



PaNeCS: A Modeling Language for Passivity-based Design of Networked Control Systems

**Emeka Eyisi, Joe Porter, Joe Hall, Nicholas Kottenstette,
Xenofon Koutsoukos, Janos Sztipanovits**

ISIS/Vanderbilt University



Motivation



- Increase in the use of distributed architecture in the design of real-world systems.
 - Need for networked control systems(NCS).
 - Networked Control Systems
 - Interaction of computational dynamics, physical dynamics and communication networks.
 - Compositionality
 - Separation of Concerns
-



Challenges



- Compositionality and stability verification
 - Network uncertainties such as time-varying delays and packet loss cause significant challenges in NCS
 - Expensive preliminary system prototyping
 - Extensive revision due to design changes
 - **Key idea: An automated model-based approach based on passivity theory.**
-



Outline



- Passivity based Control of Networked Control Systems
 - PaNeCS
 - Passivity Analysis
 - Code Generation and Implementation
 - Case Study
 - Conclusion
-



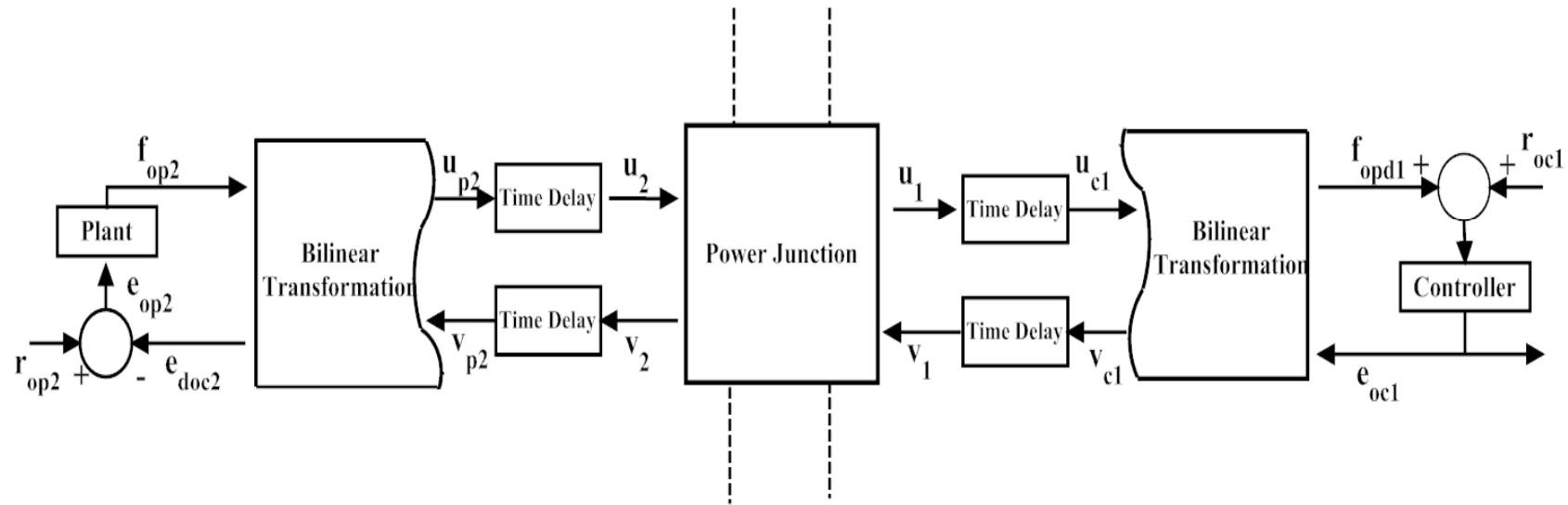
Passivity-based Control of Networked Control Systems



