Passivity-Based Design of Wireless Networked Control Systems for Robustness to Time-Varying Delays

Nicholas Kottenstette, Xenofon Koutsoukos, Joseph Hall, Janos Sztipanovits
ISIS/Vanderbilt University

Panos Antsaklis
University of Notre Dame

Abstract

Real-life cyber-physical systems, such as automotive vehicles, building automation systems, and groups of unmanned vehicles are monitored and controlled by networked control systems. The overall system dynamics emerges from the interaction among physical dynamics, computational dynamics, and communication networks. Network uncertainties such as time-varying delay and packet loss cause significant challenges. This paper proposes a passive control architecture for designing wireless networked control systems that are insensitive to network uncertainties. We describe the architecture for a system consisting of a robotic manipulator controlled by a digital controller over a wireless network and we show that the system is stable even in the presence of time-varying delays. We present simulation results that demonstrate the advantages of the architecture with respect to stability and performance and show that the system is insensitive to network uncertainties.

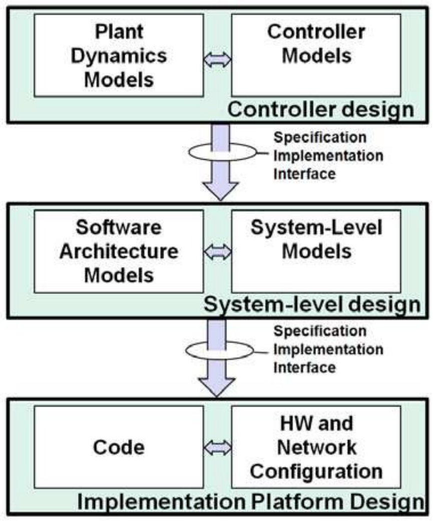
Plant Dynamics Models
Controller Models
Controller design
Specification Implementation Interface
Software Architecture Models
System-Level Models
System-level design
Specification Implementation Interface
Code
HW and Network Configuration
Implementation Platform Design

Figure 1. Simplified CPS design flow.

1 Introduction

The heterogeneous composition of computing, sensing, actuation, and communication components has enabled a modern grand vision for real-world Cyber Physical Systems (CPSs). Real-world CPSs, such as automotive vehicles, building automation systems, and groups of unmanned air vehicles are monitored and controlled by networked control systems and the overall system dynamics emerges from the interaction among physical dynamics, computational dynamics, and communication networks. Design of CPSs requires controlling real-world system behavior and interactions in dynamic and uncertain conditions.

Figure 1 represents a simplified model-based design flow of a CPS composed of a physical plant and a networked control system. In a conventional design flow, the controller dynamics is synthesized with the purpose of optimizing performance. The selected design platform (abstractions and tools used for control design in the design flow) is frequently provided by a modeling language and a simulation tool, such as MATLAB/Simulink [17, 18]. The controller specification is passed to the implementation design layer through a “Specification/Implementation Interface”. The implementation in itself has a rich design flow that we compressed here only in two layers: System-level design and Implementation platform design. The software architecture and its mapping on the (distributed) implementation platform are generated in the system-level design layer. The results - expressed again in the form of architecture and system models - are passed on through the next Specification and Implementation Interface to generate code as well as the hardware and network design. This simplified flow reflects the fundamental strategy in platform-based design [21]. Design progresses along precisely defined abstraction layers. The design flow usually includes top-down and bottom-up elements and iterations (not shown in the figure).

Effectiveness of the platform-based design largely depends on how much the design concerns (captured in the abstraction layers) are orthogonal, i.e., how much the design decisions in the different layers are independent. Heterogeneity causes major difficulties in this regard. The controller dynamics is typically designed without considering implementation side effects (e.g. numeric accuracy of computational components, timing accuracy caused by shared resource and schedulers, time varying delays caused by network effects, etc.). Timing characteristics of the implementation emerge at the confluence of design decisions in software componentization, system architecture, coding, and HW/network design choices. Compositionality in one layer depends on a web of assumptions to be satisfied by other layers. For example, compositionality on the controller design layer depends on assumptions that the effects of quantization and finite word-length can be neglected and the discrete-time model is accurate. Since these assumptions are not satisfied by the implementation layer, the overall design needs to be verified after implementation - even worst - changes in any layer may require re-verification of the full system.

An increasingly accepted way to address these problems is to enrich abstractions in each layer with implementation concepts. An excellent example for this approach is TrueTime [16] that extends MATLAB/Simulink with implementation related modeling concepts (networks, clocks, schedulers) and supports simulation of networked and embedded control systems. While this is a major step in improving designers' understanding of implementation effects, it does not help in decoupling design layers and improving orthogonality across the design concerns. A controller designer can now factor in implementation effects (e.g., network delays), but still, if the implementation changes, the controller may need to be redesigned.

Decoupling the design layers is a very hard problem and typically introduces significant restrictions and/or overdesign. For example, the Timed Triggered Architecture (TTA) orthogonalizes timing, fault tolerance, and functionality [8], but it comes on the cost of strict synchrony, and static structure. In an analogous manner, we propose to encompass passivity into traditional model-driven development processes in order to decouple the design layers and account for the effect of network uncertainties.

This paper is motivated by the rapidly increasing use of network control system architectures in constructing real-world CPSs and aims at addressing fundamental problems caused by networks effects, such as time-varying delay, jitter, limited bandwidth, and packet loss. To deal with these implementation uncertainties, we propose a model-design flow on top of passivity, a very significant concept from system theory [5]. A precise mathematical definition requires many technical details, but the main idea is that a passive system cannot apply an infinite amount of energy to its environment. The inherent safety that passive systems provide is fundamental in building systems that are insensitive to implementation uncertainties. Passive systems have been exploited for the design of diverse systems such as smart exercise machines [15], teleoperators [13], digital filters [6], and networked control systems [2, 10, 19].

