Collaborative Project
Small-medium-scale focused research project (STREP)

Grant Agreement n. 224168 FP7-ICT-2007-2

WIDE
Decentralized and Wireless Control of Large-Scale Systems

Starting date: 01 September 2008 Duration: 3 years

Deliverable number	**D4.3**
Title	**Collection of prototype design tools in MATLAB for control and estimation with WSN in the loop**
Work package	WP4 – Network-aware control and estimation (RTD)
Due date	M36
Actual submission date	01/09/2011
Lead contractor for this deliverable	UNITN
Author(s)	Alberto Bemporad `<alberto.bemporad@imtlucca.it>`
With the help of	Pavel Trnka `<Pavel.Trnka@Honeywell.com>`
Nature	Report
Revision	v1.1 (February 3, 2012)

Dissemination level

→**PU**	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
CO	Confidential, only for members of the consortium (including the Commission Services)

Executive summary
This report describes the part of the toolbox related to the design and simulation of network-aware Kalman filters. We refer the reader to the WIDE Toolbox Manual for the remaining tools for control with WSN in the loop, in particular to the “NCS Toolbox” manual, and to the “Energy Aware MPC”, “Esenza acquisition”, and “Decentralized LMI” documents included in the public toolbox distribution.

Contents

Tools for Design and Simulation of Network-aware Kalman filters

Introduction

The objective of this section is to describe the use of a collection of Simulink blocks implementing functionalities of network-aware Kalman filters. These filters, in their present implementation, are intended for the integration in Distributed Control Systems, DCS, as a support for PID loops to handle communication delays. Hence, the presented blocks assume single-input, single-output process models. The filters receive network-delayed process output data together with the time stamp marking the time, when the measurement was taken. This is assumed to be done by the network transmitter at the sensor side. Kalman filter provides a filtered, undelayed process output as is shown in Figure 1. In this configuration it is assumed that the controller and the actuator are wire-connected, i.e., there is no delay in the manipulation variable channel. This asymmetry in plant-controller communication between the sensor and actuator channels is justifiable: the actuators in process control cannot be of miniature dimension as the sensors, and need an external power source; for that reason, they are likely to be accessible to the 'wired' control infrastructure. In the case, when there actually is a network connection between the controller and the actuator, Kalman filter does not have correct data on the actuation that was applied to the plant, which may result in incorrect prediction. For that reason, the actuator sends back an acknowledgement signal containing the information on the delay of the actuator signal. This acknowledgement signal is subject to network delays as well. After its reception, Kalman filter can adjust its internal state. This configuration is shown in Figure 2.

The implemented algorithms are based on HPL results obtained on their work on WIDE project, workpackage WP4. They include the following algorithms:

- Novel efficient and robust implementation of the *optimal, time-varying Kalman filter* for the process augmented by the network delay model. This algorithm was presented in [3].
- Novel *quasi-steady state optimal Kalman* filter assuming that the communication delay is upper-bounded. Under this condition, and for time invariant plant model, the optimal filter converges to the quasi-steady state in which the state covariance attains a *finite number* of values determined by the combination of missing data. Then, Kalman filter is a linear switched system whose state injection gain is fully determined by the set of missing data and the current sample delay. The finite set of injection gains is pre-computed off-line. Hence, the on-line computational load is small (comparable to the steady state

Kalman filter); the number of pre-computed injection gains grows combinatorially with the maximum delay. In the case when the assumption on maximum communication delay is not satisfied, this estimator is suboptimal. This result is described in [2].

- Novel suboptimal Kalman filter assuming a maximum communication delay and a limited number of temporarily missing data. When neither the maximum delay nor maximum number of missing samples is exceeded, then the filter regains optimality when the missing data buffer becomes empty. This filter is also a switched linear system with a finite set of pre-computed injection gains. This set is much smaller than that of the optimal Kalman filter. This filter was introduced in [4]. This algorithm is implemented in the same block as the optimal quasi-steady-state one, and the level of sub-optimality is set by two integer parameters.
- Kalman filter with state corrections based on received MV reception acknowledgement signal. These corrections can be applied to any Kalman filter; however, they were implemented only with the sub-optimal filter described above.

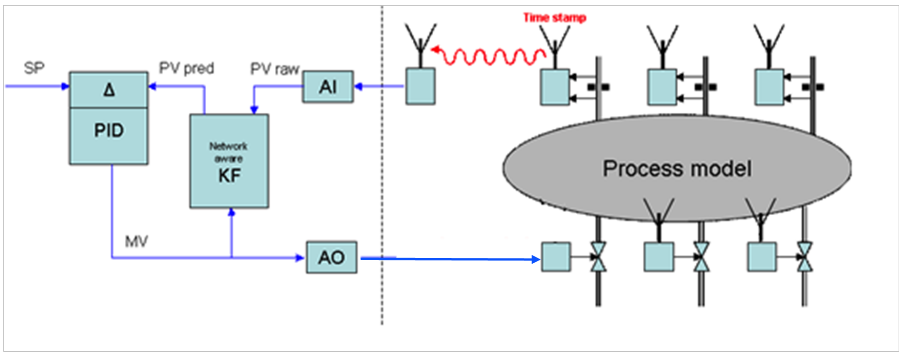


Figure 1 Configuration of control loop with network in the CV channel.

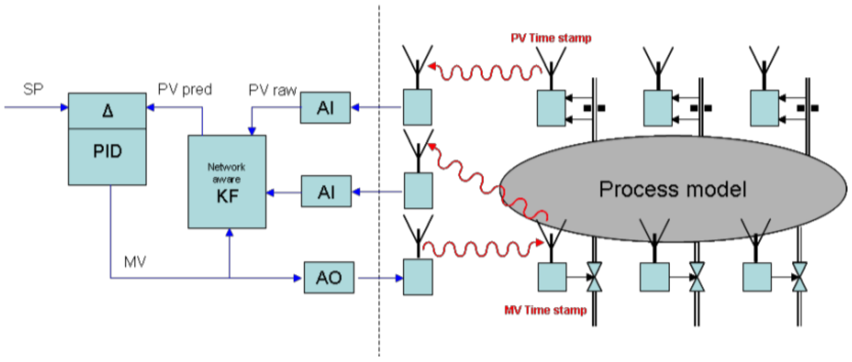


Figure 2 Configuration of control loop with network in both CV and MV channels with re-sending information on actual actuation applied.

These filters were described, besides the above mentioned publications, also in WIDE Deliverable report D4.2. Moreover, performance evaluations and closed loop simulations using the tools described in this report were shown in the DEMO evaluation report D5.x.

The remainder of this report is contains the following:

1. Detailed description of Simulink functional blocks implementing net-aware Kalman filter functionalities.
2. Description of Simulink functional blocks used for supporting Kalman filter blocks, as, e.g., a multi-hop network model, a bursting disturbance model, a simple WSN receiver holding the latest valid value.
3. Description of a Simulink demo model including a plant model, a PID controller and a range of net-aware Kalman filters for modeling the loop/network configuration as in Figure 1. The filters are connected to the plant and the controller. The feedback can be switched to any of those filters. For comparison, a wired feedback, a Smith predictor, and some network un-aware options are available.
4. Description of a Simulink demo model for modeling the loop/network configuration as in Figure 2.

The above mentioned simulation blocks were implemented as Level 2 m-file S-functions.

These tools were developed and tested in MATLAB® version 7.9.0 (R2009b) and Simulink® version 7.4 (R2009b). The simulation blocks as well as the demo models use the Control System Toolbox of Matlab.

Model blocks of net-aware Kalman filters

Overview

The overview of model blocks implementing network-aware Kalman filters is in Table 1 below. In particular, three blocks have been developed implementing functionality outlined in the previous sub-section. S-function `SKalmanFilter_Yswitch` implements both the quasi-steady-state optimal filter as well as the switched sub-optimal one, depending on the block parameters.

