the above condition is violated. A similar event-trigger is used for the left knee broadcast to the right knee.

The plot in figure 3 shows the simulation results for the walking robot. The robot is walking from 0 to 4 seconds, after which the event-triggered RHC controller is used to drive the robot to its standing position. The top plot shows that the robot successfully achieves its standing position. The middle plot shows the total number of broadcasting agents as a function of time and the bottom plot shows the time since last broadcast for the right and left knee. The middle plot shows that broadcasting between the left and right knee begins after the robot starts moving to its standing configuration. At that time, both agents broadcast to each other for short period of time and eventually after the robot reaches its standing configuration, the joints cease broadcasting to each other. As in the other examples shown throughout this proposal, the results show that event-triggering again reduces the number of messages that need to be exchanged between the two agents at the robot's knees.

Research Challenges: Our earlier work [82, 87] focused on the stability of event-triggered networked systems; whereas in the preliminary work, we show that event-triggered formalisms can be used to optimally control nonlinear systems in a manner that does not compromise the optimality of the solution. These results suggest we should shift our emphasis from stability to optimality in nonlinear distributed systems.

The event-triggered RHC controller described above has no inequality constraints on the control or state. In some of the applications we're studying, it is important the distributed control also satisfy these types of inequality constraints. So this project will also consider how to integrate constraints on state and control into the distributed RHC system.

Our earlier work provided a good understanding of the impact that packet dropouts and delays have on overall performance of the distributed controller. But the relationship between dropouts, delays and sampling period is less clear. Future work will try and determine performance curves relating delay/dropout rates to sampling period. It would also be valuable to extend these results to quantized feedback also.

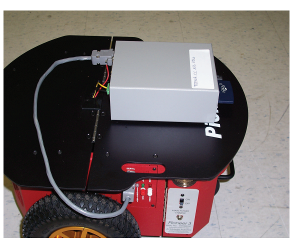
The above results assume that the communication network and physical interconnections between agents have the same graph. In general, this may not be the case. Some of our earlier work [87] formally modelled this more general case, but to date we have no results characterizing what happens to overall network performance when the two graphs don't match. This project will try to address that issue.

The distributed receding horizon controller in [16] really presume weak physical coupling between agents so that "decentralized" controllers can be used. The event-triggered analysis in our work [84] also assumes weak coupling. These assumptions can be extremely conservative and it would be valuable to determine event-triggers for less conservative distributed controllers.

6 Evaluation Testbeds

The event-triggered distributed optimization, estimation, and control algorithms developed in this project will be evaluated on three different evaluation testbeds described below.

6.1 Multi-robot Testbed: A multi-robot testbed was constructed in 2005 as part of an earlier NSF project. We will use this testbed to evaluate formation stabilization of networked robotic systems using event-triggered receding horizon controllers.



The multi-robot testbed consists of three ActiveMedia Pioneer robots. The Pioneer robot uses acoustic proximity sensors to detect obstacles around it. It uses gyro-corrected wheel encoders to determine its local position and orientation. It is controlled through an on-board embedded Linux PC that communicates to the Internet over an 802.11 wireless LAN card. Low-level robot motion control is programmed using a set of C++ classes developed by ActivMedia. The use of these classes greatly simplifies the job of developing higher-level supervisory control programs for robot swarms. The vehicles are currently controlled over the Internet using sockets. The vehicles are treated as servers and the remote TCL/TK user is treated as a client. The TCL/TK client allows the remote user to issue movement commands to the robot while automatically displaying the vehicles current position, orientation, and proximity sensor data in a graphical window. Peer-to-peer radio communication between robots can be implemented over the wifi connection or through an embedded radio such as the Mica2.

The robots provide a good testbed for studying event-triggered control of decoupled agents. In this testbed, an agent's broadcasts will be triggered by localized threshold conditions such as those described in equation 4. We currently plan to use this testbed to study event-triggered implementations of two specific problems. The first problem involves event-triggering in the swarming under consensus model [67, 68, 37]. The second problem involves distributed RHC of robot formation [17]. We will also use this testbed to develop some demonstrations that can be used in on-going K-12 outreach efforts.