Our approach advocates a concrete and important transformation of model-based methods that can improve orthogonality across the design layers and facilitate compositional component-based design of CPSs. By imposing passivity constraints on the component dynamics, the design becomes insensitive to network effects, thus establishing orthogonality (with respect to network effects) across the controller design and implementation design layers. The primary contributions of this paper are three-fold: (i) we present a passive control architecture for a system consisting of a robotic manipulator controlled by a digital controller over a wireless network, (ii) we provide analytical results that prove that our architecture ensures stability of the networked control system in the presence of time varying delays assuming that the communication protocols does not process duplicate transmissions, (iii) we implement the passive control architecture using MATLAB/Simulink/TrueTime models and present simulation results for a typical 6 degree-of-freedom robotic arm controlled by a digital controller over a 802.llb wireless network. Furthermore, the simulation demonstrate that the passivity-based design offers significant advantages with respect to stability and performance. Specifically, the proposed solution: (a) allows for lower sampling rates which reduces bandwidth requirements, (b) allows for higher gains which improve settling times, and (c) ensures robustness to time-varying network delays.

The work presented in the paper demonstrates that passivity can be exploited to account for the effects of network uncertainties, thus improving orthogonality across the controller design and implementation design layers and empowering model-driven development. Preliminary results of the approach have been presented in [12]. This paper contains a comprehensive description of the proposed architecture, theoretical analysis for stability in the presence of time-varying delays, and extensive simulation results based on systematic tuning of the control gains. It should be noted that passive structures offer additional advantages for robustness to finite length representations and saturation [6] but this paper focuses on network effects which is one of the most significant concerns in the development of CPSs.

2 Background on Passivity

There are various precise mathematical definitions for passive systems [10]. Essentially all the definitions state that the output energy must be bounded so that the system does not produce more energy than was initially stored. Continuous (discrete) *strictly-output passive* and *strictly-input passive* systems with finite gain have a special property in that they are L_2^m (l_2^m)-stable. Passive systems have a unique property that when connected in either a parallel or negative feedback manner the overall system remains passive. By simply closing the loop with any positive definite matrix, any discrete time passive plant can be rendered strictly output passive. This is an important result because it makes it possible to directly design low-sensitivity strictly-output passive controllers using the wave digital filters described in [6].

When delays are introduced in negative feedback configurations, the network is no longer passive. One way to recover passivity is to interconnect the two systems with wave variables. Wave variables were introduced by Fettweis in order to circumvent the problem of delay-free loops and guarantee that the implementation of wave digital filters is realizable [6]. Wave variables define a bilinear transformation under which a stable minimum phase continuous system is mapped to a stable minimum phase discrete-time system, and thus, the transformation preserves passivity.

Networks consisting of a passive plant and a controller are typically interconnected using power variables. Power variables are generally denoted with an effort and flow pair whose product is power. However, when these power variables are subject to communication delays, the communication channel ceases to be passive which can lead to instabilities. Wave variables allow effort and flow variables to be transmitted over a network while remaining passive when subject to arbitrary fixed time delays and data dropouts. If additional information is transmitted along with the continuous wave variables, the communication channel will also remain passive in the presence of time-varying delays [19]. More recently it has been shown that discrete wave variables can remain passive in spite of certain classes of time-varying delays and dropouts [2, 22]. In addition, a method which states how to properly handle time-varying discrete wave variables and maintain passivity has been developed in [10] and is used in our passive control architecture.

Before discussing our passive control scheme in Section 3 we recall the following definitions in regards to *passivity* and L_2^m*-stability*. These standard definitions which generalize “input-output” properties of many linear and nonlinear systems will be particularly useful when discussing the proof for Theorem 2 and understanding Corollary 1. In doing so, we choose to use the following compact notation.

$$\langle G(u), u\rangle_T \triangleq \int_0^T G(u(t))^\mathsf{T} u(t)dt$$

$$\|(G(u))_T\|_2^2 \triangleq \int_0^T G(u(t))^\mathsf{T} G(u(t))dt$$

We also denote $L_{2_e}^m(U)$ as the *extended* L_2^m space for the function $u(t) \in U$ in which $U \subset \mathbb{R}^m$ as all possible functions for a given $T \geq 0$ which satisfy:

$$\|(u)_T\|_2^2 < \infty.$$

In the limit as $T \to \infty$, then $u \in L_2^m(U)$ is any function which satisfies

$$\int_0^\infty u^\mathsf{T}(t)u(t)dt < \infty \text{ or more compactly,} \|u\|_2^2 < \infty.$$

Note also that $L_2^m(U) \subset L_{2e}^m(U)$.

Definition 1 *[23] Let $G : L_{2_e}^m(U) \to L_{2_e}^m(U)$ then for all $u \in L_{2_e}^m(U)$ and all real $T \geq 0$:*

I. G is passive if there exist a constant β such that (1) holds.

$$\langle G(u), u\rangle_T \geq -\beta \tag{1}$$

II. G is strictly-output passive if there exists some constants β and $\epsilon > 0$ such that (2) holds.

$$\langle G(u), u\rangle_T \geq \epsilon\|(G(u))_T\|_2^2 - \beta \tag{2}$$

Definition 2 *[23, Definition 1.2.3] Let $G : L_{2_e}^m(U) \to L_{2_e}^m(U)$, it is said to be L_2^m-stable if*

$$u \in L_2^m(U) \implies y = G(u) \in L_2^m(U), \tag{3}$$

and G is said to have finite-L_2^m-gain *if $\exists\gamma_q, \beta_q$ s.t. for all $T \geq 0$*

$$u \in L_{2e}^m(U) \implies \|(y)_T\|_2 \leq \gamma_q\|(u)_T\|_2 + \beta_q. \tag{4}$$

Any $G : L_{2_e}^m(U) \to L_{2_e}^m(U)$ which has *finite-L_2^m-gain* is *L_2^m-stable*.

The following theorem will allow us to complete the proof of our main result (Theorem 2) in which it is shown that the network control system depicted in Fig. 2 is *strictly-output passive* for any *passive* robot (plant).

