Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Performance Analysis of Distributed Control Systems Using the FlexRay Protocol

Thiago J. Michelin * **João M. G. da Silva, Jr.** **
Carlos E. Pereira ***

* *Electrical Engineering Department, Federal University of Rio Grande do Sul, Porto Alegre, 90035-190, Brazil (e-mail: thiago.michelin@ufrgs.br)*
** *(e-mail: jmgomes@ece.ufrgs.br)*
*** *(e-mail: cepereira@ece.ufrgs.br)*

Abstract
Spatially distributed systems which share communication resources have advantages such as lower implementation cost, better flexibility and reliability. When real-time distributed control applications make use of such architecture, the drawbacks are the networked-induced effects, which may severely degrade the controlled system performance. Efforts have been made by both communication and control communities to assess this problem: new control design techniques now consider explicitly the effects of these constraints, and some new communication protocols offer services to support the requirements from this kind of application. FlexRay is a protocol developed to support real-time safety-critical applications. In this study, a FlexRay network is built and an active suspension control is implemented in a distributed environment. The main goal is to analyse the impact the network-induced effects may bring to system performance.

1. INTRODUCTION

Networked control systems (NCS) are systems with spatially distributed components, where sensors, controllers and actuators exchange information via shared communication networks [Antsaklis and Baillieul, 2007]. This architecture concept has advantages such as lower implementation and maintenance costs, and greater flexibility [Cloosterman et al., 2010]. These advantages are particularly important for in-vehicle applications, since the space available for implementation is limited [Zhang et al., 2013].

From a dynamic control perspective though, communication through a shared channel introduces some constraints that may deteriorate system performance, or even cause instability [Zhang et al., 2001]. Among these constraints, one may mention: time delays, packet losses and quantization [Zhang et al., 2013]. To deal with network-related effects, control algorithms must consider explicitly these characteristics [Cloosterman et al., 2010]. In the literature, there are several approaches developed to study stability and controller synthesis conditions for NCSs [Zhang et al., 2013][Hespanha et al., 2007].

Within this context, the services provided by the communication protocol are of utmost importance and have great impact on system modelling. This is specially true for safety-critical systems, such as the so-called X-by-wire systems, where the bus is considered the core of the application built above it [Rushby, 2001].

In this work, we are interested in assessing the performance of control systems when communicating through a FlexRay bus. FlexRay has been developed for the next generation of in-vehicle applications, and supports both time-triggered and event-driven paradigms. The main goal is to analyse the impact that different sampling periods bring to the closed loop control system, using different control strategies and different scheduling policies. As a case study, the system known as active suspension is modelled and then implemented in a distributed environment.

This paper is organized as follows: section 2 reviews the main characteristics of the FlexRay protocol. Section 3 describes the case study of this work, namely the active suspension system, represented by a quarter vehicle model. In section 4, control objectives for the suspension system are discussed, along with synthesis conditions of three different control strategies. As of section 5, details and issues of the actual system's implementation on a distributed environment are discussed. Analysis of results is carried out in section 6. Finally, conclusions are stated in section 7.

2. FLEXRAY OVERVIEW

The development of the FlexRay protocol started in the early 2000's by the FlexRay Consortium, and is considered to become the next generation standard for in-vehicle networking [Makowitz and Temple, 2006].

2.1 Topology

A FlexRay cluster supports a maximum of two communication channels, identified as Channel A and Channel B. Each ECU in the cluster might be connected to only one channel or both of them at the same time.

There are several ways to build a cluster, such as point-to-point connections, bus network, passive or active star network, or various other hybrid combinations of the mentioned topologies [FlexRay Consortium, 2010].

2.2 Media Access Control (MAC)

The media access control of FlexRay is based on a cyclic communication scheme [FlexRay Consortium, 2010], where each cycle has the same period length and is comprised of a static segment (SS), a dynamic segment (DS), a symbol window (SW) and the network idle time (NIT) section. Among these four segments, only the first and last ones are mandatory on every FlexRay cycle. An example of a communication cycle is shown in figure 1.