The dynamics of the testbed can be described as follows. Let q_i denote the ith robot's position, then the dynamics of that agent can be expressed as $\ddot{q}_i = f_i(\dot{q}_i, q_i) + g_i(\dot{q}_i, q_i)u_i$ where f_i and g_i are appropriately dimensioned vector fields representing the uncontrolled dynamics of the robots. u_i is the control which is generated by the following equations $u_i(t) = k_i(\dot{q}_i, q_i) + \sum_{j\in N_i} L_{ij}(\hat{\dot{q}}_j, \hat{q}_j)$ where $\hat{\dot{q}}_j$ and $\hat{q}_j$ are the local states of the jth agent that are broadcast to the ith agent. k_i is the ith agent's decentralized local controller. L_{ij} is a coupling controller that is used to "coordinate" the behavior of the ith and jth agents. The main problem concerns the selection of the event-trigger and the controllers assuring some specified level of overall group performance. As noted above, we propose studying two applications with this testbed; swarming under consensus [37] and RHC formation stabilization [17].

Swarming under Consensus: The swarming under consensus model considers a dynamical system that satisfies the differential equations

$$\begin{aligned}\dot{x}_i(t) &= (y(t) - z_i(t)) + \sum_{j\in N_i} g(\|x_i(t) - x_j(t)\|)(x_i(t) - x_j(t)) \\ \dot{z}_i(t) &+ (y(t) - z_i(t)) + \sum_{j\in N_i} (x_j(t) - z_i(t)) + \sum_{j\in N_i} (z_j(t) - z_i(t))\end{aligned}$$

x_i denotes the physical state of the ith robot (i.e. $x_i = [\dot{q}_i, q_i]$) and z_i denotes the ith agent's estimate of the center of the robot swarm. The first equation given above is the well known swarm dynamic proposed by Viscek [74]. This model has been used to study animal swarms [50]. With appropriately selected interaction functions, g, the swarm asymptotically stabilizes to a formation [19] in which the agents uniformly distributed themselves in a tight ball. The second equation is the consensus filter [2]. In the swarming under consensus model, the consensus filter determines a distributed estimate of the swarm's center and that estimate is then used to guide the swarm toward a desired destination.

One problem that was seen in the original swarming under consensus model [37] was that the swarm equation assumes continuous communication between neighboring agents. A more realistic approach would have neighboring agents use an event-triggering strategy to broadcast their local states to their neighbors. We propose implementing an event-triggered form of the consensus-swarming system on our robotic testbed. Of particular interest will be the impact that variations in network topology have on the swarm's stability as well as the way in which communication cost scales with swarm size.

RHC Formation Stabilization: The other problem to be studied with this testbed is the formation stabilization problem studied in [17]. This problem is to develop a receding horizon controllers, k_i and L_{ij} that minimize the integrated cost

$$J(u) = \sum_{i=1}^{N} \int_0^T \left(\sum_{j\in N_i} \alpha \|q_i - q_j + d_{ij})\|^2 + \beta \|\dot{q}_i\|^2 + \gamma \|u_i\|^2 \right) d\tau + \left\| \frac{1}{N} \sum_{i=1}^{N} q_i(T) - q_d \right\|^2$$

where α, β, and γ are weighting constants. This optimal control problem seeks to stabilize the interagent distance d_{ij} while minimizing vehicle velocities and control effort. This optimization is done over a finite time interval, T, with a terminal constraint that minimizes the center of the swarm $\frac{1}{N}\sum_i q_i$ at a desired terminal position q_d.

We propose using the event-triggered RHC algorithm discussed in section 5 to solve the above formation control problem. We are particularly interested in seeing how the resulting RHC controllers differ from the event-triggered swarming algorithms given above. We'd like to integrate communication costs directly into the above problem to experimentally determine how this effects overall communication complexity. Note that the determination of the swarm's center (last term in above cost function) requires information from all agents. Such information can be gathered using the consensus filter, but how do the coupled dynamics of the consensus filter and this RHC controller interact and effect the stability of the swarm?