Theorem 1 *[23, Theorem 2.2.14] Let $G : L_{2_e}^m(U) \to L_{2_e}^m(U)$ be strictly-output passive. Then G has finite L_2^m-gain.*

The definitions chosen for *passivity* are chosen from the input-output perspective similar to the definition for positive systems given in [24]. Numerous linear and non-linear systems satisfy the above *passivity* definition such as positive real systems and dissipative *passive* systems [7]. When a dissipative dynamical system can be described by a Hamiltonian (the sum of kinetic and potential energy, $\mathcal{H} = \mathcal{T} + \mathcal{V}$) a *passive* mapping typically exists in which the Hamiltonian serves as $-\beta$ [7]. This is illustrated in our discussion of the *passive* structure of robotic systems. However, there are some limitations with the study of *passive* systems. For example, systems which consist of cascades of *passive* systems (such as two integrators in series) are not necessarily a *passive* system.

Nicholas Kottenstette, Xenofon Koutsoukos, Joe Hall, Panos Antsaklis, Janos Sztipanovits, "Passivity-Based Design of Wireless Networked Control Systems for Robustness to Time-Varying Delays," 29th IEEE Real-Time Systems Symposium (RTTS 2008), Barcelona, Spain, November 30 - December 3, 2008.

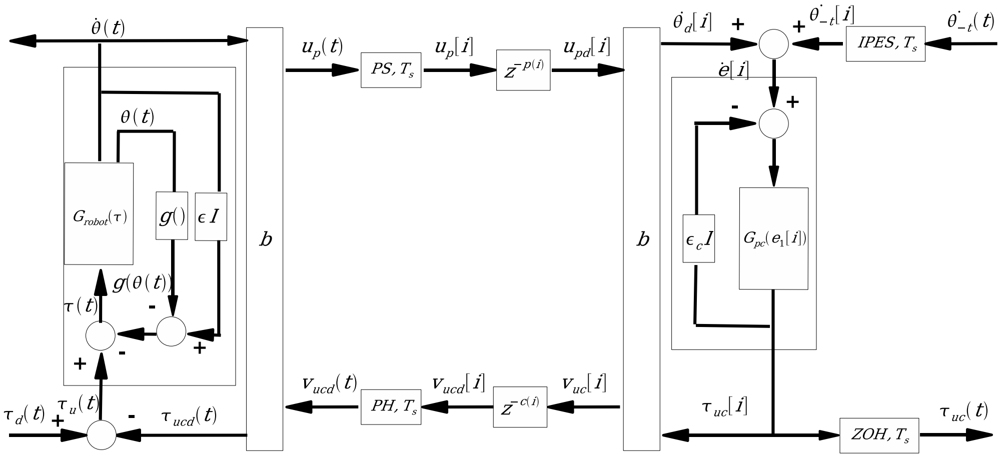

Figure 2. Proposed Wireless Control Scheme

3 Passive Control Architecture

3.1 Robotic System

Our control strategy takes advantage of the *passive* structure of a robotic system [20]. The robot dynamics which are denoted by $G_{\text{robot}}(\tau(t))$ in Figure 2 are described by

$$\tau = M(\Theta)\ddot\Theta + C(\Theta,\dot\Theta)\dot\Theta + g(\Theta). \tag{5}$$

The state variables Θ denote the robot joint angles, τ is the input torque vector, $M(\Theta)$ is the mass matrix, $C(\Theta,\dot\Theta)$ is the matrix of centrifugal and Coriolis effects, and $g(\Theta)$ is the gravity vector. The inertia matrix $M(\Theta) = M(\Theta)^{\mathsf T} > 0$ and the matrix C and $\dot M$ are related as follows:

$$-(\dot M - 2C) = (\dot M - 2C)^{\mathsf T} \implies x^{\mathsf T}(\dot M - 2C)x = 0,\ \forall x \in \mathbb{R}^n. \tag{6}$$

It is the skew-symmetry property given by (6) which makes it possible for the robot (with or without gravity compensation) to achieve a *passive* mapping. Despite the complexity of robotic manipulators, simple control laws can be used in a number of cases. A fundamental consequence of the passivity property is that a simple independent joint continuous-time proportional-derivative (PD) control can achieve global asymptotic stability for set-point tracking in the absence of gravity [14]. Therefore, we employ a PD controller but we consider a discrete-time equivalent implementation that communicates with the robotic system via a wireless network. To compensate gravity, we select as the control command $\tau_u = \tau - g(\Theta)$. Then the following supply rate

$$s(\tau_u(t),\dot\Theta(t)) = \dot\Theta^{\mathsf T}(t)\tau_u(t)$$

and corresponding storage function

$$V(\dot\Theta(t)) = \frac{1}{2}\dot\Theta^{\mathsf T}(t)M(\Theta(t))\dot\Theta(t)$$

can be used to show that the robot is a *passive dissipative* system which is also *lossless* in which all supplied energy is stored as kinetic energy in the robot [7]. Mathematically, this lossless property is characterized as follows:

$$\int_0^T \dot\Theta(t)^{\mathsf T}\tau_u(t)dt = \quad V(x(T)) - V(x(0)) \tag{7}$$

$$\int_0^T \dot\Theta(t)^{\mathsf T}\tau_u(t)dt \geq \quad -V(x(0)). \tag{8}$$

$V(x(0))$ represents all the *available storage* energy which can be extracted from the robot at time $t = 0$.

Furthermore, the robot can be made to be *strictly-output passive* by adding negative velocity feedback [10]. Therefore, we select the control command τ_u to have the following final form

$$\tau_u = \tau - g(\Theta) + \epsilon\dot\Theta,\ \epsilon \geq 0. \tag{9}$$

The gravity compensation and the velocity damping are implemented locally at the robotic system and it can be shown that the gravity compensated system with velocity damping denoted $G : \tau_u \mapsto \dot\Theta$ is *passive* when $\epsilon = 0$ and *strictly-output passive* for any $\epsilon > 0$ respectively. Therefore, the following conditions are satisfied:

$$\int_0^T \left[\dot\Theta(t)^{\mathsf T}\tau_u(t) - \epsilon\dot\Theta^{\mathsf T}(t)\dot\Theta(t)\right]dt \geq V(x(T)) - V(x(0)) \tag{10}$$

$$\int_0^T \dot\Theta(t)^{\mathsf T}\tau_u(t)dt \geq \epsilon\int_0^T \dot\Theta^{\mathsf T}(t)\dot\Theta(t)dt - V(x(0)). \tag{11}$$

Note that the velocity damped robot is a *strictly-output passive* system which is a L_2^m*-stable* system. It is the robots' *strictly-output passive* property which allows us to interconnect a *strictly-output passive* controller over a wireless network using wave variables such that the overall system remains *strictly-output passive* and L_2^m*-stable*. The proof for Theorem 2 requires these properties in order to show that digital control system depicted in Fig. 2 is L_2^m *stable*.