These blocks have two input ports for receiving data from the network in common: The first port is for values of sampled data transmitted over the network and the second one for data time stamps. The network is assumed to work on sampling rates that are integer multiples of those of the controller and the filter. The timing is shown in Figure 3. Sensor samples the process variable with controller sampling period T_s. The transmitter assigns a time stamp to a valid sample. The network transmissions operate on sampling period $T_{net} = T_s / m$. The receiver at the filter side receives delayed process data and time stamps in the T_{net} time slots. *If no valid data is received in a time slot, the network is assumed to assign a negative value to the time stamp* and the receiver holds the last valid data value, as shown in Figure 3. Net-aware Kalman filters are thus double rate discrete-time systems: it contains a part operating on the fast sampling rate $1/T_{net}$ that aggregates valid samples received in the last control sampling interval $\left[(k-1)\cdot T_s, k\cdot T_s\right]$ and supplies their delays relative to the baseline discrete time $k\cdot T_s$ to the part updating the filter state estimate and operating on the control sampling interval. As the sensor is assumed to be synchronized with the control system, these delays are integer numbers. The actual state estimation is then done at the controller sampling rate $1/T_s$. The same sampling rate thus applies to the input port for the manipulated variable as well as to the output port providing the filtered, undelayed process value. Moreover, Kalman filter marked as No 3 in Table 1 has a pair of ports to receive the actuator acknowledgement signal. *All these ports are one-dimensional.*

Table 1 Model blocks of net-aware Kalman filters

No	*S-function name*	*Functionality*
1	`SKalmanFilter_Yswitch`	Switched quasi-steady-state optimal/sub-optimal network-aware Kalman filter
2	`SKalmanFilter_YOpt`	Optimal time-varying network-aware Kalman filter, new implementation
3	`SKalmanFilter_UY`	Switched (sub-)optimal network-aware Kalman filter, using acknowledgement signal from the actuator to refine state estimates

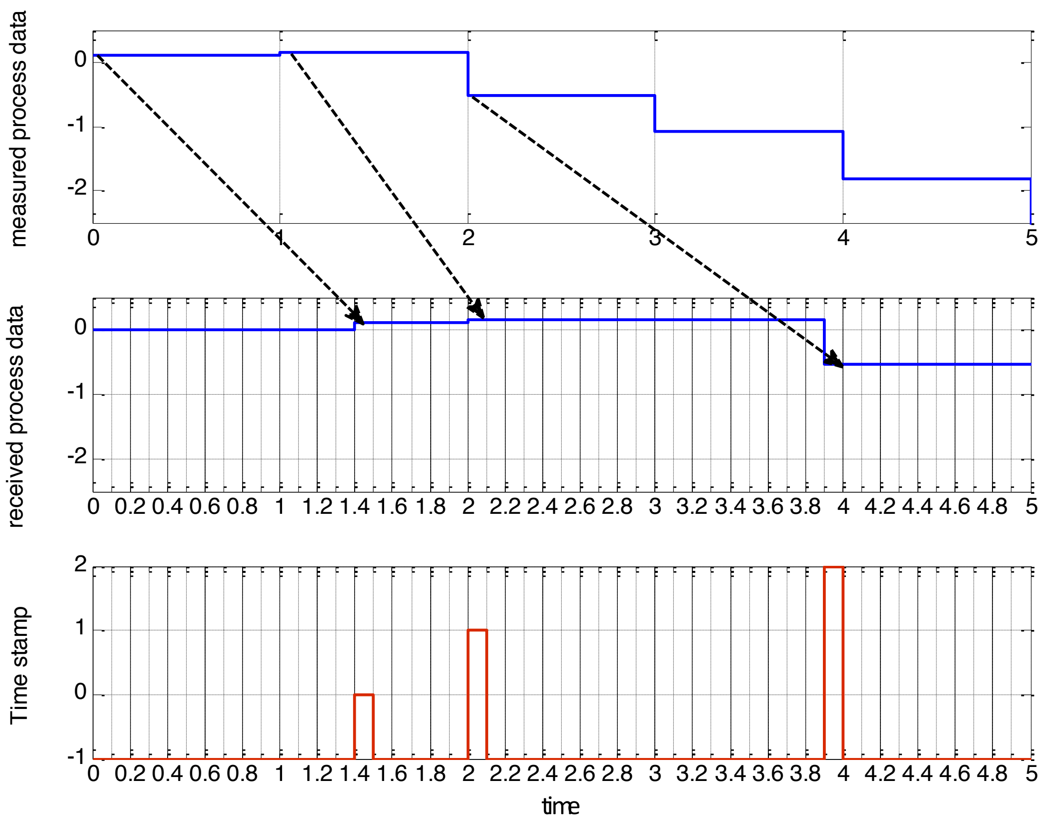


Figure 3 Timing of network output data

Kalman filter block use a standard linear-Gaussian state-space representation of the process model, i.e.,

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}u(k) + \xi(k)\\ y(k) &= \mathbf{C}\mathbf{x}(k) + \eta(k)\end{aligned}$$

There, $\mathbf{x}$ is state of the process of an arbitrary dimension, u is a manipulated variable and y is the controlled process value. Matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are passed to the block as S-function parameters. Variables ξ and η are process and measurement noises, respectively. They are assumed to be zero-mean, Gaussian and white; they are further assumed to be mutually independent and their covariances are $\mathbf{Q}$ and R, respectively; the former is a symmetric, positive-definite matrix of the same size as $\mathbf{A}$, the latter is a positive scalar. Both $\mathbf{Q}$ and R are parameters of all above mentioned simulation blocks of net-aware Kalman filters. Kalman filters are state observers that provide an internal state estimate. This particular implementation, however, provides the *filtered process output only, which is assumed to be fed-back to the PID controller.*

Finally, these filters need to know *a priori* chosen maximum communication delay $n_{d\max}$. All samples received with a valid time stamp older than $\left(k - n_{d\max}\right)T_s$ are discarded. More on the block parameters is discussed in the following paragraphs describing the particular blocks.

Initial states are a-priori assumed to be zero; moreover, an assumption is that no measurement sample is ‘pending’ at the initial state.

Block `SKalmanFilter_Yswitch`

This block implements the optimal quasi-steady state Kalman filters, that use a switched Kalman gain. The set of Kalman gains is pre-computed in the initialization phase, which may take some time. The level of sub-optimality depends on parameters that fall into the network model group. In particular, it is assumed that every measurement reaches the filter with delay not larger than `nd`. Variable `ndopt` defines the maximum delay for which the samples are handled in an optimal way. Hence, the parameter combination `nd = ndopt, nmis=0` thus gives the optimal, quasi-steady-state solution for the maximum delay nd. For `nd >= ndopt+nmis,` we obtain a sub-optimal solution. If a sample is not received within delay `ndopt`, the process data sample is replaced by its prediction and pretending that it was true process data; this is, of course, not an optimal step. The data stamp of this missed and incorrectly fused sample is recorded in the missing data buffer, whose length is limited by `nmis`. If the data sample is received later (but with delay no longer than `nd`), it is optimally fused; if the missing data buffer is empty, the state of the optimal quasi-steady state filter is restored. If the missing sample buffer is full and an additional sample is missing, the oldest time-stamp is removed from the buffer and the latest one is recorded. If the sample whose time stamp was removed from the missing sample buffer eventually arrives, it cannot be fused any longer.

The reason for resorting to the sub-optimal solution is the storage requirement (for storing the pre-computed Kalman gains): it grows exponentially with `ndopt= nd` for the optimal quasi-steady state filter. On the other hand, for `nmis = 1`, the storage grows linearly with `nd`.

