Real-Time global scheduling strategy for distributed systems

Antonio Menéndez Leonel de Cervantes, Héctor Benítez-Pérez

Departamento de Ingeniería de Sistemas Computacionales y Automatización,

IIMAS, Universidad Nacional Autónoma de México.

Apdo. Postal 20-726. Del. A. Obregón, México D.F. CP. 01000, México.

Fax: ++52 55 5616 01 76, Tel: ++52 55 5622 36 39

Email: amenendezl@uxmcc2.iimas.unam.mx

Abstract: The global scheduling for real-time distributed systems is an open research area; while there are important contributions to multiprocessor scheduling, most of this analysis do not take into account the communications network and hence do not apply to distributed systems. This paper introduces a scheduling strategy to globally schedule real-time distributed systems. The analysis of a Real-Time Distributed System (RTDS) used as a Networked Controlled System (NCS) is presented as a case study to show in a practical manner the feasibility and the benefits of this global scheduling strategy. .

Keywords: real-time, distributed systems, scheduler, scheduling strategy, networked controlled system, global scheduling.

1 Introduction

The design and implementation of Real-Time Distributed Systems (RTDS) has two major trends of choice, on one hand the emphasis is made on the tasks and the processors that are going to execute them, for this option and in the case of single processors there are very well proven scheduling algorithms such as rate monotonic "RM" or earliest deadline first "EDF" (LIU, 2000). When considering multiple real-time resources or components, the hierarchical composition of schedulers has proven to be a prolific research line and has been studied by Regehr and Stankovic (2001), Feng et al. (2002), and Mok et al. (2004), this frameworks for real-time scheduler composition have been constantly developed and produced sounded schedulability analysis for hierarchical real-rime systems, as the ones proposed by Easwaran et al. (2006-1,2) and Chakraborty et al. (2006). Furthermore the schedulability analysis for multiple processor systems has produced global schedulability analysis like the ones presented by Baruah et al., (2008), Guan et al., (2008) or Anand et al., (2008). On the other hand the option is to implement a RTDS based upon a real-time network like: computer area network "CAN", time triggered protocol "TTP" or flexible time triggered "FTT" by Almeida et al. (2004).

The global scheduling for real-time distributed systems is an open research area; while there are important contributions to global multiprocessor scheduling, most of this analysis do not take into account the communications network and hence do not entirely apply to distributed systems. On the counterpart when the emphasis is made on the real-time network, the processor scheduling is practically left aside. While in both cases the system might have a blend of both approaches and actually work quite well most of the times, the suggestion is to have a global strategy that includes both real-time elements as basic components of a RTDS, thus allowing to have a complete real-time distributed system with real-time scheduling algorithms in every processor and a real-time network. This proposed strategy will eventually allow

the design of a global RTDS scheduler and the possibility to have a global point for validating that all time constraints of a RTDS are to be fulfilled. In order to demonstrate the feasibility of such a strategy, a case study consisting of a networked controlled system is presented.

Feng et al. (2002) have stated that an acceptable performance of a Networked Controlled System (NCS) depends on the sampling periods of its individual components. Even though this is generally true, for real-time distributed systems the area of acceptable performance tends to be smaller and not precisely continuous. One of the reasons for this behaviour is that the relationships amongst the periods of the RTDS components (i.e. nodes and network) have a high degree impact on the overall system performance.

It is possible to have a RTDS, in which the Real-Time constraints of the tasks at every node are fulfilled, meaning that each task is scheduled in such a way that meets its deadline. In these cases the scheduling work is considerable, the system is very inflexible and there is no global guarantee whatsoever that the system is going to remain stable through time.

The objective of this paper is to present a scheduling strategy and an analysis of the behaviour and limitations of a Real-Time Distributed System (RTDS) using it. In order to do this a case study of a RTDS acting as a Networked Controlled System (NCS) has been implemented. The case study consists of a "Quanser 2 DOF Helicopter" which is a helicopter model mounted on a fixed base with two propellers that are driven by DC motors. The front propeller controls the elevation of the helicopter nose about the pitch axis and the back propeller controls the side-to-side motions of the helicopter about the yaw axis. The pitch and yaw angles are measured using high-resolution encoders (Quanser, 2006).