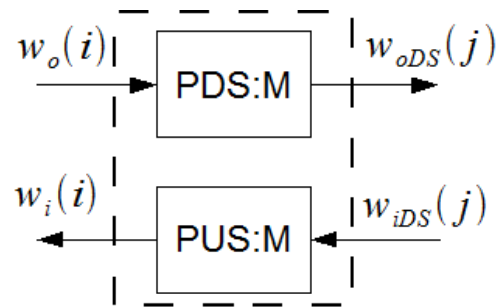
- **Passive Systems**
 - A passive system only stores and dissipates energy but cannot generate energy of its own
 - **Unique Properties of Passive Systems**
 - Passive Systems connected in either a parallel or negative feedback results in an overall passive system.
 - **Benefits**
 - Separation of concerns allows model-based design approach to be extended to networked control systems.
-



Passivity-based Architecture



A networked control system



Passive Upsampler and DownSampler Pair



Outline



- Passivity based Control of Networked Control Systems
 - PaNeCS
 - Passivity Analysis
 - Code Generation and Implementation
 - Case Study
 - Conclusion
-



PaNeCS



- Developed using the meta-configurable Model Integrated Computing (MIC) tool, Generic Modeling Environment (GME).
 - PaNeCS Meta-model
 - **Main Components**
 - Plant Subsystem
 - Controller Subsystem
 - PowerJunction Subsystem
 - Network Subsystem
-