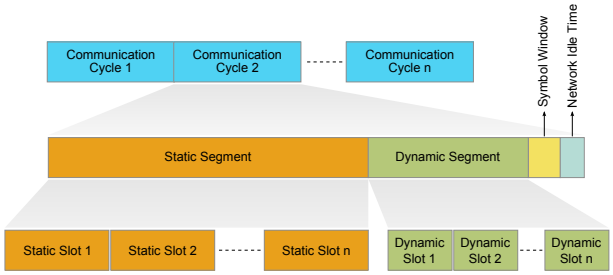

Figure 1. FlexRay Communication Cycle. Based on [FlexRay Consortium, 2010].

Static Segment (SS) The static segment is based on TDMA (Time Division Multiple Access), and is composed of time-slots. Each time-slot has the same period, as each communication cycle has the same number of time-slots [FlexRay Consortium, 2010].

Dynamic Segment (DS) The dynamic segment is based on FTDMA (Flexible Time Division Multiple Access), where mini time-slots (with the same period) are put together to form a bigger time-slot, according to requirements of the message to be transmitted [FlexRay Consortium, 2010]. In this segment, message scheduling makes use of a priority scheme, where a message with lower ID has higher priority.

2.3 Framing

A FlexRay frame is composed of three segments: header, payload and trailer. Figure 2 illustrates the logical scheme of a FlexRay frame.

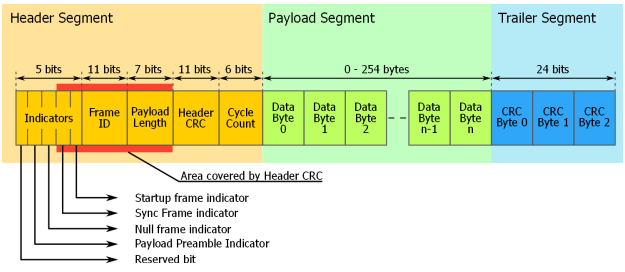

Figure 2. FlexRay Frame. Based on [FlexRay Consortium, 2010].

Apart from the first bit (reserved bit), the first five bits of the header segment are indicators specifying certain characteristics of the frame, such as indicating whether or not the frame is used for synchronization and startup. The frame ID is composed of eleven bits and defines the position of the frame within the current communication cycle. In the dynamic segment, the frame ID also specifies the priority of the frame (a lower identifier indicates higher priority). The next field is the payload length, which indicates how many data bytes the frame has in its Payload segment. The following field is a CRC sequence of eleven bits, used to check for errors over the Sync frame indicator bit, the Startup frame indicator bit, the frame ID sequence and the payload sequence. The payload segment may contain up to 254 bytes of data. Finally, the trailer segment contains a CRC calculated over the entire header and payload segments.

3. CASE STUDY

The case study chosen for this work is an active suspension system. The main goals of a suspension system are to isolate the vehicle's chassis from the disturbances caused by road irregularities and, at the same time, provide good handling by maintaining the contact between road and tyre at all times [Poussot-Vassal, 2008]. This way, the design of a suspension system is always a trade-off between comfort and handling [van der Sande et al., 2012].

3.1 Modelling

For the study of this kind of system, it is very common to find in the literature the model known as *quarter-car model* (figure 3). Through the use of this model it is possible to study the vertical behavior of a car, without having to consider the whole vehicle dynamics, thus simplifying the analysis.

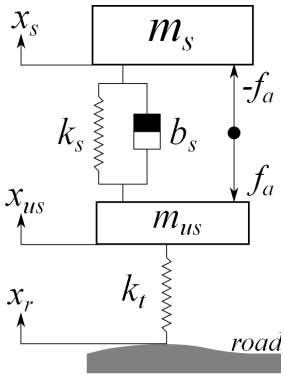

Figure 3. Quarter-car model. Based on [Poussot-Vassal, 2008].