3.2 Wireless Control Architecture

Figure 2 depicts the proposed wireless control architecture. The robotic system $G : \tau_u \mapsto \dot{\Theta}$ is controlled by a a *passive* digital controller $G_{\text{pc}} : \dot{e}_1[i] \mapsto \tau_{uc}[i]$ using wave variables defined by the bilinear transformation denoted as b in Figure 2. The communication of the wave variables is subject to time-varying delays incurred in the wireless network that must be accounted for in order to ensure passivity and stability of the overall closed loop system.

The digital controller G_{pc} is interconnected to the robot via a *passive* sampler (PS) at sample rate T_s which converts the continuous *wave variable* $u_p(t)$ to an appropriate scaled discrete *wave variable* $u_p[i]$. Conversely, a *passive* hold device (PH) converts the discrete time *wave variable* $v_{ucd}[i]$ to an appropriately scaled *wave variable* $v_{ucd}(t)$ which is held for T_s seconds.

The *inner-product equivelant sampler* (*IPES*) and zero-order-hold (*ZOH*) blocks at the input of the digital controller are used to ensure that the overall system $G_{net} : [\dot{\Theta}_{-t}^{\mathsf{T}}(t), \tau_d^{\mathsf{T}}(t)]^{\mathsf{T}} \mapsto [\tau_{uc}^{\mathsf{T}}(t), \dot{\Theta}^{\mathsf{T}}(t)]^{\mathsf{T}}$ is (*strictly output*) *passive*. $\dot{\Theta}_{-t}(t)$ denotes a (negative) desired velocity profile for the robot to follow, $\tau_{uc}(t)$ is the continuous time *passive* control command, and $\tau_d(t)$ is a corresponding "disturbance" torque applied to the robots joints.

3.3 Wave Variables

The continuous robot input and output *wave variables* $v_{ucd}(t),\ u_p(t)\ \in \mathbb{R}^m$ depicted in Figure 2 are related to the corresponding torque and velocity vectors $\tau_{ucd}(t),\ \dot{\Theta}(t)\ \in \mathbb{R}^m$ as follows:

$$\frac{1}{2}(u_p^{\mathsf{T}}(t)u_p(t) - v_{ucd}^{\mathsf{T}}(t)v_{ucd}(t)) = \dot{\Theta}^{\mathsf{T}}(t)\tau_{ucd}(t). \quad (12)$$

The *wave variable* $v_{ucd}(t)$ and velocity measurement $\dot{\Theta}(t)$ are considered inputs and the *wave variable* $u_p(t)$ and delayed control torque $\tau_{ucd}(t)$ are considered outputs and are computed as follows:

$$\begin{bmatrix} u_p(t) \\ \tau_{ucd}(t) \end{bmatrix} = \begin{bmatrix} -I & \sqrt{2b}I \\ -\sqrt{2b}I & bI \end{bmatrix} \begin{bmatrix} v_{ucd}(t) \\ \dot{\Theta}(t) \end{bmatrix} \quad (13)$$

where $I \in \mathbb{R}^{m\times m}$ denotes the identity matrix.

The digital control input and output *wave variables* $u_{pd}[i],\ v_{uc}[i]\ \in \mathbb{R}^m$ depicted in Figure 2 are related to the corresponding discrete torque and velocity vectors $\tau_{uc}[i],\ \dot{\Theta}_d[i]\ \in \mathbb{R}^m$ as follows:

$$\frac{1}{2}(u_{pd}^{\mathsf{T}}[i]u_{pd}[i] - v_{uc}^{\mathsf{T}}[i]v_{uc}[i]) = \tau_{uc}[i]^{\mathsf{T}}\dot{\Theta}_d[i] \quad (14)$$

The *wave variable* $u_{pd}[i]$ and control torque $\tau_{uc}[i]$ are considered inputs and the *wave variable* $v_{uc}[i]$ and delayed velocity $\dot{\Theta}_d[i]$ are considered outputs and are computed as follows:

$$\begin{bmatrix} v_{uc}[i] \\ \dot{\Theta}_d[i] \end{bmatrix} = \begin{bmatrix} I & -\sqrt{\frac{2}{b}}I \\ \sqrt{\frac{2}{b}}I & -\frac{1}{b}I \end{bmatrix} \begin{bmatrix} u_{pd}[i] \\ \tau_{uc}[i] \end{bmatrix} \quad (15)$$

The received wave variables $u_{pd}[i],\ v_{ucd}[i]$ are delayed versions of the transmitted wave variables $u_p[i],\ v_{uc}[i]$ such that

$$u_{pd}[i] = u_p[i - p(i)]$$
$$v_{ucd}[i] = v_{uc}[i - c(i)].$$

3.4 *Passive* Sampler and *Passive* Hold

The *passive* sampler denoted (PS,T_s) in Figure 2 and the corresponding *passive* hold denoted (PH,T_s) must be designed such that the following inequality is satisfied $\forall N > 0$:

$$\int_0^{NT_s} (u_p^{\mathsf{T}}(t)u_p(t) - v_{ucd}^{\mathsf{T}}(t)v_{ucd}(t))dt - \sum_{i=0}^{N-1}(u_p^{\mathsf{T}}[i]u_p[i] - v_{ucd}^{\mathsf{T}}[i]v_{ucd}[i]) \geq 0. \quad (16)$$

This condition ensures that no energy is generated by the sample and hold devices, and thus, passivity is preserved.