Plant Subsystem



■ Components

■ Plant

- Discrete-Time LTI

■ BilinearTransformP

■ PassiveDownSampler

■ PassiveUpSampler

$$x(k+1) = Ax(k) + Bu(k)$$

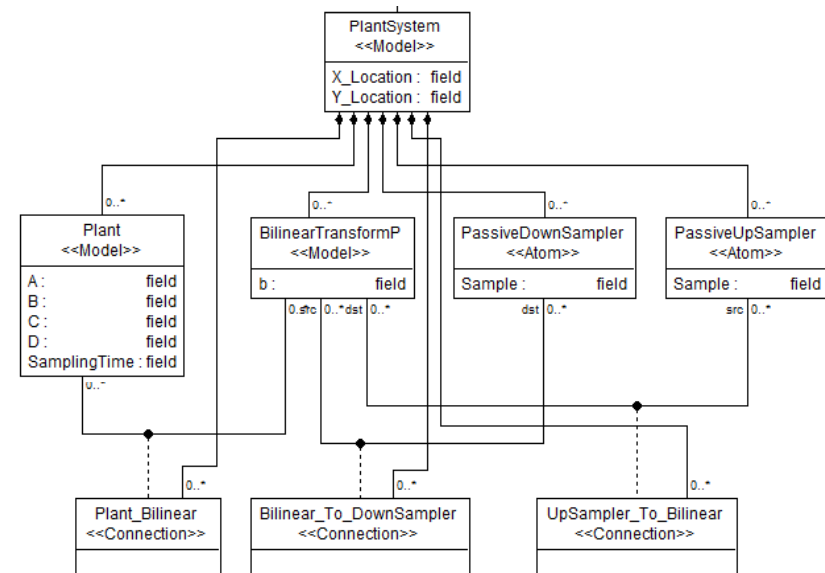
$$y(k) = Cx(k) + Du(k)$$

■ Interconnections

■ Plant_Bilinear

■ Bilinear_To_DownSampler

■ UpSampler_To_Bilinear

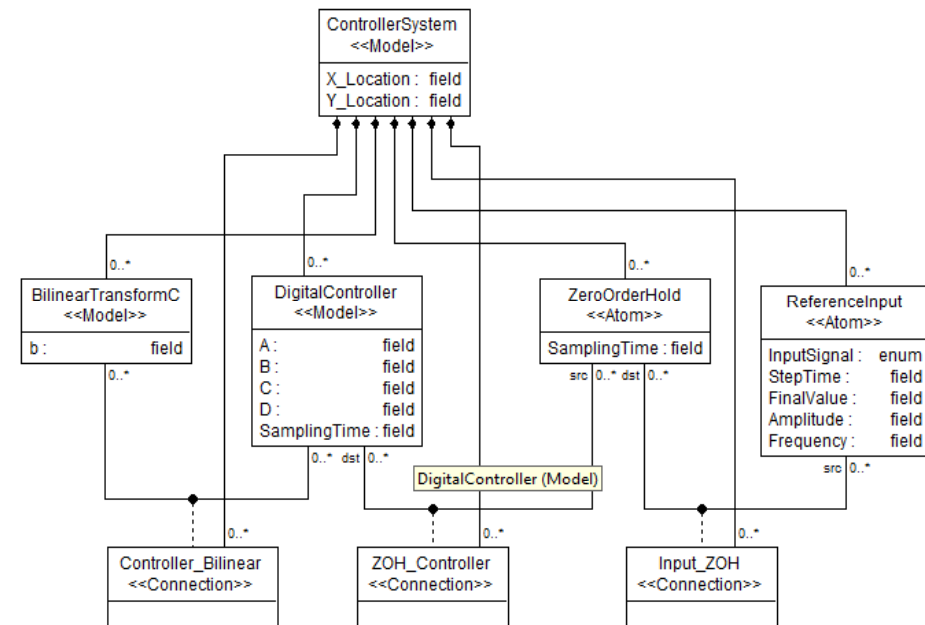




Controller Subsystem



- **Components**
 - DigitalController
 - BilinearTransformC
 - Reference Input
 - ZeroOrderHold
- **Interconnections**
 - Controller_Bilinear
 - ZOH_Controller
 - Input_ZOH



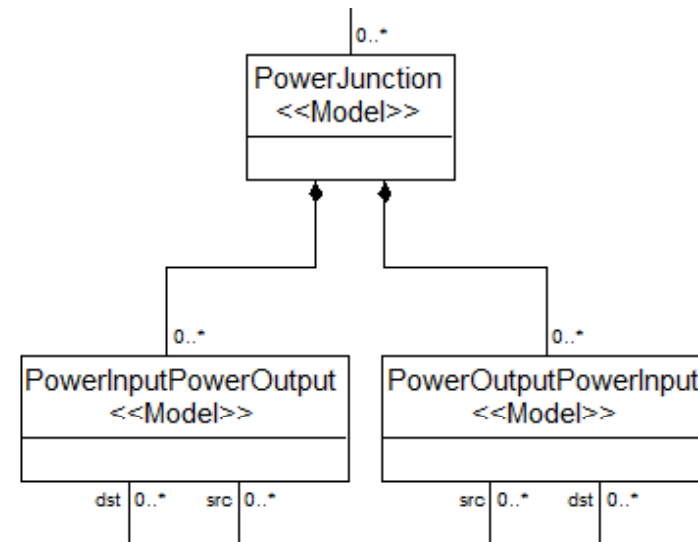


PowerJunction Subsystem



- **Components**
 - PowerInputPowerOutput
 - PowerOutputPowerInput

$$\sum_{k=m+1}^n (u_k^T u_k - v_k^T v_k) \geq \sum_{j=1}^m (u_j^T u_j - v_j^T v_j)$$

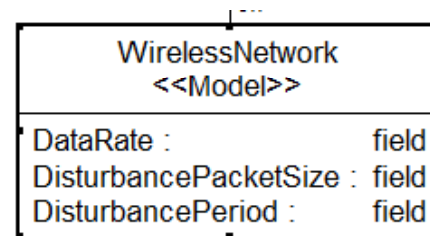




Network Subsystem



- **Network representation**
 - Defines parameters for the network
 - Ability to introduce network disturbance for simulation purposes