6.2 CSOnet Testbed: For the past four years, the project PI has been working with a company (EmNET LLC) to build a metropolitan scale wireless sensor-actuator network called CSOnet. CSOnet [60, 77, 53] is being built in the City of South Bend Indiana to control the frequency of combined sewer overflow (CSO) events. The construction of the initial prototype was funded by Indiana's 21st Century Technology Fund with subsequent funding in 2007 to expand the system to cover the entire city of South Bend. At present the system consists of a 110+ sensor network covering a 13,000 acre area with an additional 18-20 actuation points being added by summer 2009.

CSO events occur when stormwater flows overload a sewer's capacity, thereby forcing city enginners to divert excess storm water into a river [48] [49]. This diversion is called a CSO event and because the diverted waters are highly impacted with chemical and biological contaminants these events constitute a significant environmental hazard. One way for addressing the CSO problem is to store excess water in unused parts of the city's sewer system. This approach is called in-line storage and its implementation requires real-time monitoring and control of the water flows in the sewer system. Current approaches to in-line storage use a centralized model-predictive control method [47] [15] [7] in which all sensors transmit their data over a *supervisory control and data acquisition* (SCADA) network to a central computer. This computer makes a global control decision and broadcasts its decision over the SCADA network to the actuation points within the system. Prior implementations of this centralized approach in Quebec Canda [57] and Milwaukee Wisconsin [61] demonstrated that the cost scales poorly in large metropolitan areas.

CSOnet addresses the issue of cost scalability by distributing the control decisions throughout the entire sewer system. This means that actuation (control) decisions are made locally at the actuation point using information from neighboring actuation points. Because decision making is distributed, we no longer need a SCADA network and can, instead, use a lower cost mesh radio network. CSOnet is therefore a wireless sensor-actuator network that uses controllers distributed across the physical sewer network.

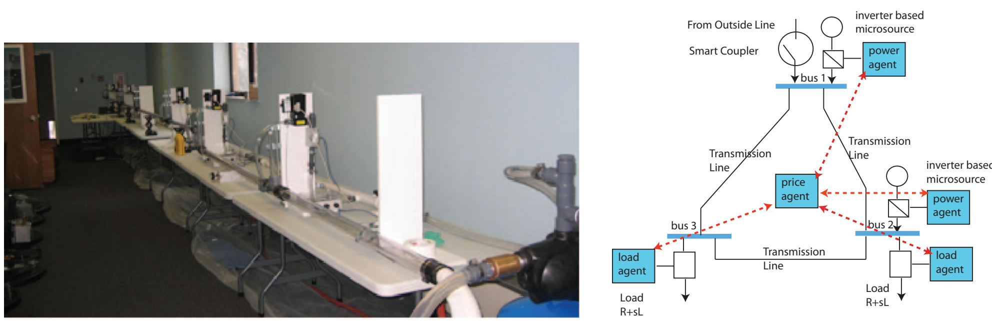


Figure 4: (left) CSOnet Testbed - (right) Odyssian Mesh Microgrid Simulation Model

The current controller for CSOnet is a decentralized controller that only uses discrete events to coordinate actions between nodes. It was realized very early on that better performance could be achieved if real-time information from neighboring nodes was available for control also. CSOnet's current middleware, however, is unable to provide hard real-time guarantees on data streams between adjacent nodes due to the unreliability of the wireless communication network. The objective of this testbed is to see whether or not an event-triggered implementation of distributed CSOnet controllers would be feasible for the system's wireless network. In particular, we propose developing distributed event-triggered RHC controllers for the CSOnet system. The resulting algorithms will be validated on the hardware testbed (see figure 4) currently at our industrial collaborator's (EmNet LLC) offices. EmNet will commit time and financial resources to assist in the project's completion (see EmNet's letter of commitment).