Table 2 Parameters for block `SKalmanFilter_Yswitch`

No	Name	Description	Type	Group
1	`A`	*Discrete-time state-space model: A-matrix*	`n x n double array`	Process model
2	`B`	*Discrete-time state-space model: B-matrix*	`n x 1 double array`	
3	`C`	*Discrete-time state-space model: C-matrix*	`1 x n double array`	
4	`Q`	*Process noise covariance matrix: Q-matrix, symmetric, positive definite*	`n x n double array`	Noise model
5	`R`	*Measurement noise covariance, positive scalar.*	`double scalar`	
6	`nd`	*Sample buffer length/maximum assumed network delay in multiples of* `Ts`	`integer`	Network model
7	`ndopt`	*Optimal buffer data length/ maximum delay that is handled optimally; greater than or equal to 0, smaller than/equal to* `nd`	`integer`	
8	`nmis`	*Missing data buffer length: maximum number of unreceived samples of delay between* `ndopt` *and* `nd`*.*	`integer`	
9	`Fig`	*Figure number (for visualization of missing data buffers); set zero for no visualization.* ***It results in an error if two blocks share the same figure handle!!***	`integer`	Graphics
10	`pau`	*Pause in seconds in the simulation execution; done during the visualization routine. Not applicable if* `Fig=0`*.*	`double`	
11	`Ts`	*Control sampling interval; positive*	`double`	Sampling
12	`Tsmin`	*Network data sampling interval* T_{net}*; positive,* `Tsmin<Ts`	`double`	

Table 3 Input and output ports for block `SKalmanFilter_Yswitch`

No	I/O	Name	Description	Type	Sampling period	Direct feed-through
1	Inputs	`y`	*Data samples received via network.*	`double`	`Tsmin`	yes
2		`yts`	*Time stamps of data value at port #1 marking time of its acquiring by the sensor; valid, if non-negative; negative value indicates no new information received.*	`double`	`Tsmin`	yes
3		`u`	*Manipulated variable provided by the controller (assumed to be equal to the actuator input)*	`double`	`Ts`	no
1	Outputs	`yf`	*Filtered plant output*	`double`	`Ts`	-
2		`miss`	*Missing data indicator (if a sample is not received within delay* nd*, or the missing data buffer are overflown). An off-specs situation, sub-optimal behavior*	`{0,1}`	`Ts`	-

This S-function implements a visualization of the state of internal buffers. If parameter `Fig` is nonzero, the block creates a MATLAB figure of the same number. An error occurs if two blocks share the same figure number; this can easily happen when this block is cut-and-pasted in the Simulink model. The figure is shown in Figure 4. The columns denote data of a particular delay, specifically from 0 to 4. Delay `ndopt = 1` is marked by a yellow color of the digit. Duly received data are marked by a rectangle filled in the green colour. The rectangle with the yellow face denote samples currently not received, but conditionally fused by fake data and whose time stamps were recorded in the missing data buffer. The empty columns represent unreceived data: either those of delay smaller than or equal to `ndopt`, or those greater than `ndopt` that are not in the missing data buffer any more. The current simulation time is shown in the upper right corner. The text 'smpl loss' in the bottom right corner indicates data losses: If a data sample is considered lost for good by the filter, this textbox glows red, otherwise it is dark green. If the simulation is too fast to follow the graphics, it can be delayed by setting parameter `Pau` – it is pause in seconds, which is done after updating the plot. If there are several filter blocks in the model, these idle times accumulate and may slow down the simulation significantly. This visualization can be disabled by setting `Fig = 0`; in that case, parameter `pau` has no effect.

Finally, parameters `Ts` and `Tsmin` are set to define the control and communication sampling intervals, respectively. The user needs to ensure that other blocks connected to the filter have sampling times consistent with these.

The input ports are as described in the previous paragraph. One output port is the filtered controlled variable, and the other is a two-level function indicating the data loss: zero if none, one if a sample was lost in the current sampling interval. Data loss is a violation of prior assumptions; it occurs either if a data sample does not arrive after `nd` sampling intervals, or, a data stamp of a missing sample was removed from the missing data list due to its overflow. Occasional data loss does usually not cause a significant performance decrease.

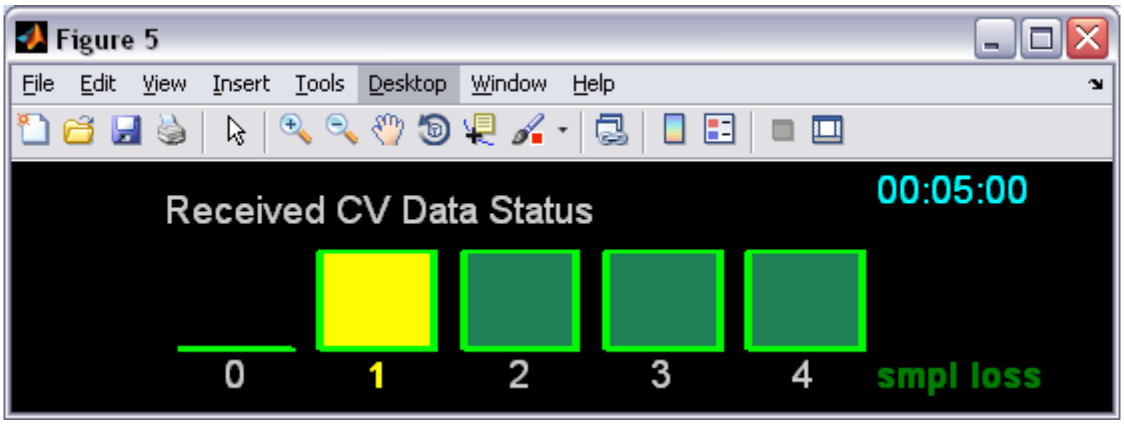


Figure 4 Visualization of received data in block `SKalmanFilter_Yswitch` (optional).

Block SKalmanFilter_YOpt

This block implements the time-varying optimal Kalman filter for process models augmented by a network model represented by a chain of delays. The algorithm is new, and the execution is about twice faster than that based on the classical square-root filter implementation and it is as robust. This new implementation exploits the model structure, uses some pre-computed gains and a specific decomposition of the state covariance matrix. The initialization is less demanding than that of the switched filter. Is need for data storage does not exceed that of the classical square root algorithm and it is much smaller than this required by the filter described in the previous paragraph. The on-line effort is larger, though. The difference to the switched quasi-steady-state filter is that the maximum network delay is not strictly required. Samples arriving with a longer delay, or not arriving at all are not considered, but the covariance matrix is updated correctly. The practical difference between these strategies may not be significant: the effect of old samples on the state estimate is likely relatively low, and hence, the sub-optimal treatment of lost samples by the quasi steady state filter may not degrade the performance, if data loss does not occur too often. For larger delays, moderate data storage needs may become more important than larger on-line computational load.

Parameters of this block are in Table 4. Compared to the parameter set of `SKalmanFilter_Yswitch`, the network is characterized by the maximum delay `nd` only. The input/output ports are as in Table 3. The visualization of the received data is in Figure 5.

Table 4 Parameters of block `SKalmanFilter_YOpt`

No	Name	Description	Type	Group
1	`A`	*Discrete-time state-space model: A-matrix*	`n x n double array`	Process model
2	`B`	*Discrete-time state-space model: B-matrix*	`n x 1 double array`	
3	`C`	*Discrete-time state-space model: C-matrix*	`1 x n double array`	
4	`Q`	*Process noise covariance matrix: Q-matrix, symmetric, positive definite*	`n x n double array`	Noise model
5	`R`	*Measurement noise covariance, positive scalar.*	`double scalar`	
6	`nd`	*Sample buffer length/maximum assumed network delay in multiples of* `Ts`	`integer`	Network model
7	`Fig`	*Figure number (for visualization of missing data buffers); set zero for no visualization.* ***It results in an error if two blocks share the same figure handle!!***	`integer`	Graphics
8	`pau`	*Pause in seconds in the simulation execution; done during the visualization routine. Not applicable if* `Fig=0`*.*	`double`	
9	`Ts`	*Control sampling interval; positive*	`double`	Sampling
10	`Tsmin`	*Network data sampling interval* T_{net}*; positive,* `Tsmin<Ts`		