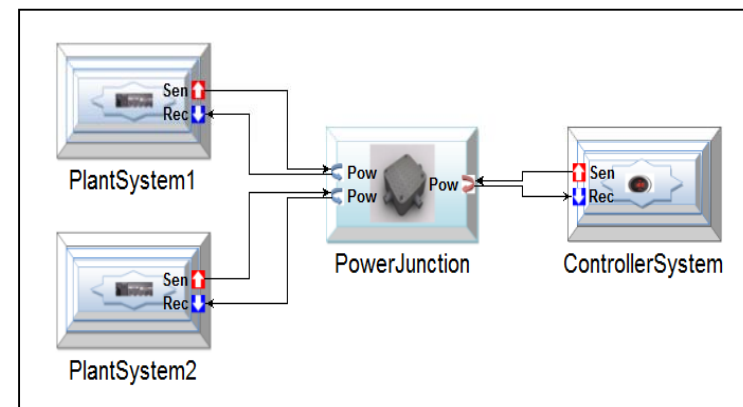




Control Design Aspect



- Provides visualization of the control modeling layer indicating flow of control and sensor signals.
- Components represent control design concepts.
- Visible Components
 - Plant Subsystem
 - Controller Subsystem
 - Powerjunction



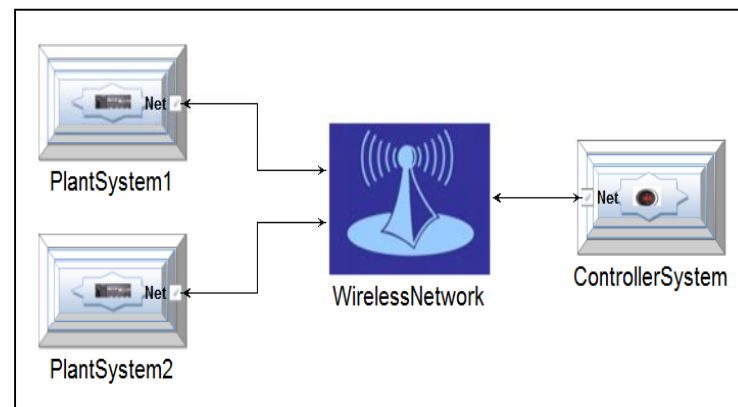
Control Design Aspect



Platform Aspect



- Provides visualization of the physical platform layer indicating the flow of data packets over the network.
- Components represent physical entities
- Visible components
 - Plant Subsystem
 - Controller subsystem
 - Wireless network



Platform Design Aspect



Structural Semantics



- Defined using
 - Meta-model notations
 - Object Constrained Language (OCL)
 - GME is embedded with an OCL engine
 - Used to define constraints that are enforced at design time.
 - Implemented Constraints
 - Cardinality Constraints
 - Connection Constraints
 - Unique Name Constraints
-



Sample OCL Implementation



- OCL Implementation
 - Connection between BilinearTransformC and DigitalController

Description: There must be one connection between the DigitalController block and the BilinearTransformC block

Equation: `self.connectionParts("Controller_Bilinear").size()=1`



Outline



- Passivity based Control of Networked Control Systems
 - PaNeCS
 - Passivity Analysis
 - Code Generation and Implementation
 - Case Study
 - Simulation
 - Conclusion
-



Passivity Analysis



- In order to achieve the desirable properties of passive systems
 - Analyze the networked control system
 - Analysis of the NCS
 - Component Analysis
 - System-level Analysis
-



Component Analysis



- Analyze individual components of the NCS
 - Only Plant and Controller Components
- Designed Model Interpreter Tool integrated in GME visits each tool and invokes the analysis function.

$$\begin{bmatrix} A^T P A - P - \tilde{Q} & A^T P B - \tilde{S} \\ (A^T P B - S)^T & -\tilde{R} + B^T P B \end{bmatrix} \preceq 0$$

$$\tilde{Q} = C^T Q C, \quad \tilde{S} = C^T S + C^T Q D$$

$$\tilde{R} = D^T Q D + (D^T S + S^T D) + R$$

$$\exists \varepsilon > 0, \quad Q = -\varepsilon I, \quad R = 0, \quad S = \frac{1}{2} I$$

- CVX semi-definite programming tool (SDP) used in a Matlab script to solve LMI.
-