6.3 Microgrid Simulation: Microgrids [29] are power distribution networks in which users and generators are in close proximity. Generation is often done using renewable microsources such as photovoltaic cells or wind turbines. The microgrid is often connected to the main power grid through an intelligent coupling switch. This switch can be disconnected from the main grid when the main grid's power quality is no longer acceptable. This disconnect results in the "islanding" of the microgrid. A major challenge involves assuring the microgrid's transient stability in the presence of disconnects. While operating as an island, microgrids must address the usual issues regarding generation dispatch and load shedding. Due to their limited inertia, however, these issues are more critical for microgrids than conventional power grids.

Since March of 2008, this project's PI has been working with a local company (Odyssian LLC) to develop advanced distribution and control architectures for microgrids. This work was part of an Army SBIR/STTR contract to Odyssian LLC. The proposed control architecture was a two-level system whose lowest level used local microsource controllers [28] developed at the University of Wisconsin Madison for their CERTS microgrid [29]. The top level of the architecture [32] consisted of a set of ”intelligent” agents that communicated over a wireless mesh radio network. These agents were used to control power dispatch and load shedding using the event-triggered formalisms.

A detailed simulation for this control architecture was developed for a small mesh 30 kW microgrid that had two microsources and two resistive loads (see figure 4). This simulation was developed with the help of our collaborators at University of Illinois Urbana-Champaign (P. Chapman) and University of Wisconsin Madison (R. Lasseter). We developed and simulated a multi-agent system solving a power dispatch problem that was similar to the NUM problem described in section 3. As above, we used an event-triggered approach to solve this problem and demonstrated through simulation that our approach reduced the communication traffic between agents while preserving the grid’s transient stability. We intend to use what we learned in developing this mesh microgrid simulation to study the use of event-triggering in microgrid control. We hope to build on our earlier work with Odyssian to obtain a better understanding of how event-triggering and networked control might be used in electrical power grids. Odyssian is extremely interested in seeing whether the results of this NSF project may be used to commercialize their earlier STTR work (see letter of support from Odyssian).

7 Curriculum Development Activities

This project will develop new curriculum for graduate students and K-12 education. In the summer of 2009, the PI will be lecturing on event-triggered control to a Ph.D. summer school at the University of Siena, Italy. The substance of this lecture will be expanded into a set of on-line lectures that can be accessed from the world-wide web. A print version of these lectures will also be included in a book on networked control systems. The lectures will provide fundamental instruction on event-triggered optimization, estimation, and control. The lectures will be used to instruct the students of the PI and his collaborators. In making these lectures freely available, we hope to more broadly disseminate the project’s results. In fall of 2008, the PI began working with a local middle school in South Bend Indiana to discuss some of our recent work with embedded control of sewer networks [53]. It was observed that many of these students are taking part in an after school program using Lego’s mind-storm system to develop autonomous robots. Since part of this project involves using our multi-robotic testbed to study event-triggered control, we could with a little additional effort develop a demonstration using the project’s multi-robotic testbed to show how their mind storm robots would scale up. So as part of this project, we intend to develop graphical and programming interfaces that these students can use to control our more powerful robots. We feel that this would help students see beyond the ”toy-like” nature of the mind-storm robots and would stimulate their interest in entering systems engineering.