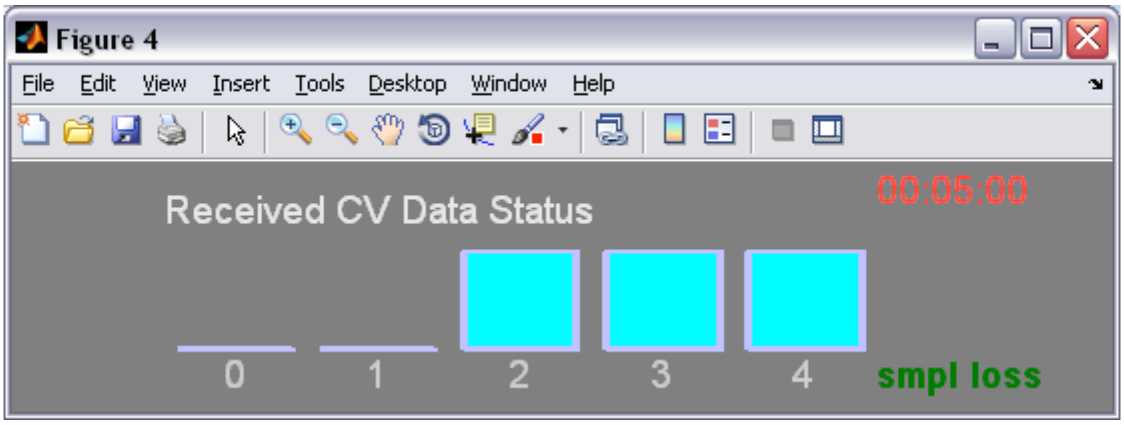


Figure 5 Visualization of received data in block `SKalmanFilter_YOpt`

Block SKalmanFilter_UY

This simulation block implements the quasi-steady-state/suboptimal filter as in `SKalmanFilter_Yswitch`. The added functionality is handling network delays in the actuator channel as in Figure 2. This filter does not have a stochastic network model and handles the delays in the input channel deterministically. It a priori assumes that the actuator receives its commands as were computed by the controller. The actuator actually applies the latest valid signal received from the network that is delayed by an integer multiple of the control sampling interval. Then, the actuator sends this delay information back to the filter with the time stamp attached. This is here, for brevity, referred to as the 'acknowledgement signal'. Thus block `SKalmanFilter_UY` has two additional input ports operating on the network sampling interval, one for the acknowledgement signal and the other one for its time stamp – see Table 6. Thus, at time $t = kT_s$, Kalman filter can reconstruct, using its own manipulated variable history, the value of the actuation signal which was applied at time $\left(k - d_{ack}(k)\right)T_s$, where $d_{ack}(k)T_s$ is the time stamp of the acknowledgement signal. Kalman filter then adjusts its state estimate in the way as if the true actuator input were applied in the past correctly. The communication network of the acknowledgement channel is assumed to have the same maximum signal delay `nd` as the measurement channel. Moreover, the filter has a buffer for time stamps of missing acknowledgement data whose length is `nmisU`; its maximum size is equal to `nd`. Minimum value of `nmisU` is zero; then, the block does not use the acknowledgement signal and behaves as `SKalmanFilter_Yswitch`. The output port denoted as `miss` becomes a two-dimensional, 2-level signal, the first dimension being for the measurement channel and the second one for the acknowledgement signal channel; value 0 means no detected (unrecoverable) sample loss in the last sampling period, value 1 means a sample loss in the respective channel.

The state of missing data and internal buffers can be visualized, as in Figure 6; data for the sensor and acknowledgement channels are handled separately.

Table 5 Parameters of S-function `SKalmanFilter_UY`

No	Name	Description	Type	Group
1	`A`	*Discrete-time state-space model: A-matrix*	`n x n double array`	Process model
2	`B`	*Discrete-time state-space model: B-matrix*	`n x 1 double array`	
3	`C`	*Discrete-time state-space model: C-matrix*	`1 x n double array`	
4	`Q`	*Process noise covariance matrix: Q-matrix, symmetric, positive definite*	`n x n double array`	Noise model
5	`R`	*Measurement noise covariance, positive scalar.*	`double scalar`	
6	`nd`	*Sample buffer length/maximum assumed network delay (in multiples of `Ts`) for both sensor and actuator channels*	`integer`	Network model
7	`ndopt`	*Optimal buffer data length/ maximum delay in the sensor channel that is handled optimally; greater than or equal to 0, smaller than/equal to `nd`*	`integer`	
8	`nmisY`	*Missing data time stamp buffer length for the sensor channel: maximum number of unreceived samples of delay between `ndopt` and `nd`.*	`integer`	
9	`nmisU`	*Missing data time stamp buffer length for the actuator acknowledgement channel: maximum number of unreceived samples of delay between `0` and `nd`.*	`integer`	
10	`Fig`	*Figure number (for visualization of missing data buffers); set zero for no visualization.* ***It results in an error if two blocks share the same figure handle!!***	`integer`	Graphics
11	`pau`	*Pause in seconds in the simulation execution; done during the visualization routine. Not applicable if `Fig=0`.*	`double`	
12	`Ts`	*Control sampling interval; positive*	`double`	Sampling
13	`Tsmin`	*Network data sampling interval T_{net}; positive, `Tsmin<Ts`*	`double`	

Table 6 Input/Output ports of simulation block `SKalmanFilter_UY`

No	I/O	Name	Description	Type	Sampling period	Direct feed-through
1	Inputs	`y`	*Data samples received via network.*	`double`	`Tsmin`	yes
2		`yts`	*Time stamps of data value at port #1 marking time of its acquiring by the sensor; valid, if non-negative; negative value indicates no new information received.*	`double`	`Tsmin`	yes
3		`u`	*Manipulated variable provided by the controller*	`double`	`Ts`	no
4		`ua`	*Actuator acknowledgement data: time delay of the actuation signal relative to the time of its generation by the controller*	`double`	`Tsmin`	yes
5		`uats`	*Time stamps of data value at port #4 marking time of its creation by the transmitter at the actuator side; valid, if non-negative; negative value indicates no new information received.*	`double`	`Tsmin`	yes
1	Outputs	`yf`	*Filtered plant output*	`double`	`Ts`	-
2		`miss`	*Missing data indicator (if a sample is not received within delay* `nd`*, or the missing data buffer are overflown). An off-specs situation, sub-optimal behavior*	`two-dimensional, two-level signal{0,1}`	`Ts`	-

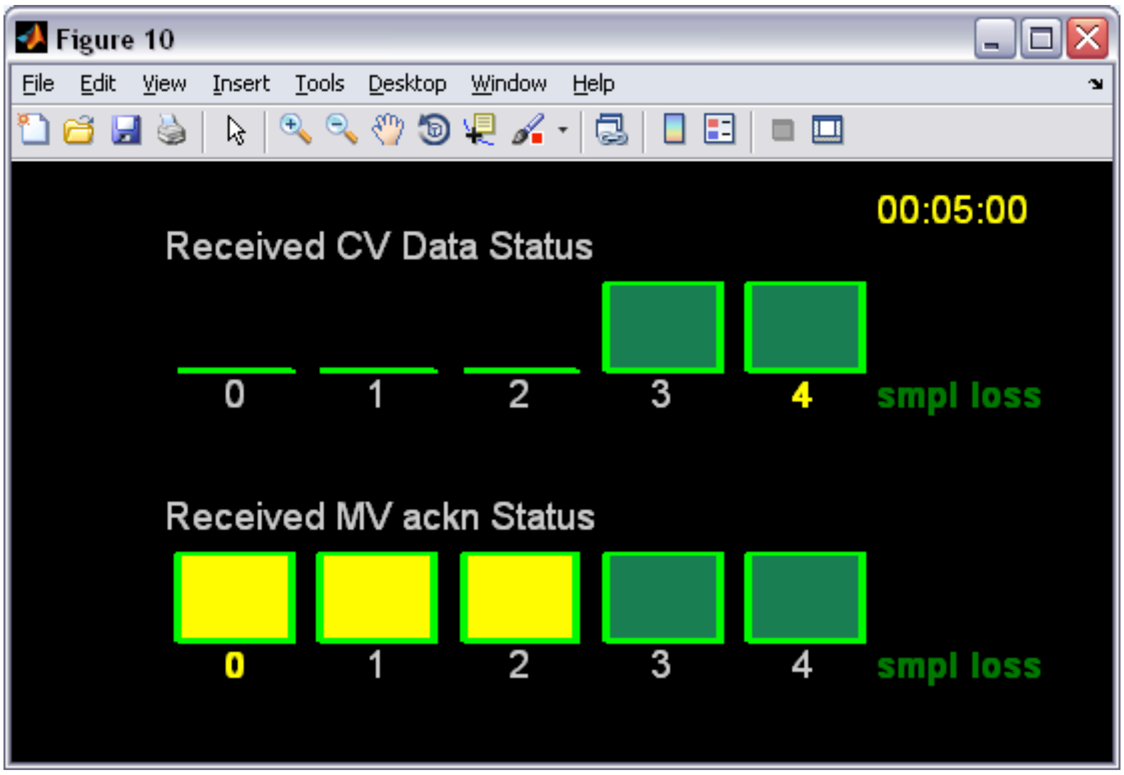


Figure 6 Received data visualization in block `SKalmanFilter_UY`

Supporting model blocks

Overview

In this section we shall overview the custom-made modeling blocks used in the demonstration model for the evaluation of net-aware Kalman filters. These blocks are level 2 M-file S-functions used for network simulation and interfacing the network model with the control system. These S-functiond are listed in Table 7. These block are specified in mode detail in the following paragraphs.