From the model, the sprung mass (m_s) represents the mass of the vehicle, the unsprung mass (m_{us}) represents the set formed by tyre, wheel, suspension and all its auxiliary devices that provide the "link" between chassis and road [Poussot-Vassal, 2008]. k_s and k_t are the suspension spring stiffness and the spring constant used to model the tyre, respectively. b_s is the damper's coefficient. As variables, x_s, x_{us}, x_r and f_a are the vertical body travel, unsprung mass movement (considered to have its centre of gravity on the wheel mounting point), road disturbance and actuator force, respectively.

The dynamical equations of the quarter vehicle model are given by (1) and (2).

$$m_s\ddot{x}_s = -k_s(x_s - x_{us}) - b_s(\dot{x}_s - \dot{x}_{us}) - f_a \quad (1)$$

$$m_{us}\ddot{x}_{us} = k_s(x_s - x_{us}) + b_s(\dot{x}_s - \dot{x}_{us}) + k_t(x_r - x_{us}) + f_a \quad (2)$$

The quarter car model considered in this work is based on the model presented in [Do et al., 2011], which uses specific suspension parameters of a "Renault Mégane Coupé", which are: $m_s = 315\ kg$, $m_{us} = 37,5\ kg$, $k_s = 29500\ N/m$, $k_t = 210000\ N/m$ and $b_s = 1000\ Ns/m$.

3.2 State-Space Representation

From equations (1) and (2) it is possible to obtain the state space description for the quarter car model, given by (3).

$$\dot{x} = Ax + B_1 x_r + B_2 f_a \quad (3)$$

Considering $x = [x_s - x_{us} \quad \dot{x}_s \quad x_{us} - x_r \quad \dot{x}_{us}]^T$ as the state vector, the system matrices are given by (4).

$$A = \begin{bmatrix} 0 & 1 & 0 & -1 \\ -\frac{k_s}{m_s} & -\frac{b_s}{m_s} & 0 & \frac{b_s}{m_s} \\ 0 & 0 & 0 & 1 \\ \frac{k_s}{m_{us}} & \frac{b_s}{m_{us}} & -\frac{k_t}{m_{us}} & -\frac{b_s}{m_{us}} \end{bmatrix}$$

$$B_1 = [0\ 0\ -1\ 0]^T \quad (4)$$

$$B_2 = \begin{bmatrix} 0 & \frac{1}{m_s} & 0 & -\frac{1}{m_{us}} \end{bmatrix}^T$$

4. CONTROL OBJECTIVES AND CONTROLLER SYNTHESIS

To evaluate system performance, four transfer functions are most often used [Poussot-Vassal, 2008]:

- $\ddot{x}_s/x_r$: from road disturbance to vertical body acceleration. This transfer function evaluates comfort at high frequencies, between 4 and 30 Hz.
- x_s/x_r: from road disturbance to vertical body movement. This transfer function evaluates comfort at low frequencies, between 0 and 5 Hz.
- x_{us}/x_r: from road disturbance to vertical wheel movement. This transfer function evaluates road holding in the range [0, 20] Hz.
- x_{def}/x_r: from road holding to suspension deflection, which is defined as $x_{def} = x_s - x_{us}$. This transfer function represents a constraint on the deflection of the actuator, evaluated in the range [0, 20] Hz.

Two different control strategies were applied for the analysis of the distributed active suspension system:

1) Pole placement via state feedback;
2) State feedback considering sampled-data approach.