8 Results from Prior NSF Sponsored Research

Dr. Lemmon received prior support under NSF grants ECS04-00479 ”Scalable Decentralized Control over Ad Hoc Sensor Actuator Networks” (2004-2007 - $210,000) and CNS07-20457 ”Real-time Environments for the Self-triggered Decentralized Control of Networked Dynamical Systems” (2007-2010 - $160,000). These grants studied networked control systems and real-time systems. Our earliest work in this area developed a power spectral analysis for linear control systems with dropped feedback messages [39] [40] [42]. A Markov-chain model for data dropouts in control was considered in [41] [35]. These Markov-chain models were later used in developing a novel real-time QoS constraint for control systems [45] [44] [24]. Later papers [34] [43] studied the stability of dynamically quantized feedback systems. We also studied the use of periodic communication logics for multi-agent systems [66] and the interconnection of swarms with consensus filters in [36] [68]. We developed an event-triggered and self-triggered approach to feedback control [89] [83] [82] [86] [84] [85] [33]. Event-triggering was applied to utility maximization problems in [78] [79]. We also examined extensions of elastic scheduling as they might apply to networked control [10] [11] [12] [9] [23]. This project is most closely related to NSF grant CNS07-20457. Project CNS07-20457 focuses on the impact that self-triggered feedback [85] has on real-time scheduling. The work proposed in this project is significantly different from the work in CNS07-20457. The work proposed above focuses on system science issues associated with event-triggering for distributed optimization, estimation, and control. CNS07-20457, on the other hand, focuses on the impact of event-triggering on real-time computing systems, specifically with regard to scheduling algorithms.

References

[1] L. Acar. Boundaries of the receding horizon control for interconnected systems. *Journal of Optimization Theory and Applications*, 84(2), 1995.

[2] R. Olfati-Saber adn J. Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the IEEE Conference on Decision and Control*, 2005.

[3] K.E. Arzen. A simple event-based pid controller. In *Proceedings of the 14th IFAC World Congress*, 1999.

[4] K.J. Astrom and B.M. Bernhardsson. Comparison of riemann and lebesgue sampling for first order stochastic systems. In *Proceedings of the IEEE Conference on Decision and Control*, 1999.

[5] A. R. Bergen and V. Vittal. *Power Systems Analysis*. Prentice-Hall, 2 edition, 2000.

[6] D. Carnevale, A.R. Teel, and D. Nesic. A lyapunov proof of improved maximum allowable transfer interval for networked control systems. *IEEE Transactions on Automatic Control*, 52:892–897, 2007.

[7] G. Cembrano, J. Quevedo, M. Salamero, V. Puig, J. Figueras, and J. Marti. Optimal control of urban drainage systems. A case study. *Control Engineering Practice*, 12(1):1–9, 2004.

[8] J-F Chamberland and V.V. Veeravalli. Decentralized detection in sensor networks. *IEEE Transactions on Signal Processing*, 51(2):407–416, 2003.

[9] T. Chantem, X. Hu, and M.D. Lemmon. Generalized elastic scheduling. In *IEEE Real Time Systems Symposium*, 2006.

[10] T. Chantem, X.S. Hu, and M.D. Lemmon. Generalized elastic scheduling for real-time tasks. to appear in IEEE Transactions on Computers, 2008.

[11] T. Chantem, X. Wang, M.D. Lemmon, and X. Hu. Period and deadline selection for schedulability in real-time systems. In *Proceedings of the European Conference on Real-time Systems*, 2008.

[12] T. Chantem, X.Hu, and M.D. Lemmon. Period and deadline section problem for real-time systems. In *Proceedings of the IEEE Real-time Systems Symposium - work in progress track*, 2007.

[13] L. Chen, SH Low, and JC Doyle. Joint congestion control and media access control design for ad hoc wireless networks. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 3, 2005.

[14] M. Chiang and J. Bell. Balancing supply and demand of bandwidth in wireless cellular networks: utility maximization over powers and rates. *Proc. IEEE INFOCOM*, 4:2800–2811, 2004.

[15] S. Duchesne, A. Mailhot, and J.P. Villeneuve. Global Predictive Real-Time Control of Sewers Allowing Surcharged Flows. *Journal of Environmental Engineering*, 130(5):526–534, 2004.

[16] W.B. Dunbar. A distributed receding horizon control algorithm for dynamically coupled nonlinear systems. In *IEEE Conference on Decision and Control*, 2005.

[17] W.B. Dunbar and R.M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558, 2006.

[18] N. Elia and S. Mitter. Stabilization of linear systems with limited information. *IEEE Transactions on Automatic Control*, 46(9):1384–1400, 2001.