Denote each j^{th} element of the column vectors $u_p(t), u_p[i]$ as $u_{p_j}(t), u_{p_j}[i]$ in which $j = \{1, \ldots, m\}$. An implementation of the PS that satisfies condition (16) is given by

$$u_{p_j}[i] = \sqrt{\int_{(i-1)T_s}^{iT_s} u_{p_j}^2(t)dt}\ \mathsf{sign}(\int_{(i-1)T_s}^{iT_s} u_{p_j}(t)dt). \quad (17)$$

in which $j = \{1, \ldots, m\}$.

Denote each j^{th} element of the column vectors $v_{ucd}(t), v_{ucd}[i]$ as $v_{ucd_j}(t), v_{ucd_j}[i]$ in which $j = \{1, \ldots, m\}$. An implementation of the PH that satisfies condition (16) is

$$v_{ucd_j}(t) = \frac{1}{\sqrt{T_s}} v_{ucd_j}[i-1],\ t \in [iT_s, (i+1)T_s]. \quad (18)$$

We note that the PS effectively scales the feedback velocity from the robot as follows:

$$\dot{\Theta}_d[i] \propto \sqrt{T_s}\dot{\Theta}((i-1)T_s - \tau((i-1)T_s)). \quad (19)$$

Furthermore, we note that the *passive* controller G_{pc} has infinite DC gain, and for a small $\epsilon_c > 0$ at steady state:

$$\dot{\Theta}[i] \approx -\dot{\Theta}_{-t}[i].$$

Therefore, $\dot{\Theta}_{-t}[i]$ can be related to a discrete time sampled robot velocity trajectory $\dot{\Theta}_t[i] = \dot{\Theta}_t(iT_s)$ as follows:

$$\dot{\Theta}_{-t}[i] = -\sqrt{T_s}\dot{\Theta}_t[i].$$

3.5 *Passive* controller

Typically a *passive* continuous-time PD controller is implemented as

$$\begin{aligned} \dot{e}_1(t) &= (\dot{\Theta}_d(t) + \dot{\Theta}_{-t}) \\ \tau_{uc}(t) &= K_p e_1(t) + K_d(\dot{\Theta}_d(t) + \dot{\Theta}_{-t}). \end{aligned}$$

A state-space realization of the controller can be described by

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (20)$$
$$y(t) = Cx(t) + Du(t). \quad (21)$$

where $A = 0,\ B = I,\ C = K_p = K_p^{\mathsf{T}} > 0,\ D = K_d = K_d^{\mathsf{T}} > 0\}$ (all matrices are in $\mathbb{R}^{m\times m}$).

To obtain a digital controller, we implement the discrete-time equivalent *passive* controller $G_{\text{pc}} : \dot{e}_1[i] \mapsto \tau_{uc}[i]$ computed from the state-space realization (20,21) with sampling period T_s. The resulting controller is implemented as

$$\begin{aligned} x[k+1] &= \mathbf{\Phi_o} x[k] + \mathbf{\Gamma_o} u[k] \\ y[k] &= \mathbf{K_s C_p} x[k] + \mathbf{K_s D_p} u[k]. \end{aligned} \quad (22)$$

where $\mathbf{K_s} > 0$ is a real diagonal scaling matrix and $u[k] = (\dot{\Theta}_d[k] + \dot{\Theta}_{-t}[k])$. Details for computing the digital controller and a theoretical result showing that the controller is *strictly-output passive* can be found in [9, Section 2.3.1].

4 Stability of the Networked Control System

This section presents the main analytical result that proves the stability of the networked control system. The proof can be found in [11, Appendix A].

Theorem 2 *For the wireless control architecture depicted in Fig. 2 consists of the passive robot described by (5) and (6) and the passive digital controller described by (22), if the communication protocol ensures that*

$$\int_0^{NT_s} \dot{\Theta}^{\mathsf{T}}(t)\tau_{ucd}(t)dt \geq \sum_{i=0}^{(N-1)} \tau_{uc}^{\mathsf{T}}[i]\dot{\Theta}_d[i] \quad (23)$$

always holds then when

$$\epsilon_c = \epsilon = 0$$

the system depicted in Figure 2 is passive. Furthermore, if

$$\epsilon_c > 0, \text{ and } \epsilon > 0$$

then the system is both strictly-output passive and L_2^m stable.

Condition (23) can be imposed on the wireless communication protocol by not processing duplicate transmissions of wave variables [10, Lemma 3-I]. The proof is fairly intuitive in noting that if the controller or plant processes duplicated transmitted wave variables the system will generate energy which is a non passive operation. Communication protocols such as TCP are appropriate because they provide an un-duplicated ordered stream of data where as the User Datagram Protocol UDP protocol is not appropriate (without checking for duplicated transmissions) since checking of duplicated datagrams is not required. Note that (23) *does not* require that the data needs to be ordered or for all the data to arrive as is guaranteed by the TCP protocol, so choosing to use the UDP protocol may be a better choice for transmitting data as long as the control application is able to drop duplicated datagrams.

Corollary 1 *For the wireless control architecture depicted in Fig. 2 in which the robot ($G_{robot}(\tau(t))$) is replaced by any passive system satisfying Definition 1-I (with gravity compensation disabled $g(\Theta(t)) = 0$) and the passive digital controller ($G_{pc}(\dot{e}_1[i])$) satisfies Definition 1-I, if the communication protocol ensures that*

$$\int_0^{NT_s} \dot{\Theta}^{\mathsf{T}}(t)\tau_{ucd}(t)dt \geq \sum_{i=0}^{(N-1)} \tau_{uc}^{\mathsf{T}}[i]\dot{\Theta}_d[i] \quad (24)$$

always holds then when

$$\epsilon_c = \epsilon = 0$$

the system depicted in Figure 2 is passive. Furthermore, if

$$\epsilon_c > 0, \text{ and } \epsilon > 0$$

then the system is both strictly-output passive and L_2^m stable.