System-level Analysis



- Due to the “correct-by-construction” approach
 - Network as a whole ensure global robustness by a combination of
 - Individual components satisfaction of passivity constraints.
 - Passive Composition constraints encoded in the modeling language.
 - Reduction in the analysis burden of verifying passivity.
-



Outline



- Passivity based Control of Networked Control Systems
 - PaNeCS
 - Passivity Analysis
 - Code Generation and Implementation
 - Case Study
 - Simulation
 - Conclusion
-



Code Generation and Implementation



- Network Control Systems modeled in PaNeCS is implemented in Matlab/Simulink
 - Extended with TrueTime (a Simulink Toolbox from Lund University) to simulate real network behavior.
 - Model Interpreter
 - Developed in C++ using Builder Object Network (BON2) API provided with GME
 - Traverses the instance model in PaNeCS
 - Generates Matlab code
 - Builds Simulink and Truetime models
-



Network Dynamics



- Two blocks from the TrueTime Library are utilized to implement the network dynamics
 - TrueTime Kernel
 - TrueTime Wireless Network





TrueTime Kernel



- Represents each subsystem as a node in a network.
 - Responsible for I/O operations and network data acquisition as well as implementing other user-defined tasks.
 - Two main scripts
 - **Initialization Script:** Define number of inputs and outputs, the function code name and the kernel's node id.
 - **Function script:** Implements user defined tasks such as sending and receiving data over a network and other computational tasks .
-



TrueTime Wireless Network



- Simulates network dynamics
 - Implementing the transfer of data packets over a wireless network from one node to another.
 - Parameters such as path loss functions allow for simulating various network dynamics.
-



Implementation



- Each Plant Subsystem and Controller Subsystem is implemented as a Simulink Subsystem.
 - Each of the Plant Subsystem and Controller Subsystem are connected to a Truetime Kernel block.
 - Tasks in each TrueTime kernel connected to a Plant Subsystem is performed periodically based on the user specified sampling time.
 - Power junction is implemented as a task in TrueTime kernel connected to the Controller Subsystem.
-