Table 7 S-functions for supporting simulation blocks

No	*S-function name*	*Functionality*
1	`msfcn_multi_hop`	A simple multi-hop wireless network model.
2	`msfcn_burst`	Gilbert-Elliot model of bursting disturbance.
3	`WNReceiver`	A simple interface of the network model holding the last good value of received data at its output and providing the time delay of its output signal relative to its transmission from the source.

Simulation block `msfcn_multi_hop`

Network model contains multiple data paths, each consisting of several buffers, as shown in Figure 7. Input data are written to the input buffer, from which it is sent to one of the data paths; the active data path is changed with a given probability: if the event of path change occurs, the active path is switched to the next one in the sequence; the last path switches to the first one.

The path is a series of data buffers. A probability of a successful transmission from one buffer to the next one is given: the probability is common for all buffers in the path. In the event of a successful transmission, the data item is removed from the source buffer and is written to the target buffer; otherwise, the data item waits in the source buffer till the next time slot. There exists a probability of the event that the transmission fails and the data item gets lost. The last buffer of each path is connected to the output buffer. The data accumulated in a path after it is switched to the inactive state are continued to be transmitted towards the output buffer. The probability of a successful transmission in all buffers of the network can be degraded in a coordinated way based on the logical-valued signal from an external input.

The network buffers operate at a sampling rate $1/T_{net}$ that is much higher than that of the controller. Usually, it is an integer multiple of the control sampling rate $1/T_s$. The input buffer appends the time stamp to the sample value. The output buffer outputs the signal value and time stamp separately. In the case when there is no valid data value on the output, the network model sets the time stamp to a negative value. The signal value may hold the last good value.

Network model parameters are summarized in Table 9, including the sampling intervals. We note that *the consistency of sampling rates with rates of other simulation blocks connected to the ports of this network model has to be ensured*. The overview of input/output ports is in Table 8.

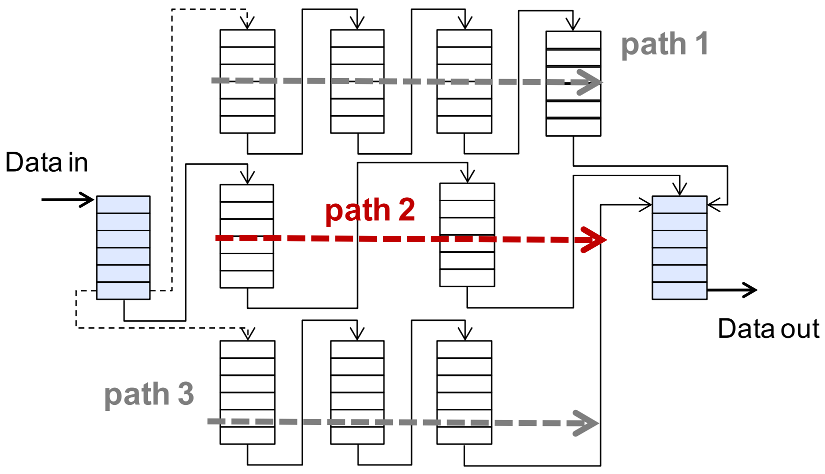


Figure 7 Model of a multi-hop network

Table 8 Ports of S-function `msfcn_multi_hop`

No	I/O	Name	Description	Type	Sampling period	Direct feed-through
1	Inputs	`IN`	*Data input.*	`double`	`Ts`	no
2		`Burst`	*Bursting disturbance, logical-valued. If zero, network operates in a standard way with pre-specified probabilities of successful transmission in parameter* `path_prob`*; if one, all transition probabilities are degraded by the factor* `burst_deg`.	`two-level signal{0,1}`	`Ts_net`	no
1	Outputs	`Out`	*Output data: Delayed data received at input port IN; If the output buffer is empty, the output value is zero when* `hold_output` *is false; otherwise, value from the last network sampling interval is hold.*	`double`	`Ts_net`	-
2		`Time`	*Time stamp of the data at output port#1; correspond to the time of arrival to the input buffer. If the output buffer is empty (no valid data at output #1), this port outputs value -1.*	`two-dimensional, two-level signal{0,1}`	`Ts_net`	-

Table 9 Parameters of S-function `msfcn_multi_hop`.

No	Name	Description	Type	Group
1	`Ts_net`	*Network data sampling interval T_{net}; positive,*	`double`	Timing
2	`Ts`	*Control sampling interval; positive* `Ts`> `Ts_net`	`double`	
3	`path_len`	*Path lengths: number of buffers in each transmission paths*	`double array 1x#paths`	Network
4	`path_prob`	*Probability of a successful transmission between buffers in a path.* ***Dimensions of*** `path_prob` ***and*** `path_len` ***must be identical!***	`double array 1x#paths`	
5	`p_switch`	*Probability of the path change*	`double`	
8	`p_lost`	*Probability of data loss in the transmission between buffers*	`double`	
9	`burst_deg`	*Burst degradation factor between 0 and 1: multiplies probabilities of successful transmission, if input port #2 is non-zero*	`double`	
7	`hold_output`	*If true, Output port #1 holds the last valid value (not necessarily with the latest time stamp)*	`logical`	Options
6	`raise_error`	*If true, an error is raised if some of the internal buffers is overflown.*	`logical`	
10	`buffer_type`	*Switch between FIFO/LIFO types of internal buffers*	`{FIFO,LIFO}`	
11	`synchronize`	*Pseudo-random number generator is initialized to a fixed sees and the state of the generator is kept as an internal state of the block. The network model gives repeatable results. If several network models have this parameter set to true, they have the same state of this generator; if other parameters are the same, they have the same network delays.*	`logical`	

Simulation block msfcn_burst

This S-function represents a source block – a burst disturbance generator. It has a single output port and provides a two-valued output, zero/one. Sampling time is inherited. The generator is a Gilbert-Elliot model of a bursting disturbance, a two-state Markov chain, characterized by two transitional probabilities: error burst probability `p_error`, and the recovery probability `p_recovery`. The purpose of this block is to simulate, a bursting disturbance affecting the overall communication network via input ports #2 of network models `msfcn_multi_hop`.

Simulation block `WNReceiver`

This block is used to interface the network model to blocks of the control chain other than net-aware Kalman filters, operating on the slower control rate. This block receives its input from the network model, together with time stamps. Then, it outputs the latest value of the received data (the latest in terms of the time of sending by the data source, not the reception at the network output) at output port #1, and, at port #2, the time delay (in multiples of the control sampling period) between the current time and the time stamp of the data value currently at port #1. This block is used in multiple ways in the demo models. First, if the manipulated variable is transmitted over the network, it is the interface between the communication network and the actuator. In that case, the time delays at output port#2 can be used as the acknowledgement signal to be sent to net-aware Kalman filter `SKalmanFilter_UY`. Further, it can be connected to the network channel communicating process measurements for the 'network un-aware feedback' to the PID controller, or as an interface of a variable delay *Smith Predictor*, which is a network-aware control strategy, less sophisticated than that using net-aware Kalman filters.