5 Evaluation

5.1 Experimental Setup

We consider the Pioneer 3 (P3) arm which is a robotic manipulator built for the P3-DX and P3-AT ActivMedia mobile robots. The P3 Arm has two main segments, the manipulator and the gripper. The manipulator has five degrees of freedom and the gripper has an additional one. Figure 3 shows the home position of the P3 arm including the locations for the centers of gravity using the point mass assumption. The simulation model includes three main sub-

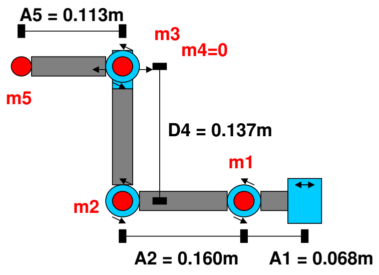

Figure 3. Pioneer 3 Arm

systems. The dynamic model of the robotic arm is described by (5) and simulated via a Simulink block from the "Robotics Toolbox for MATLAB" [4]. The simulated robot is provided gravity compensation and velocity damping as described in Section 3. To evaluate the performance of the *passive* digital control scheme over a wireless network, we use the "TrueTime Toolbox 1.5" [16]. We consider that the controller is interconnected to the robotic arm via an 802.llb wireless network. The network subsystem contains three nodes implemented as TrueTime kernel blocks. The first node (node 1) implements the network interface of the digital controller and the second node (node 2) the interface for the P3 arm. A third network node (node 3) is used as a disturbance node in order to incur time-varying packet delay as described in [3]. For our simulations, we use the 802.11b wireless block in TrueTime with the throughput is set to 11 Mbps, which is the theoretical limit of 802.11b, and the remaining parameters set to the default values. The controller wireless node and robot node are 10 meters apart while the disturbance node is 5 meters away from both. The packet size contains a 120 bit header plus preamble and a payload of 384 bits required to fit 6 double precision floating point values.

The controller subsystem contains two components: a block from the robotics toolbox (jtraj) which provides the reference velocity trajectory for the robotic arm to follow and a discrete state-space model of the controller. The controller receives as input the reference trajectory along with the actual robot velocity and computes the torque control command for the robot. To demonstrate the advantages of the passive control architecture, we performed two sets of experiments, one using a non-passive control architecture and one using the passive control scheme presented in Section 3. In all the experiments, the reference provided to the controller commands the robot to go to a position of [1 0.8 0.6 0.4 0.2 0] from the start position of all joints equal to zero.

5.2 Non-passive Control Architecture

In the first set of experiments, we consider a non-passive control scheme. To implement the digital controller, we discretize the continuous-time PD controller described by Eqs. (20)-(21) using a standard zero-order hold operation [1]. The digital controller communicates with the robot directly without using wave variables. The gravity compensation and velocity damping are implemented locally as in the passive control scheme.

Since we are using a zero-order hold operation to convert the continuous controller to a digital controller and are applying a zero-order hold to the input of the robotic plant we can take working control gains for the *passive* framework $k_{p-\text{passive}}$ and $k_{d-\text{passive}}$ and scale them using the following set of formulas:

$$\begin{aligned} \alpha &= 2T_s^2 \\ k_p &= \alpha k_{p-\text{passive}} \\ k_d &= \alpha k_{d-\text{passive}}. \end{aligned}$$

In spite of our best efforts to scale the gains, the non-passive system requires $\epsilon > .8$ in order to add enough damping to stabilize the nominal system. As ϵ is increased the system will begin to exhibit steady state error, so we chose to limit $\epsilon = 1$ for the non-passive system. Figure 4 compares the *passive* system ($k_{p-\text{passive}} = 321$, $k_{d-\text{passive}} = 82$, $\epsilon = .5$) to the non-passive system ($k_p = 1.6$, $k_d = .41$, $\epsilon = 1.0$). System responses are provided for both the nominal case and when subject to moderate time varying delays (disturbance= 0.5). Due to the added phase lag from the uncompensated zero-order hold, the overall non-passive system has little flexibility in adjusting its gains. Therefore, only the non-passive response for $T_s = 0.05$ seconds could be evaluated.

To simulate the system in the case of time-varying delays, we incorporate the disturbance node. The sampling period is kept constant (0.05 sec), but the amount of disturbance packets on the network varies. The disturbance node samples a uniformly distributed random variable $X[k] \in [0, 1]$ every 0.01 seconds. If $X[k] > d$ in which d is denoted as the disturbance parameter, a disturbance packet is sent out over the network. Figure 5 is a graph of the network delay between the controller node and the robot node caused by the disturbance traffic when $d = 0.5$. For comparison, Figure 6 depicts the corresponding network delay when $d = 1$ and $T_s = .05$ seconds. Figure 4 shows that the time-varying delays (when $d = 0.5$) have destabilized the robotic arm in the case of the non-passive control scheme.

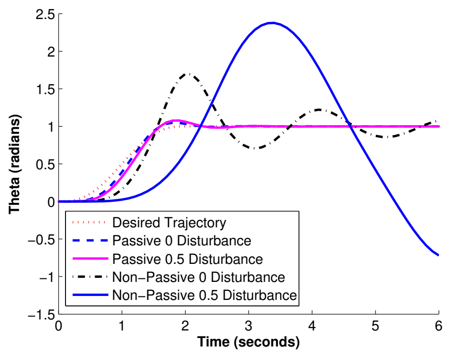

Figure 4. Comparing joint 1 response to *passive* and non-passive control.

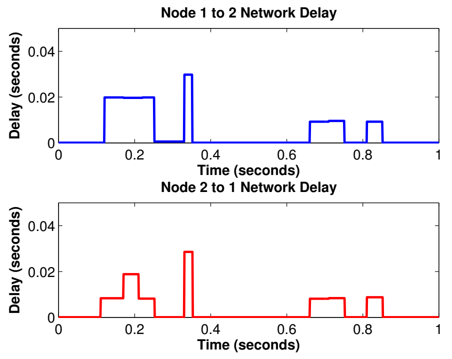

Figure 5. Network delay for $d = 0.5$.

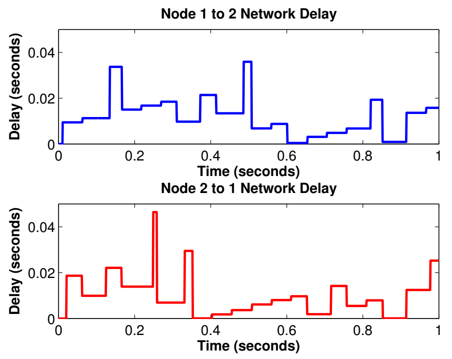

Figure 6. Network delay for $d = 1$.