[19] V. Gazi and K. Passino. Stability analysis of swarms. *IEEE Transactions on Automatic Control*, 48(4):692–697, 2003.

[20] D. Haussler. Learning conjunctive concepts in structural domains. *Machine Learning*, 4(1):7–40, 1989.

[21] W.P.M.H. Heemels, A.R. Teel, N. van de Wouw, and D. Nesic. Networked control systems with communication constraints: tradeoffs between sampling intervals, delays and performance. submitted to the 2009 European Control Conference (ECC), 2008.

[22] D. Hristu-Varsakelis and P.R. Kumar. Interrupt-based feedback control over a shared communication medium. In *Proceedings of the IEEE Conference on Decision and Control*, 2002.

[23] X. Hu, T. Chantem, and M.D. Lemmon. Optimal elastic scheduling. In *IEEE Real-time and Embedded Technology and Applications Symposium (works in progress track)*, 2006.

[24] X. Hu, D. Liu, M.D. Lemmon, and Q. Ling. Firm real-time system scheduling based on a novel qos constraint. In *Real Time Systems Symposium*, 2003.

[25] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding horizon control with a generalized terminal cost. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[26] D. Jia and B. H. Krogh. Distributed model predictive control. In *Proceedings of the American Control Conference*, 2003.

[27] F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, 1998.

[28] R. Lasseter. Control and design of microgrid components. Final Project Report - Power Systems Engineering Research Center (PSERC-06-03), 2006.

[29] R.H. Lasseter, A. Akhil, C. Mrnay, J. Stephens, J. Dagle, R. Guttronson, A. Meliopoulous, R. Yinger, and J. Eto. The certs microgrid concept. White paper for transmission reliability program, office of power technolgoies, U.S. Department of Energy, 2002.

[30] E.A. Lee. Modeling concurrent real-time processes using discrete-events. *Annals of Software Engineering*, 7:25–45, 1999.

[31] E.A. Lee and A. Sangiovanni-Vincentelli. A framework for comparing models of computation. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, 17(12):1217–1229, 1998.

[32] M.D. Lemmon. Final report for odyssian sttr project entitled "advanced distribution and control for hybrid intelligent power systems. Final Report submitted to Odyssian LLC, February, 2009.

[33] M.D. Lemmon, T. Chantem, X. Hu, and M. Zyskowski. On self-triggered full information h-infinity controllers. In *Hybrid Systems: computation and control*, 2007.

[34] M.D. Lemmon and Q. Ling. Control system perforamnce under dynamic quantization: the scalar case. In *IEEE Conference on Decision and Control*, 2004.

[35] M.D. Lemmon, Q. Ling, and Y. Sun. Overload management in sensor-actuator networks used for spatially-distributed control systems. In *1st ACM Conferenced on Embedded Network Sensor Systems (SenSys03)*, 2003.

[36] M.D. Lemmon and R. Sun. Performance rate curves for dynamically quantized feedback systems. In *IEEE Conference on Decision and Control*, 2006.

[37] M.D. Lemmon and Y. Sun. Cohesive swarming under consensus. In *IEEE Conference on Decision and Control*, 2006.

[38] D. Liberzon. On stabilization of linear systems with limited information. *IEEE Transactions on Automatic Control*, 48:304–307, 2003.

[39] Q. Ling and M.D. Lemmon. Robust performance of soft real-time networked control systems with data dropouts. In *IEEE Conference on Decision and Control*, 2002.

[40] Q. Ling and M.D. Lemmon. Optimal dropout compensation in networked control systems. In *IEEE Conference on Decision and Control*, 2003.

[41] Q. Ling and M.D. Lemmon. Soft real-time scheduling of networked control systems with dropouts governed by a markov chain. In *American Control Conference*, 2003.

[42] Q. Ling and M.D. Lemmon. Power spectral analysis of networked control systems with data dropouts. *IEEE Transactions on Automatic Control*, 49(6):955–959, 2004.