Table 10 Parameters of block `WNReceiver`

No	Name	Description	Type	Group
1	`nd`	*maximum delay in multiples of* `Ts`	`double`	Network
2	`Ts`	*Control sampling interval; positive* `Ts`> `Ts_net`	`double`	Sampling
3	`Ts_net`	*Network data sampling interval* T_{net}*; positive.*	`double`	

Table 11 Ports of block `WNReceiver`

No	I/O	Name	Description	Type	Sampling period	Direct feed-through
1	Inputs	`y`	*Data samples received via network.*	`double`	`Tsmin`	yes
2		`yts`	*Time stamps of data value at port #1 marking time of its acquiring by the sensor; valid, if non-negative; negative value indicates no new information received.*	`double`	`Tsmin`	yes
1	Outputs	`yout`	*Value corresponding to the latest time stamp at the input*	`double`	`Ts`	-
2		`ack`	*delay between the current time and the time stamp of the sample value at output #1, in multiples of period* `Ts.`	`integer`	`Ts`	-

Demonstration model 1: WSN in the measurement channel

The Simulink model demonstrating multiple network-aware strategies in handling wireless network in the loop is in file `Y_Channel_Sim_PID_04.mdl`. This model is shown in Figure 8. Detailed simulational results can be found in the accompanying document xxxx.

The main elements of the model are as follows:

- Plant model introduced as a continuous-time transfer function model, corrupted by measurement and process input noises. Process input noise is low-pass filtered to simulate a slowly varying disturbance.
- Controller: discrete-time PID controller operated at sampling period `Ts` that is passed to this controller as a parameter, along with gains `P`, `I` and `D`, see Figure 9.. Implemented as a masked subsystem. It takes the difference between the set-point and the process output feedback as an input. The feedback can be taken from multiple sources, depending on the setting of the multi-port switch, named as *Feedback Switch*. The feedback options include

    1. Quasi-steady state Kalman filter with maximum delay set to 4, denoted as Optimal switched KF. Parameters are set via a mask with grouped parameters and prompted edit-boxes, see Figure 10. Parameter grouping is as in Table 2.
    2. Sub-optimal Kalman filter (`nd = 4, nopt = 1, nmis = 1`) denoted as Suboptimal switched KF.
    3. Optimal time-varying Kalman filter, `nd = 4`, denoted as Optimal varying KF.
    4. Optimal time-varying Kalman filter, `nd = 2`, denoted as Optimal varying KF mem length 2.
    5. Optimal time-varying net-unaware Kalman filter ignoring time stamps, taking values available at the first time slot in the sampling interval.
    6. Optimal Kalman filter *wired* to the sensor.
    7. Wired sensor data.
    8. Latest known process value produced by WSN receiver. This is a network un-aware feedback strategy.
    9. Process value predicted by a Smith predictor with variable delay (connected to WSN receiver). Smith predictor uses the same discrete-time state-space

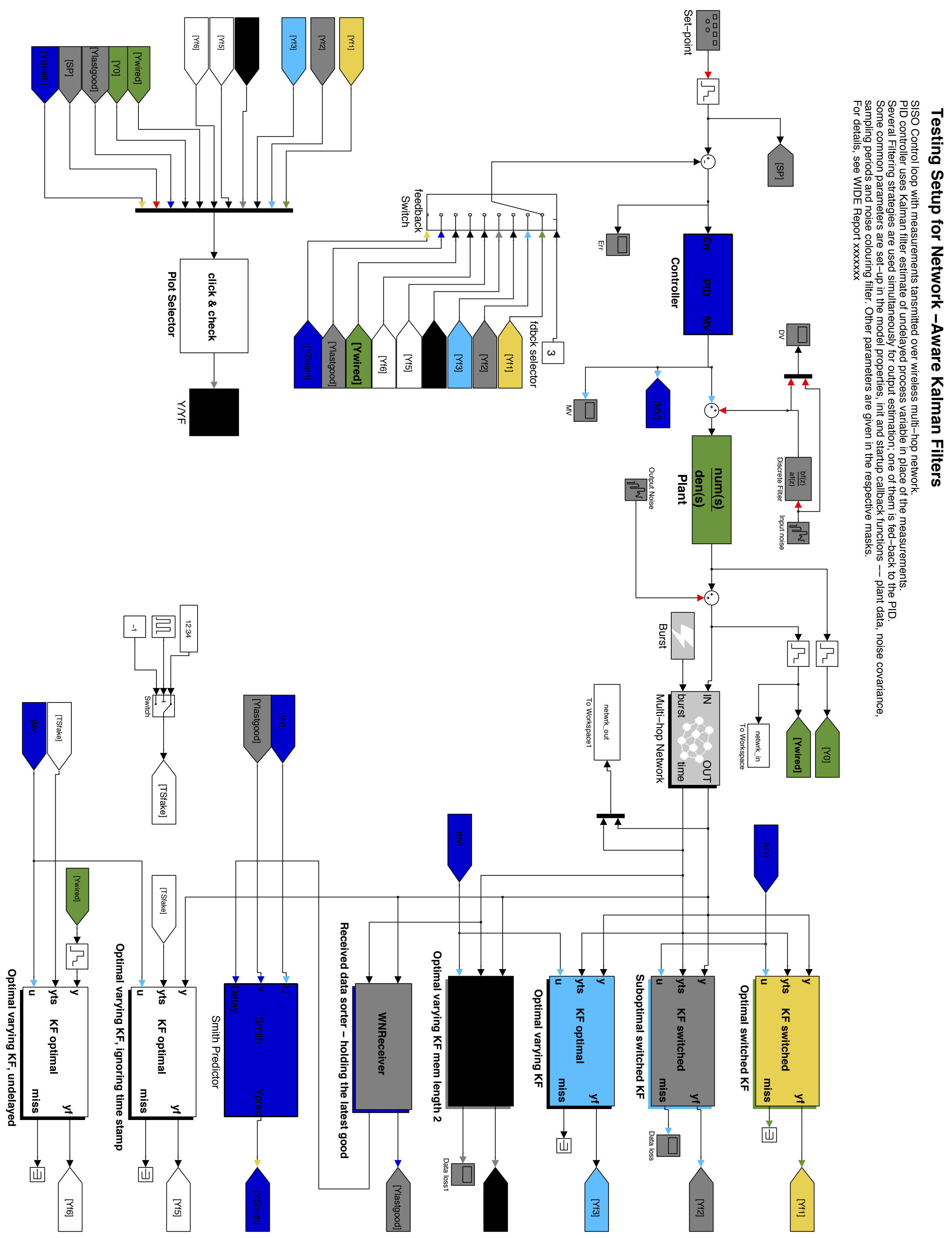
representation as Kalman filters and this model is set in the mask, see Figure 11. It is implemented as a masked sub-system. It uses an initialization script to augment the state-space model by an appropriate number of delays.

- Network-aware Kalman filters 1—4 and the WNReceiver 8—9 above are connected to the process via a wireless network model Multi-hop Network. Parameters are easily entered via a customized mask with prompted edit boxes and grouping as in Table 10.
- Plant output (raw/uncorrupted), Set-point and data from all sources of data are aggregated by a multiplex block. The resulting vector is passed, via a simple selector to a scope. Variables to be plotted are selected by check-boxes.