5.3 Passive Control Architecture

The second set of experiments involves the proposed passive control architecture. In order to choose an appropriate set of continuous time gains k_p and k_d we focus our attention on joint 1 which is subject to the largest (changes of) inertia J as can be deduced from Figure 3.

$$G_{pm}(s) = \frac{1}{Js} \tag{25}$$

Similarly we approximate the controller to be of the form

$$G_c(s) = \frac{k_p + k_d s}{s}. \tag{26}$$

Next using basic loop shaping techniques we desire the system to have a crossover frequency (ω_c s.t. $20\log_{10}(|G_{pm}(j\omega_c)G_c(j\omega_c)|) = 0$ dB), in which $\omega_c = \frac{\omega_n}{N}$. $\omega_n = \frac{\pi}{T_s}$ is denoted as the Nyquist frequency. Therefore, the control gains can be computed based on a desired phase margin $0 < \phi \leq 90$ (degrees) as follows:

$$\tau = \frac{(\phi - 40)}{5\omega_c}$$
$$k_p = \frac{\omega_c^2}{J(\tau\omega_c + 1)}$$
$$k_d = k_p\tau.$$

Although the phase margin will never exceed 90 degrees, you can still calculate appropriate gains for k_p and k_d for $\phi > 90$ using the above straight line approximation. Due to the highly non-linear nature of our system (with $\epsilon = 1.0\text{e}{-6}$) we adjusted J to closely match the expected rise time given a 1 second trajectory since overshoot was still quite a significant component of the system response. All simulations given are for $\phi = 80$ degrees, $N = 2$, and $J = 2.93$ kg-m^2. Next, we chose an appropriate trajectory time (τ_t) which minimized overshoot and settling time. Finally, we evaluated the effectiveness of increasing ϵ while maintaining tracking. Since, ϵ and ϵ_c serve primarily to show that the overall system is L_2^m*-stable* we kept $\epsilon_c = 1.0\text{e}{-6}$ for all cases, the remaining system parameters are summarized in Table 1.

Figure 8 shows that as the sampling period is increased the overall system requires a larger trajectory time in order to minimize overshoot. Next, Figures (7, 10, 12) clearly show that by increasing $\epsilon = 0.5$ the *passive* system achieves faster settling times while exhibiting greater insensitivity to time varying delays when compared to Figures (9, 11) in which $\epsilon = 1.0e - 6$. This robustness to time-varying delays stems from the passivity constraints imposed on all the components of the networked control architecture and the damping effects of ϵ.

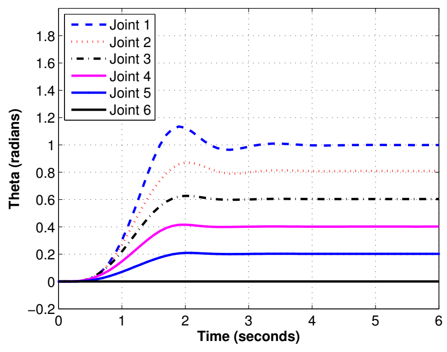

Figure 7. Joint 1 response ($T_s = .05$, $\epsilon = 0.5$, $d = 1.0$).

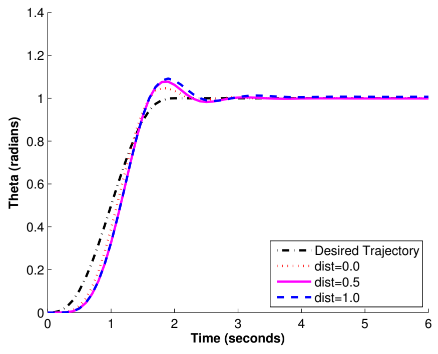

Figure 10. Joint 1 response ($T_s = .05$, $\epsilon = 0.5$, $d = \{0, 0.5, 1.0\}$).

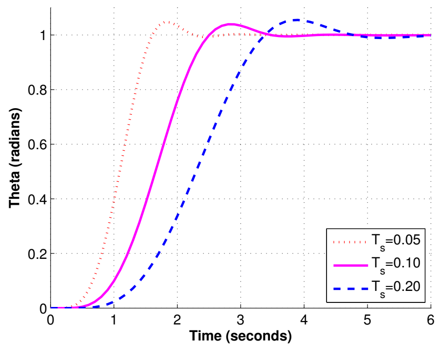

Figure 8. Joint 1 response ($T_s = \{.05, .10, .20\}$, $\epsilon = 0.5$, $d = 0.0$).

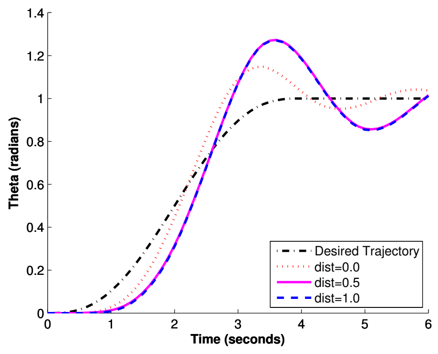

Figure 11. Joint 1 response ($T_s = .20$, $\epsilon = 1.0$e-6, $d = \{0, 0.5, 1.0\}$).

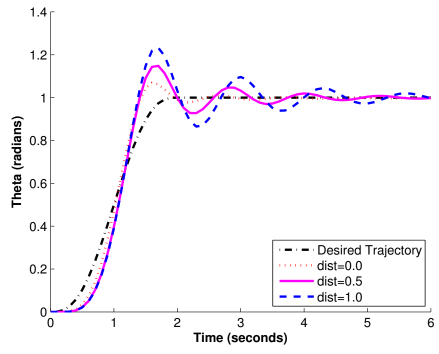

Figure 9. Joint 1 response ($T_s = .05$, $\epsilon = 1.0$e-6, $d = \{0, 0.5, 1.0\}$).