4.1 Pole Placement State-Feedback Controller

The first controller developed is a simple state-feedback, of the form of (5), for closed loop pole placement [Moore, 1976]. The closed loop poles were chosen so as to increase the system's damping factor (ζ).

$$f_a = K_{pp}x \quad (5)$$

4.2 Sampled Data State-Feedback Controller

The third controller is a state-feedback considering the sampled data approach. In this methodology, the system is modelled as a continuous time system with delayed input, of the form of (6), where $\tau(t)$ is the control input delay and t_k is the sampling instant.

$$u = Kx(t - \tau(t)), \quad \tau(t) = t - t_k \quad (6)$$

The goal is to find a matrix K_{dde} that guarantees exponential stability with decay rate α and a maximum control input delay $h = 10.5\ ms$. From the solution of the LMI presented by (7) (which is based on Gomes da Silva Jr. et al. [2011]), it is possible to find the static gain matrix given by (8).

$$\begin{bmatrix} 2\alpha Q_1 + AQ_2^T + Q_2A^T - 2he^{-2\alpha h}\tilde{R} & & \\ * & & \\ * & & \\ Q_1 + Q_2^T + Q_2A^T\xi & BY + 2he^{-2\alpha h}\tilde{R} & \\ h^2\tilde{R} - \xi(Q_2^T + Q_2) & \xi BY & \\ * & -2he^{-2\alpha h}\tilde{R} & \end{bmatrix} < 0 \quad (7)$$

$$K_{dde} = Y(Q_2^T)^{-1} \quad (8)$$

4.3 Simulation Results

For evaluation purpose, simulations were run using MATLAB/Simulink, where the transfer function $x_{def}/\dot{x}_r$ is considered. The road profile used as disturbance is described by the function given by (9).

$$x_r = \begin{cases} 0.025 + 0.025\text{sen}(2\pi t - \pi/2), & 0 \le t \le 1\ s \\ 0, & t > 1\ s \end{cases} \quad (9)$$

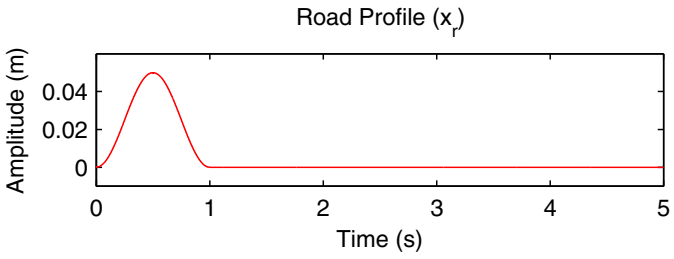

Figure 4. Road Profile

Figure 6 shows the system's response to the disturbance.

5. EXPERIMENTAL SETUP

For experimental evaluation of the system, two VN8910 devices with VN8970 plug-in modules, together with CANoe.FlexRay software [Vector Informatik GmbH, 2013] were available.