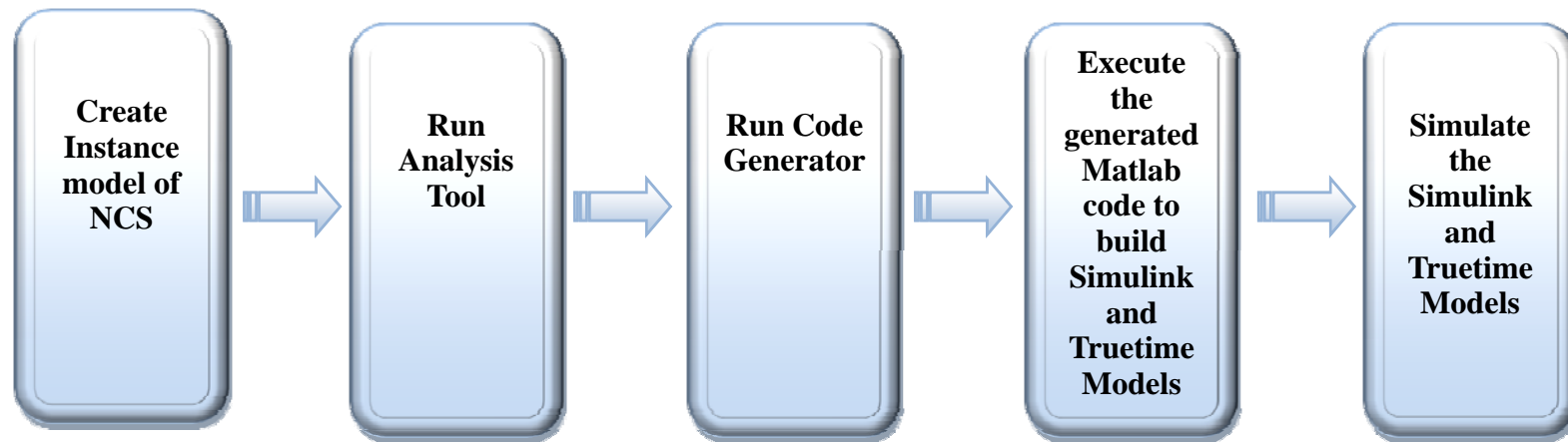
Execution Step



- Wave variables from PlantSystem are sent to the PowerJunction
 - Powerjunction computes the average power of the wave variables and sends it to the ControllerSystem
 - The controller receives the wave variables and computes the control signal which is sent back to the Powerjunction and then sent back to the individual Plant Subsystems.
-



PaNeCS Design Flow



NCS design workflow using PaNeCS



Outline



- Passivity based Control of Networked Control Systems
 - PaNeCS
 - Passivity Analysis
 - Code Generation and Implementation
 - Case Study
 - Conclusion
-



DEMO



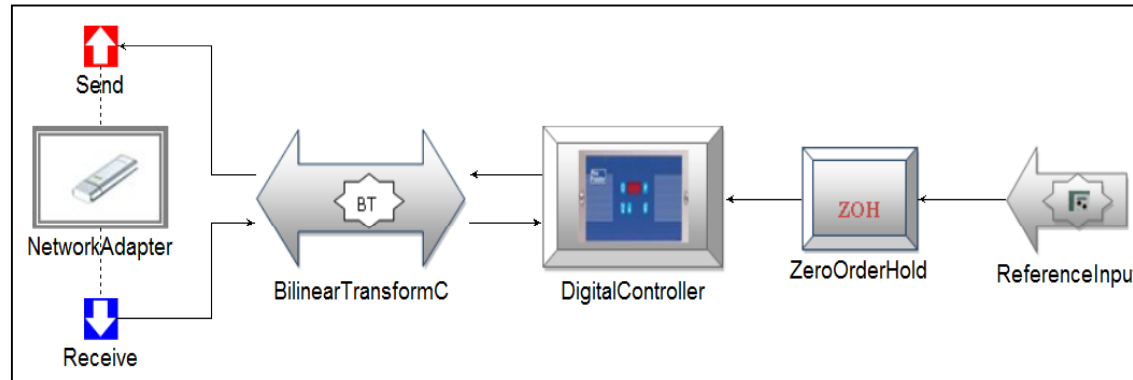
Case Study



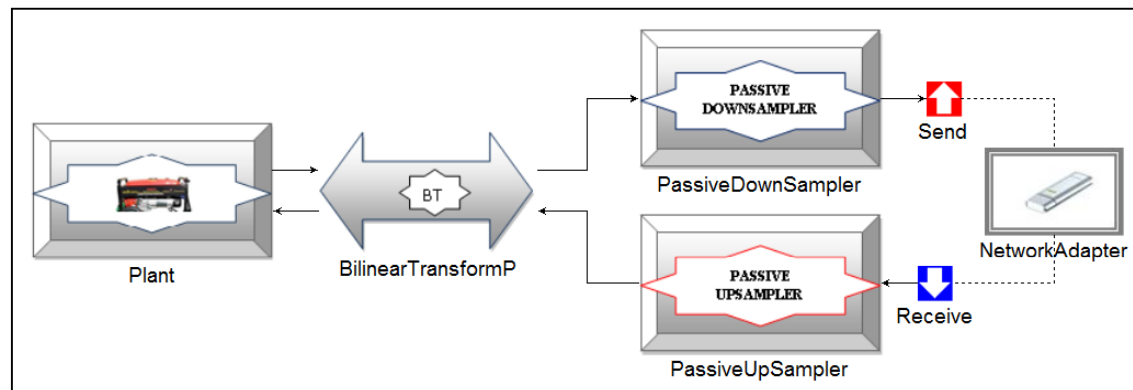
- Control of two distinct plants to track a specified trajectory over a wireless network using the power junction.
 - Discrete Plants with inertial mass 2kg and 0.25kg respectively. Each with a sampling time of 0.01s.
 - Proportional Controller gain, 10π
 - Sinusoidal Reference signal
 - Amplitude, 0.025
 - Frequency, $2\pi/30$
-