To ensure consistency of the models used by the above filters, as well as correct sampling rates, parameters common to multiple blocks are set in callback scripts. These scripts are accessible from Model Properties (right-click on any empty spot of the model), Pre-loadFcn and StartFcn Callbacks (both are identical). The script is as follows:

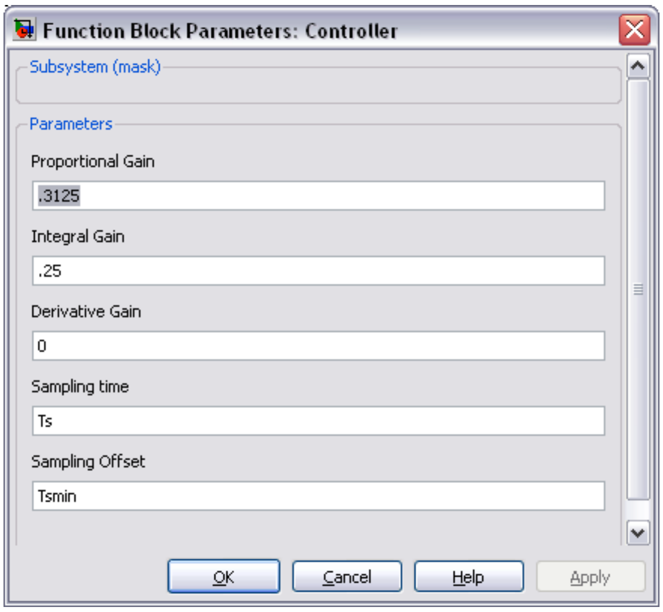
```
%plant model
num=1.2; den=conv([1.3,1],[3.5,1]);
%num=1;  den=conv([1.75,1],[3.5,1]);  %uncomment to set the plant
                                      model equal
                                      %to the perturbed one
%perturbed model
num_p=1; den_p=conv([1.75,1],[3.5,1]);
sys=tf(num_p,den_p);
%PID data
P=.875; I=.25; D=0;
%state-space, continuous time representation
[Ac,Bc,Cc,Dc]=ssdata(sys);
%Control sample time
Ts=1;
%discretization
sysd=c2d(sys,Ts);
[Ad,Bd,Cd,Dd]=ssdata(sysd); nd=size(Ad,1);
A0=[Ad,Bd;zeros(1,nd),1];
B0=[Bd;0]; C0=[Cd,0]; D0=Dd;
Bx=[zeros(nd,1);1];
%noise data
Q0=1e0*(B0*B0')+1e2*(Bx*Bx');
R0=1;
%network sampling time
Tsmin = 0.1;
%noise coloring filter coefs

[bf,af]=butter(3,.1);
```

The above script presumes the plant model to be different from the model assumed by Kalman filters. Kalman filter models are nominal plant state-space models augmented by an input disturbance observer to achieve offset-free disturbance attenuation.


Testing Setup for Network –Aware Kalman Filters

SISO Control loop with measurements tansmitted over wireless multi–hop network.
PID controller uses Kalman filter estimate of undelayed process variable in place of the measurements.
Several Filtering strategies are used simultaneously for output estimation; one of them is fed–back to the PID.
Some common parameters are set–up in the model properties, init and startup callback functions –– plant data, noise covariance,
sampling periods and noise colouring filter. Other parameters are given in the respective masks.
For details, see WIDE Report xxxxxxx

Figure 8 Simulink model with wireless network in the measurement channel: `Y_Channel_Sim_PID_04.mdl`


Function Block Parameters: Controller

Subsystem (mask)

Parameters

Proportional Gain: .3125

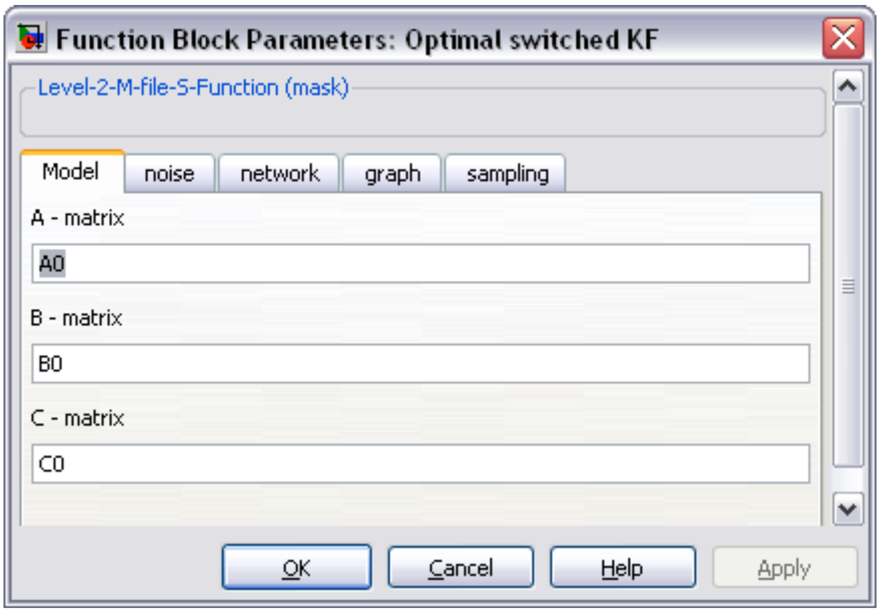
Integral Gain: .25

Derivative Gain: 0

Sampling time: Ts

Sampling Offset: Tsmin

Figure 9 PID Controller Mask


Function Block Parameters: Optimal switched KF

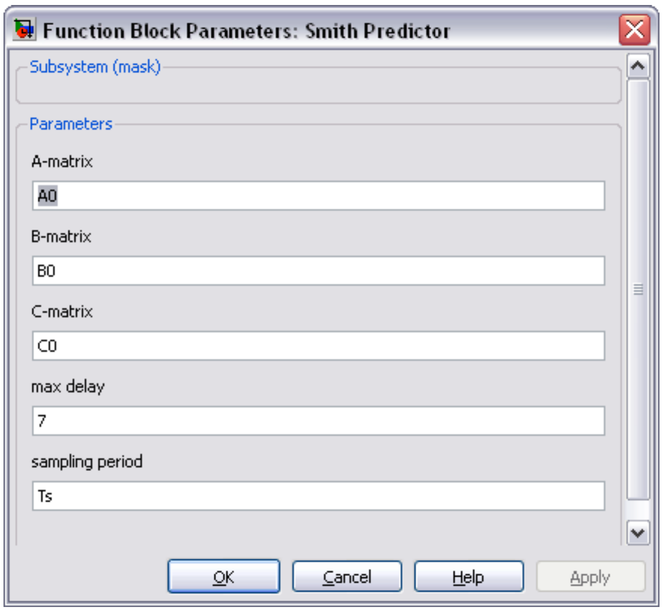
Level-2-M-file-S-Function (mask)

A - matrix: A0

B - matrix: B0

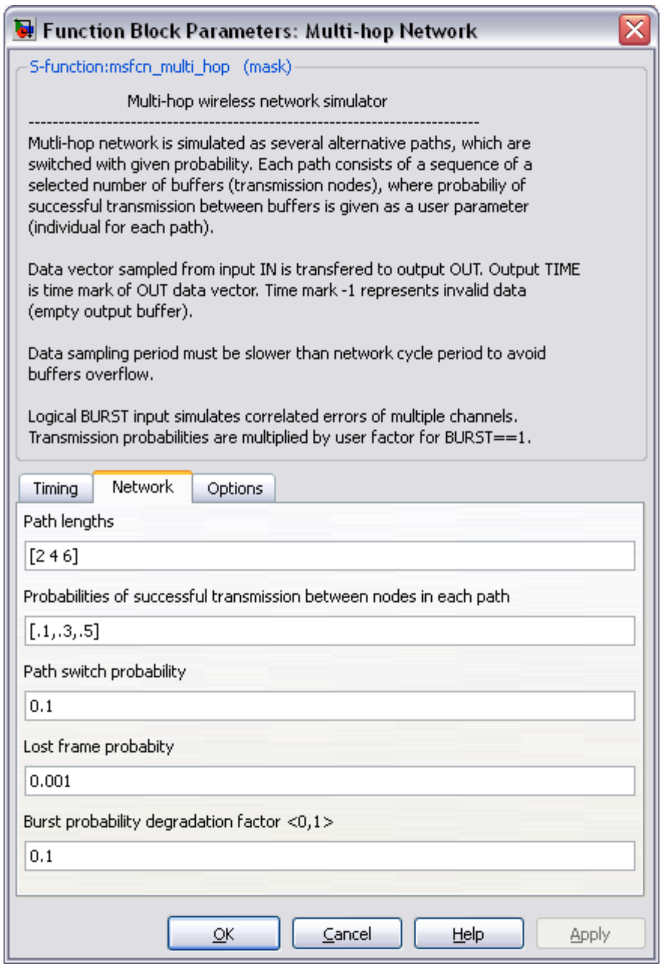
C - matrix: C0

Figure 10 Mask for network-aware Kalman filters.