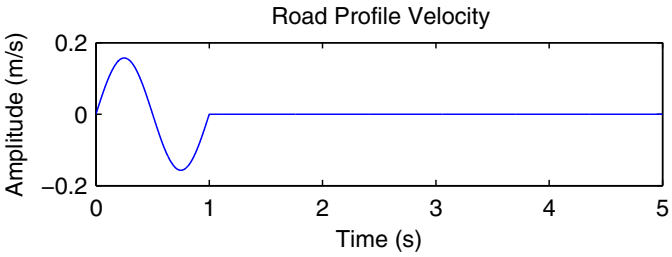

Figure 5. Road Profile Velocity

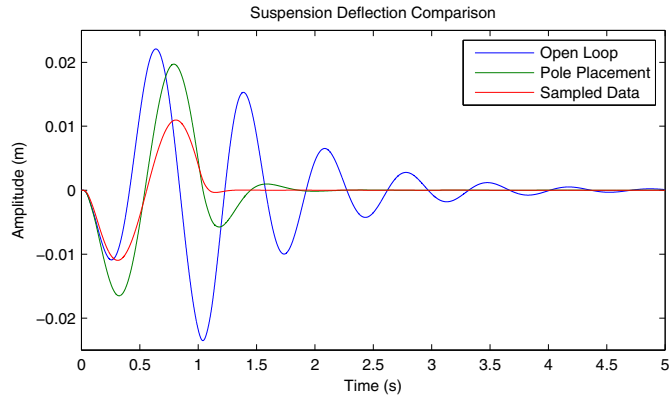

Figure 6. Disturbance Response Comparison

5.1 Logical and Physical Setups

Logical setup is illustrated by figure 7. In this configuration, sensors are time-driven, sampling the plant at a constant rate. After sampling, the newly acquired information is sent through the bus to a remote controller. The controller is event-driven, executing its algorithm as soon as new information arrives. After the new control value is available, the controller sends it through the bus to the actuator, which is also event-driven, and will change its outputs as soon as the new control law arrives.

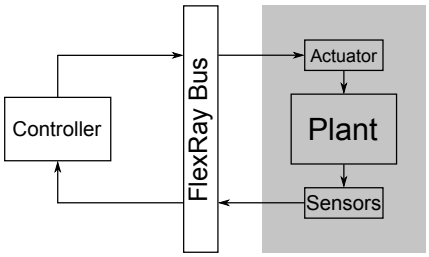

Figure 7. Logical setup. Based on [Hespanha et al., 2007].

Physical arrangement is shown by figure 8. One of the VN8910 devices (VN01) is responsible for simulating the plant, actuator and sensors. This device is connected to a PC running CANoe.FlexRay, which collects all data from communication and system variables for later analysis. The other VN8910 (VN02), runs on standalone mode, and is responsible for executing control algorithms. All components within this system were implemented using the CAPL language.

5.2 Timing Analysis

As each device runs only one process, there is no competition for CPU processing time. This way, the most

Figure 8. Physical Setup.

significant sources of delay are buffer update time by the communication controller, and the waiting time for the device to gain access to the bus (i.e., waiting time for its assigned time-slot). In order to evaluate these conditions, a test was carried out with both VN8910 devices to determine their update buffer time requirements, prior to running the system under study itself. The test consists of device VN01 sending two different data packages to VN02, and then checking if its reply matches the expected results, within the same communication cycle.

Protocol Parameters FlexRay protocol has over 70 parameters distributed among cluster global and specific node parameters. Table 1 illustrates the main protocol configuration parameters used during timing tests.

Table 1. Protocol parameters used for time delay tests

Parameter	Value	Unit
Communication Cycle Period	1000	μs
Bit	0.1	μs
Macrotick (MT)	1	μs
Number of Static Slots per Cycle	29	–
Static Slot (SS)	33	MT
Payload Length (SS)	8	$WORD$

As can be seen from table 1, the communication cycle in this configuration has a period of 1 ms and does not have a dynamic segment.

VN01 test On the VN01 test, the goal is to measure the time needed for the device to update its buffer right after sensor sampling. Sensor sampling is set to be performed at the beginning of each communication cycle. Since it is not possible to measure the device's buffer update time directly, the results will have to be based on the communication cycle timing. The test consists on VN01 updating its buffer with two different a priori known values, and then check on which time slot, within the same communication cycle, these values will be transmitted through the bus.

Executing the test, with the configuration illustrated by table 1, the correct values appeared in the bus only during time slot number 14, meaning a delay in the interval $[429,\ 462]\ \mu s$.

VN02 test The test carried out by the VN02 is very similar to the previous one. The goal is to evaluate how much time the device needs to decode incoming packets, calculate the new control law and update its buffer for

transmission. Once again, the measurements rely on the communication cycle timing.

The correct results expected from VN02, wouldn't appear until time-slot 28, meaning a delay in the interval $[429,\ 462]\ \mu s$.

5.3 Scheduling

Based on the results obtained in the last section, the following two scheduling policies were elaborated.

Static scheduling with 5 ms cycle period The first scheduling policy is based on the default configuration shipped with CANoe, as illustrated by table 2.