Component Details



Control Subsystem



Plant Subsystem



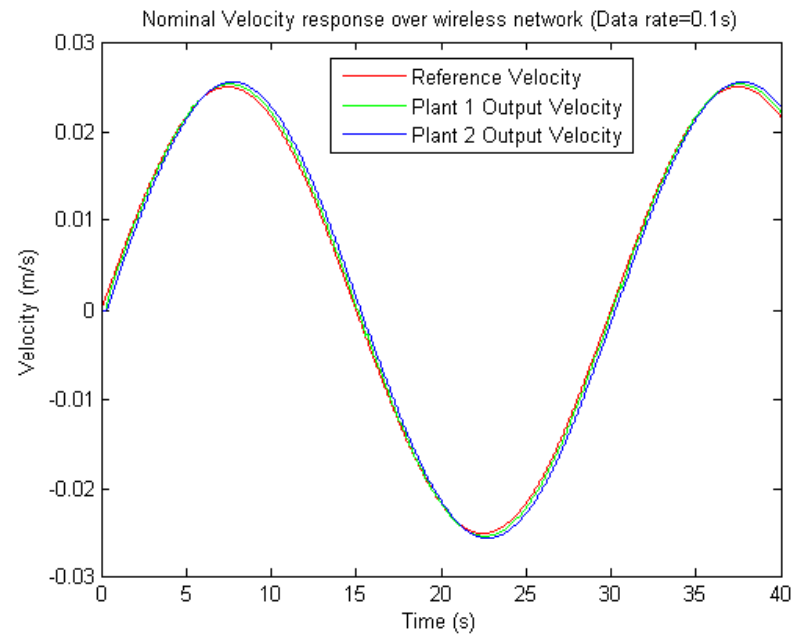
Experiments



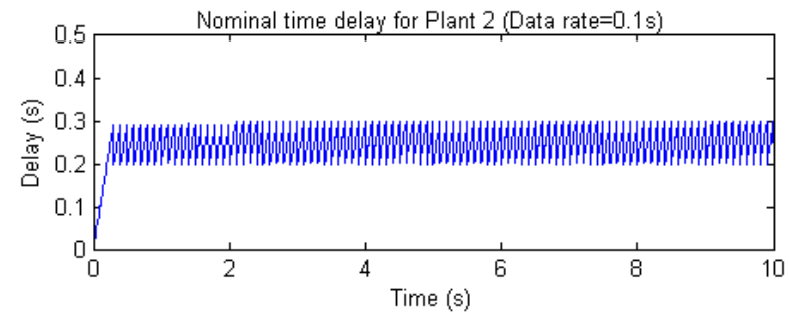
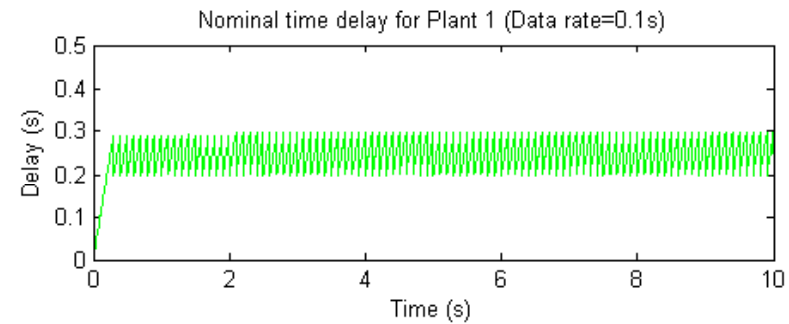
- Two experiments
 - Nominal Condition
 - Network Disturbance
 - Introduction of disturbance in the network to illustrate robustness
-