[43] Q. Ling and M.D. Lemmon. Stability of quantized control systems under dynamic bit assignment. *IEEE Transactions on Automatic Control*, 50(5):734–740, 2005.

[44] D. Liu, X. Hu, M.D. Lemmon, and Q. Ling. Scheduling tasks with markov-chain constraints. In *17th Euromicro Conference on Real-time Systems*, 2005.

[45] D. Liu, X. Hu, M.D. Lemmon, and Q. Ling. Firm real-time system scheduling based on a novel qos constraint. *IEEE Transactions on Computers*, 55(3):320–333, 2006.

[46] S.H. Low and D.E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking (TON)*, 7(6):861–874, 1999.

[47] M. Marinaki and M. Papageorgiou. A non-linear optimal control approach to central sewer network flow control. *International Journal of Control*, 72(5):418–429, 1999.

[48] L.W. Mays. *Stormwater Collection Systems Design Handbook*. McGraw Hill, 2001.

[49] P.E. Moffa. *Control and Treatment of Combined Sewer Overflows*. Van Nostrand Reinhold, 1990.

[50] A. Mogilner and L. Edelstein-Keshet. A nonlocal model for a swarm. *Journal of Mathematical Biology*, 38:534–570, 1999.

[51] A.K. Mok and X. Feng. Towards compositionality in real-time resource partitioning based on regularity bounds. In *Proceedings of the 22nd IEEE International Real-Time Systems Symposium*, 2001.

[52] A.K. Mok, X. Feng, and D. Chen. Resource parition for real-time systems. In *Proceedings of the 7th IEEE Real-time Technology and Applications Symposium (RTAS)*, 2001.

[53] L. Montestruque and M.D. Lemmon. Csonet: a metropolitan scale wireless sensor actuator network. In *Proceedings of the International Workshop on Mobile Device and Urban Sensing (MODUS)*, 2008.

[54] M. Morari and J.H. Lee. Model predictive control: past, present and future. *Computers and Chemical Engineering*, 23:667–682, 1999.

[55] D. Nesic and A.R. Teel. Input-output stability properties of networked control systems. *IEEE Transactions on Automatic Control*, 49:1650–1667, 2004.

[56] D. Nesic, A.R. Teel, and E.D. Sontag. Formulas relating kl stability estimates of discrete-time and sampled-data nonlinear systems. *Systems and Control Letters*, 38:49–60, 1999.

[57] M. Pleau, H. Colas, P. Lavalee, G. Pelletier, and R. Bonin. Global optimal control of the quebec urban drainage system. *Environmental Modelling and Software*, 20:401–413, 2005.

[58] E. Polak. Stability and graphical analysis of first order of pulse-width modulated sampled data regulator systems. *IRE Trans. Automatic Control*, AC-6(3):276–282, 1963.

[59] J.W. Polderman and J.C. Willems. *Introduction to Mathematical Systems Theory: a behavioral approach*. Springer, New York, 1998.

[60] T. Ruggaber, J. Talley, and L. Montestruque. Using embedded sensor networks to monitor, control, and reduce cso events: A pilot study. *Environmental Engineering Science*, 24(2):172–182, 2007.

[61] N. Schultz S. Heinz and D. Wre. Milwaukee case study in example evolution of sewer control. In *Proceedings of World Water and Environmental Resources Congress*, 2006.

[62] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *Proceedings of the 24th IEEE International Real-Time Systems Symposium (RTSS)*, pages 2–13, 2003.

[63] I. Shin and I. Lee. Compositional real-time scheduling framework. In *Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS)*, pages 56–67, 2004.

[64] E.D. Sontag and Y. Wang. On characterizations of the input-to-state stability property. *Systems and Control Letters*, 24:351–359, 1995.

[65] E.D. Sontag and Y. Wang. New characterizations of input-to-state stability. *IEEE Transactions on Automatic Control*, 41:1283–1294, 1996.

[66] Y. Sun and M.D. Lemmon. Periodic communication logics for the decentralized control of multi-agent systems. In *IEEE Conference on Control Applications*, 2005.