Table 2. Main standard protocol parameters

Parameter	Value	Unit
Communication Cycle Period	5000	μs
Bit	0.1	μs
Macrotick (MT)	1.375	μs
Number of Static Slots per Cycle	91	–
Static Slot (SS)	24	MT
Payload Length (SS)	8	$WORD$
Mini Slot	5	MT
Number of Mini Slots	289	–

Taking into account the time intervals obtained during the tests, time slot allocation was elaborated according to table 3.

Table 3. Scheduling Policy I

ECU	Time-slot
VN01 (Plant)	20
VN02 (Controller)	40

Static scheduling with 1 ms cycle period In this case, a new communication cycle with shorter period has been considered, with the cluster's main parameters being the ones shown by table 1. Time slot allocation was elaborated according to table 4.

Table 4. Scheduling Policy II

ECU	Time-slot
VN01 (Plant)	15
VN02 (Controller)	29

6. RESULTS AND ANALYSIS

During system operation in a distributed environment, many measurements were performed for each of the controllers developed previously. As mentioned before, the goal is to analyse system performance under varying network sampling intervals. To this end, for each controller, the sampling period was progressively increased between consecutive measurements.

6.1 System Response (CANoe Output)

Figures 9 and 10 show the resulting closed loop responses for the transfer function $x_{def}/\dot{x}_r$ for each controller under different plant sampling intervals.

Figure 9 illustrates the results when controller 1 is used. As can be seen, the system remains stable when the maximum period between consecutive network samplings is not greater than 32 ms. To reach this result, both scheduling policies had to be used. It is interesting to notice that there is no significant performance degradation for the mentioned sampling interval.

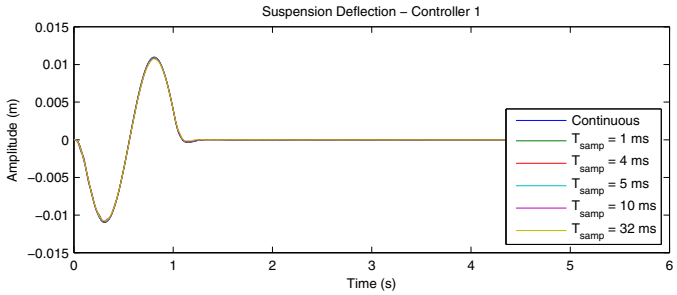

Figure 9. State feedback pole placement approach. CANoe Simulation.

Figure 10 depicts the closed loop response when the system employs controller 2. In this case, the system is able to support a much longer sampling period when compared to the first setup (controller 1). This time, performance degradation can be observed given the difference in amplitude among the curves of different sampling periods. It is also important to mention that, although when applying this controller the system can tolerate longer sampling periods, it does not provide the system with the same closed loop performance as the first one. Another interesting point, is that, in this case, the system did not reach an unstable condition even when network sampling was 160 ms, although this controller was designed to guarantee stability when the sampling interval is at most 10.5 ms.

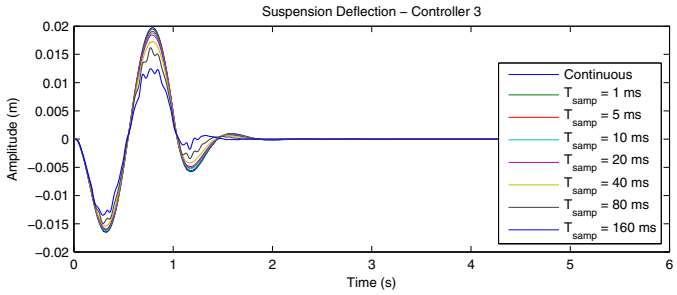

Figure 10. State feedback sampled data approach. CANoe Simulation.

7. CONCLUSIONS

In this work, two different control strategies and two different scheduling policies were developed for the case study of a distributed active suspension system, using the FlexRay protocol as communication backbone. The main goal has been to study the effects that different network sampling periods, would bring to system performance.