Nominal Experiment



Velocity Plots

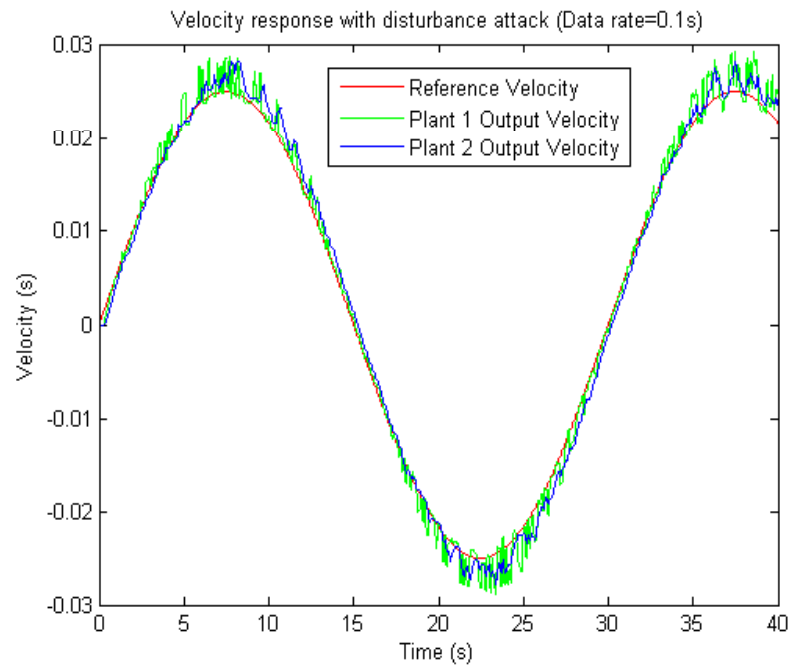


Delay Plots



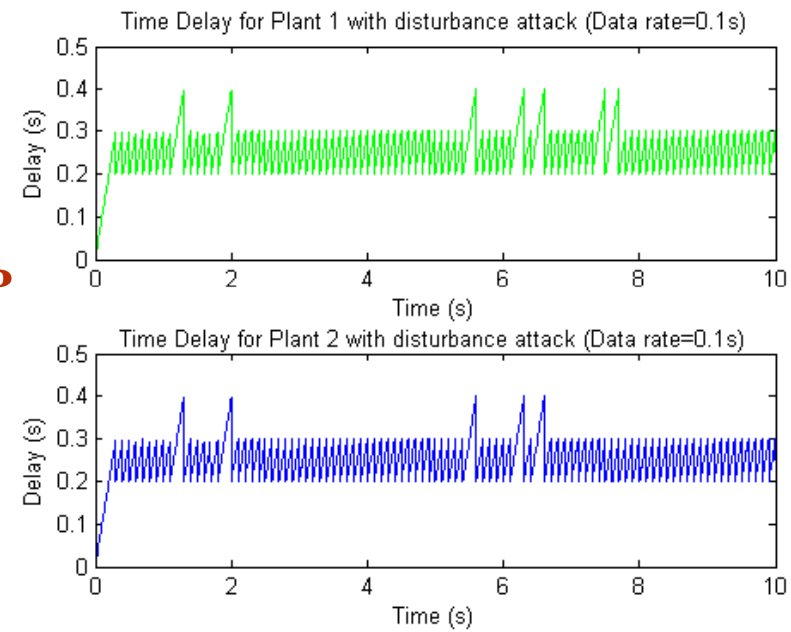


Experiment with Network Disturbance



Velocity Plots

P



Delay Plots



Outline



- Challenges
 - Passivity based Control of Networked Control Systems
 - PaNeCS
 - Code Generation and Implementation
 - Case Study
 - Conclusion
-



Conclusion



- Developed a Domain Specific Modeling language for modeling robust Networked Control Systems using passivity theory.
 - Developed Integrated analysis tool for enforcing component satisfaction of passivity constraints
 - Developed a code generator for simulating models of networked control systems using Matlab/Simulink/Truetime.
-



Questions??