[67] Y. Sun and M.D. Lemmon. Convegence of consensus filtering under network throughput limitations. In *Proceedings of the IEEE Conference on Decision and Control*, 2007.

[68] Y. Sun and M.D. Lemmon. Swarming under perfect consensus using integral action. In *American Control Conference*, 2007.

[69] M. Tabbara, D. Nesic, and A.R. Teel. Stability of wireless and wireline networked control systems. *IEEE Transactions on Automatic Control*, 52(9):1615–1630, 2007.

[70] P. Tabuada and X. Wang. Preliminary results on state-triggered scheduling of stabilizing control tasks. In *IEEE Conference on Decision and Control*, 2006.

[71] S. Tatikonda and S.K. Mitter. Control under communication constraints. *IEEE Transactions on Automatic Control*, 49(7):1056–1068, 2004.

[72] Y.Z. Tsypkin. *Relay Control Systems*. Cambridge University Press, 1984.

[73] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[74] T. Viscek, A. Czirook, B. Ben-Jacob, C. Cohen, and I. Shochet. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, 76(6):1226–1229, 1995.

[75] P. Voulgaris. Control of asynchronous sampled data systems. *IEEE Transactions on Automatic Control*, 39(7):1451–1455, 1994.

[76] G.C. Walsh, H. Ye, and L. Bushnell. Stability analysis of netowrked control systems. In *Proceedings of American Control Conference*, pages 2876–2880, 1999.

[77] P. Wan and M.D. Lemmon. Distributed flow control using embedded sensor-actator networks for the reduction of combined sewer overflow (cso) events. In *Proceedings of the IEEE Conference on Decision and Control*, 2007.

[78] P. Wan and M.D. Lemmon. Distributed network utility maximization using event-triggered augmented lagrangian methods. submitted to American Control Conference, 2009.

[79] P. Wan and M.D. Lemmon. Distributed network utility maximization using event-triggered barrier methods. submitted to the European Control Conference, 2009.

[80] P. Wan and M.D. Lemmon. Event-triggered distributed optimization in sensor networks, 2009.

[81] E. Wandeler and L. Thiele. Interface-based design of real-time systems with hierarchical scheduling. In *Proceedings of the 12th IEEE Real-time and Embedded Technology and Applications Symposium (RTAS)*, pages 243–252, 2006.

[82] X. Wang and M.D. Lemmon. Decentralized event-triggering broadcast over networked systems. In *Proceedings of Hybrid Systems: computation and control*, 2008.

[83] X. Wang and M.D. Lemmon. Event design in event-triggered feedback control systems. In *Proceedings of the IEEE Conference on Decision and Control*, 2008.

[84] X. Wang and M.D. Lemmon. Event-triggered broadcasting across distributed networked control systems. In *Proceedings of the American Control Conference*, 2008.

[85] X. Wang and M.D. Lemmon. Self-triggered feedback control systems with finite-gain l2 stability. to appear in the IEEE Transactions on Automatic Control, 2008.

[86] X. Wang and M.D. Lemmon. State based self-triggered feedback control systems with l2 stability. In *Proceedings of the 17th IFAC World Congress*, 2008.

[87] X. Wang and M.D. Lemmon. Event-triggering in distributed networked systems with data dropouts and delays. In *Proceedings of Hybrid Systems: computation and control*, 2009.

[88] X. Wang and M.D. Lemmon. Finite-gain l2 stability in distributed event-triggered networked control systems with data dropouts. submitted to the European Control Conference, 2009.

[89] X. Wang and M.D. Lemmon. Self-triggered feedback systems with state-independent disturbances. submitted to the American Control Conference, 2009.

[90] Y. Xue, B. Li, and K. Nahrstedt. Optimal resource allocation in wireless ad hoc networks: a price-based approach. *IEEE Transactions on Mobile Computing*, 5(4):347–364, 2006.

[91] W. Zhang, M.S. Branicky, and S.M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, 21:84–99, 2001.