Plant and controller were each implemented in separate VN8910 devices with VN8970 plug-in modules.

Considering the time interval $\Delta t_u = t_a - t_k$, where t_k is the plant sampling instant and t_a is the instant when the actuator updates its output, what is clear from the results obtained, is that system stability and performance are very dependent on Δt_u. The slot allocation scheme chosen for each scheduling policy was selected in a way that each device would be able to send its data as soon as it was available in the buffers. With this approach, the period Δt_u was kept as short as possible, since the update buffer task was identified to be the most time consuming one within the application. On the other hand, from a control point of view, it is possible to observe that each controller presents a certain level of tolerance to the interval Δt_u. The pole placement via state-feedback (controller 1) approach was able to maintain system performance even when network sampling was 32 ms. The state feedback using the sampled data approach (controller 3), on the other hand, is very robust against time delays, as can be seen from how much delay it could endure while maintaining the system in a stable condition, even though this characteristic comes with the drawback of performance degradation.

During the experiments, it was possible to observe that the FlexRay protocol was able to guarantee an upper bound on control message period.

ACKNOWLEDGEMENTS

This work was supported by CAPES. We would like to also thank *Vector Informatik GmbH* for their support, by kindly providing both devices VN8910 with plug-in modules VN8970 and also for their technical support during implementation phase.

REFERENCES

P. Antsaklis and J. Baillieul. Special issue on technology of networked control systems. *Proceedings of the IEEE*, 95(1):5–8, January 2007.

M.B.G. Cloosterman, L. Hetel, N. van de Wouw, W.P.M.H. Heemels, J. Daafouz, and H. Nijmeijer. Controller synthesis for networked control systems. *Automatica*, 46(10):1584 – 1594, 2010. ISSN 0005-1098.

A.L. Do, J. M. Gomes da Silva Jr., O. Sename, and L. Dugard. Control design for lpv systems with input saturation and state constraints: An application to a semi-active suspension. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 3416–3421, 2011.

FlexRay Consortium. Flexray communications system - protocol specification - version 3.0.1. `http://www.flexray.com`, October 2010. [Online; last access on 28.08.2012].

J. M. Gomes da Silva Jr. et al. Stabilisation of neutral systems with saturating control inputs. *International Journal of Systems Science*, 42(7):1093–1103, 2011.

Joao P Hespanha, Payam Naghshtabrizi, and Yonggang Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, 2007.

R. Makowitz and C. Temple. Flexray - a communication network for automotive control systems. In *Factory Communication Systems, 2006 IEEE International Workshop on*, pages 207–212, 2006.

B.C. Moore. On the flexibility offered by state feedback in multivariable systems beyond closed loop eigenvalue assignment. *Automatic Control, IEEE Transactions on*, 21(5):689–692, 1976.

Charles Poussot-Vassal. *Commande Robuste LPV Multivariable de Châssis Automobile*. PhD thesis, Grenoble Institut Polytechnique, September 2008.

John Rushby. Bus architectures for safety-critical embedded systems. In *Embedded Software*, pages 306–323. Springer, 2001.

TPJ van der Sande, BLJ Gysen, IJM Besselink, JJH Paulides, EA Lomonova, and H Nijmeijer. Robust control of an electromagnetic active suspension system: Simulations and measurements. *Mechatronics*, 2012.

Vector Informatik GmbH. Solutions for flexray networking. `http://vector.com/vi_flexray_solutions_en.html`, July 2013.

Lixian Zhang, Huijun Gao, and O. Kaynak. Network-induced constraints in networked control systems - a survey. *Industrial Informatics, IEEE Transactions on*, 9(1):403–416, February 2013.

Wei Zhang, Michael S Branicky, and Stephen M Phillips. Stability of networked control systems. *Control Systems, IEEE*, 21(1):84–99, 2001.