A-matrix
A0
B-matrix
B0
C-matrix
C0
max delay
7
sampling period
Ts

Figure 11 Mask of Smith Predictor


Function Block Parameters: Multi-hop Network
S-function:msfcn_multi_hop (mask)

Multi-hop wireless network simulator

Mutli-hop network is simulated as several alternative paths, which are switched with given probability. Each path consists of a sequence of a selected number of buffers (transmission nodes), where probabiliy of successful transmission between buffers is given as a user parameter (individual for each path).

Data vector sampled from input IN is transfered to output OUT. Output TIME is time mark of OUT data vector. Time mark -1 represents invalid data (empty output buffer).

Data sampling period must be slower than network cycle period to avoid buffers overflow.

Logical BURST input simulates correlated errors of multiple channels. Transmission probabilities are multiplied by user factor for BURST==1.

Path lengths
[2 4 6]
Probabilities of successful transmission between nodes in each path
[.1,.3,.5]
Path switch probability
0.1
Lost frame probabity
0.001
Burst probability degradation factor <0,1>
0.1

Figure 12 Block mask for the network model

Demonstration model 2: Network in the measurement and actuator channels

The Simulink model demonstrating multiple network-aware strategies in handling wireless network in the loop is in file `YU_Channels_Sim_PID_03.mdl`. This model is shown in Figure 13. Simulational results can be found in the accompanying document xxxx.

The main elements of the model are as follows:

- Plant and controller models are as described in the previous section. As the communication delay is present in both sensor and actuator channels, tuning of PID controller is now more conservative.
- Three independent wireless network models are present, for the measurement, actuation and the actuation signal acknowledgement channels. They are subject to a common bursting disturbance.
- The feedback options for PID controller include
    1. Plant output estimate by Kalman filter aware of the network in sensor channel only, with manipulated variable input being wire-connected to the actuator. Implemented as a quasi-steady state optimal switched filter, labeled as Opt switched KF - applied MV.
    2. Plant output estimate by Kalman filter aware of the network in sensor channel only, with manipulated variable input being wire-connected to the controller. Implemented as a quasi-steady state optimal switched filter, labeled as Opt switched KF - computed MV.
    3. Plant output estimate by Kalman filter aware of the network in both channels. Its manipulated variable input is wire-connected to the actuator, but it receives acknowledgement signals that are used for correcting the internal estimate. It is labeled as Opt switched KF computed MV/corrected.
    4. Raw sensor measurement.

    Plant model as well as Kalman filter data and sampling periods are specified in callback functions, as described in the previous section.

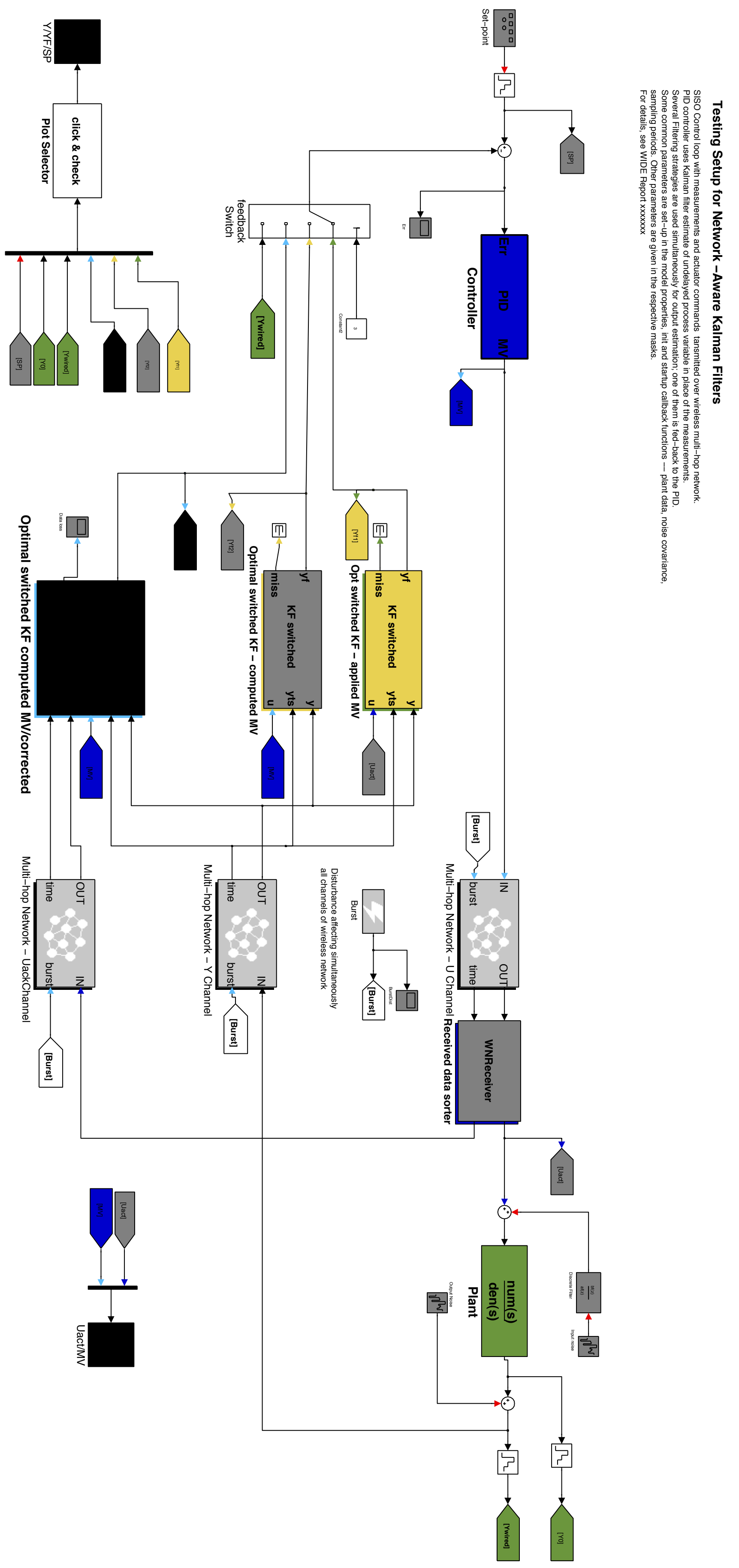

Testing Setup for Network–Aware Kalman Filters
SISO Control loop with measurements and actuator commands transmitted over wireless multi–hop network.
PID controller uses Kalman filter estimate of undelayed process variable in place of the measurements.
Several Filtering strategies are used simultaneously for output estimation; one of them is fed–back to the PID.
Some common parameters are set–up in the model properties, init and startup callback functions — plant data, noise covariance,
sampling periods. Other parameters are given in the respective masks.
For details, see WIDE Report xxxxxxx
Set–point
[SP]
Err
Err PID MV
Controller
feedback Switch
[Ywired]
Multi–hop Network – U Channel
IN burst OUT time
[Burst]
WNReceiver
Received data sorter
[Uact]
num(s)
den(s)
Plant
[Y0]
[Ywired]
Burst
Disturbance affecting simultaneously all channels of wireless network
Multi–hop Network – Y Channel
Multi–hop Network – UackChannel
KF switched
yf miss y yts u
Opt switched KF – applied MV
Optimal switched KF – computed MV
Optimal switched KF computed MV/corrected
[Y1]
[Y2]
click & check
Plot Selector
Y/YF/SP
Uact/MV

Figure 13 Simulation model `YU_Channels_SIm_PID_03`.

References

[1] Anderson, B.D.O., Moore, J.B.: *Optimal Filtering.* Prentice Hall, NJ, 1979.

[2] Pachner, D., Havlena, V.: An Approach to Out-Of-Sequence Measurements in Feedback Control Systems. Proceedings of 6*th IEEE International Workshop on Factory Communication Systems*, Dresden, Germany, 2008.

[3] Baramov, L., Pachner, D. and Havlena, V.: Kalman Filter for Systems with Communication Delay Proceedings of *1st IFAC Workshop. on Estimation and Control of Networked Systems* NecSys2009, Venice, Italy, September 2009.

[4] Pachner, D. Baramov, L. and Havlena, V.: Suboptimal State Estimation under Communication Delay. *Proceedings of 9th IFAC Workshop of Time Delay Systems*, Prague, Czech Republic, 2010.