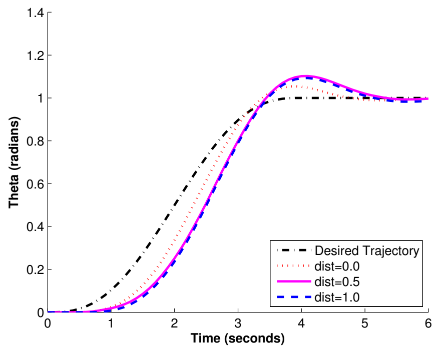

Figure 12. Joint 1 response ($T_s = .20$, $\epsilon = 0.5$, $d = \{0, 0.5, 1.0\}$).

Table 1. Passive Control Parameters Summary.

T_s	τ_t	ϵ	k_p	k_d	Figures
.05	2.0	1.0e−6	321.0	81.7	9
.05	2.0	0.5	321.0	81.7	4, 7, 10, 8
.10	3.0	0.5	80.2	40.9	8
.20	4.0	1.0e−6	20.1	20.4	11
.20	4.0	0.5	20.1	20.4	12, 8

6 Conclusions and Future Work

The paper presents a passive control architecture that offers advantages in building CPSs that are insensitive to network uncertainties, thus improving orthogonality across the controller design and implementation design layers and empowering model-driven development. We have presented an architecture for a system consisting of a robotic manipulator controlled by a digital controller over a wireless network in which stability of the networked control system is assured. Finally, we have evaluated the system using simulations results based on a detailed model that offers significant advantages especially in the presence of time-varying delays. Our future work focuses on three major directions: (i) theoretical methods that provide an effective way to interconnect multiple passive systems and controllers, (ii) an integrated end-to-end tool chain for the model-based design of CPSs based on passivity, (iii) and experimental studies to evaluate the proposed design methodology.

7 Acknowledgements

The authors gratefully acknowledge the support of the DoD (N00164-07-C-8510), NSF (NSF-CCF-0820088, NSF-CNS-0347440), and the Air Force (FA9550-06-1-0312, FA9550-06-1-0267).

References

[1] P. J. Antsaklis and A. Michel. *Linear Systems*. McGraw-Hill Higher Education, 1997.

[2] P. Berestesky, N. Chopra, and M. W. Spong. Discrete time passivity in bilateral teleoperation over the internet. In *IEEE International Conference on Robotics and Automation*, pages 4557 – 4564, 2004.

[3] A. Cervin, D. Hendriksson, B. Lincoln, and K. Arzen. Jitterbug and truetime analysis tools for real-time control systems. In *2nd Workshop on Real-Time Tools*, 2002.

[4] P. I. Corke. Robotic toolbox for matlab, release 7.1. Technical report, CSIRO, 2002.

[5] C. Desoer and M. Vidyasagar. *Feedback Systems: Input-Output Properties*. Academic Press, Inc., 1975.

[6] A. Fettweis. Wave digital filters: theory and practice. *Proceedings of the IEEE*, 74(2):270 – 327, 1986.

[7] W. M. Haddad and V. S. Chellaboina. *Nonlinear Dynamical Systems and Control: A Lyapunov-Based Approach*. Princeton University Press, 2008.

[8] H. Kopetz. The time-triggered architecture. *Proceedings of the IEEE*, 91(1):92 – 126, 2003.

[9] N. Kottenstette. *Control of Passive Plants With Memoryless Nonlinearitites Over Wireless Networks*. PhD thesis, University of Notre Dame, 2007.

[10] N. Kottenstette and P. J. Antsaklis. Stable digital control networks for continuous passive plants subject to delays and data dropouts. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 4433 – 4440, 2007.

[11] N. Kottenstette, X. Koutsoukos, J. Hall, J. Sztipanovits, and P. Antsaklis. Passivity-based design of wireless networked control systems subject to time-varying delays. Technical Report ISIS-08-904, ISIS Vanderbilt, 2008.

[12] X. Koutsoukos, N. Kottenstette, J. Hall, P. Antsaklis, and J. Sztinapovits. Passivity-based control design of cyber-physical systems. In *International Workshop on Cyber-Physical Systems - Challenges and Applications (CPS-CA'08)*, 2008.

[13] D. Lee and P. Li. Passive coordination control of nonlinear mechanical teleoperator. *IEEE Trans. Rob. Autom.*, 21(5):935–951, 2005.

[14] W. S. Levine. *Control System Applications*. CRC Press, 2000.

[15] P. Li and R. Horowitz. Control of smart machines, part 1: Problem formulation and non-adaptive control. *IEEE/ASME Trans. Mechatron.*, 2(4):237–247, 1997.

[16] A. C. M. Ohlin, D. Henriksson. TrueTime 1.5 - reference manual. Technical report, Department of Automatic Control, Lund University, 2007.

[17] I. T. MathWorks. Matlab. *The Language of Technical Computing, Version 7.6*, 2008.

[18] I. T. MathWorks. Simulink. *Dynamic System Simulation for MATLAB, Version 7.1*, 2008.

[19] G. Niemeyer and J.-J. E. Slotine. Towards force-reflecting teleoperation over the internet. In *IEEE International Conference on Robotics and Automation*, pages 1909 – 1915, 1998.

[20] R. Ortega and M. Spong. Adaptive motion control of rigid robots: A tutorial. In *27th IEEE Conference on Decision and Control*, pages 1575 – 84, 1988.

[21] A. L. Sangiovanni-Vincentelli. Quo vadis sld: Reasoning about trends and challenges of system-level design. *Proceedings of the IEEE*, 95(3):467 – 506, 2007.

[22] C. Secchi, S. Stramigioli, and C. Fantuzzi. Digital passive geometric telemanipulation. In *IEEE International Conference on Robotics and Automation*, pages 3290 – 3295, 2003.

[23] A. van der Schaft. *L2-Gain and Passivity in Nonlinear Control*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999.

[24] G. Zames. On the input-output stability of time-varying nonlinear feedback systems. i. conditions derived using concepts of loop gain, conicity and positivity. *IEEE Transactions on Automatic Control*, AC-11(2):228 – 238, 1966.