The rest of the paper has the following sections. Section 2 introduces the proposed scheduling strategy in section 3 the NCS case study and its implementation as a RTDS using the proposed strategy are presented. The obtained results are in section 4 and finally the conclusion and future line of work are in section 5.

2. Scheduling Strategy

The common RTDS implementation consists of several nodes (processors) and a communications network running each one as an independent real-time component with its own scheduler. This general scheme presents a time coordination problem, or to put it in other words there is no guarantee whatsoever that the distributed system is going to perform as such (i.e. information flowing from one processor to another through the network) within the limits of all its hard real-time deadlines. This can be clarified with the following example: a very simple distributed system consists on three real-time kernel nodes, namely a sensor node, a controller node and an actuator node, linked together by a fixed priority real-time network as "CAN". In the desirable scenario, the sensor node gathers information and is ready to send it at time sd1, by time sd1 plus the transmission time of the sensor's data lets say tsd1, the controller is ready to start the controller task and consequently to send it the actuator at time cd1. The actuator receives the data form the controller at time cd1 plus the network imposed time at time tcd1, hence the actuator task has an offset of tcd1 before it can start the actuator task. This very straightforward example presents the basic complexity of the problem, which can be stated as the difficulty to calculate the relative time offset of a particular task, imposed by all the preceding tasks and network transmissions. Needless to say that as more elements are involved the more complex the scheduling mission results.

If instead of trying to calculate all the time components affecting a particular task, all the related tasks are scheduled to operate at a given time, the problem gets remarkably easier to solve. Using the same three nodes example lets assume that at every node the task that is responsible to transmit the data is scheduled with the same periodicity and the highest priority relative to its respective processor schedule. Also assume, for the sake of simplicity that the three tasks have the same execution time, this turns into three nodes ready to transmit through the communications network at the same time, then all what is left is to let the transmissions to be placed in the desired order (i.e. sensor, controller and actuator).

With this strategy in mind, the proposal consists in defining a common operational period for all the tasks involved in the transmissions from all nodes participating in the distributed system. This common operational period is called "base" period.

Even tough all the nodes can use this "base" period as their operational period, the proposal includes the option of having different periods for each node by means of the "base" period and a dispersion factor. This *dispersion* factor might be added or subtracted from the "base" period at every node, its value depends on the particular implementation case, but as a rule of thumb it is suggested that it remains not larger than 20% of the value of the "base" period.

As part of the scheduling strategy an ordered bandwidth share is allocated to every node by means of assigning a time-window when to transmit, this occurs independently from the (real-time) network protocol used which must be considered solely as the medium access control.

The "base" period also allows evaluating the performance of a RTDS as a NCS, since it becomes very simple to find the under or over sampling periods of the system.

3 **Case study**

The case study consists of two major components, a helicopter physical and mathematical model in subsection 3.1 and a simulated Real-time distributed system implemented in *Matlab* with *TrueTime* tool developed by Cervin et al. (2007) in subsection 3.2.

3.1 Helicopter

While the description of the helicopter is beyond the scope of this paper, a brief description of it is provided in this subsection, for further information about it the original documentation in Quanser (2006) is recommended. The Quanser 2 DOF Helicopter experiment consists of a helicopter model mounted on a fixed base with two propellers that are driven by DC motors. The front propeller controls the elevation of the helicopter nose about the pitch axis and the back propeller controls the side-to-side motions of the helicopter about the yaw axis. The pitch and yaw angles are measured using high-resolution encoders. The two degrees of freedom helicopter pivots about the pitch axis by angle θ and about the yaw axis by angle ψ. The pitch is defined positive when the nose of the helicopter goes up and the yaw is defined positive for a clockwise rotation. The following table (Table 1) lists various lengths, masses and moment of inertias associated with the helicopter model.

Table 1 Helicopter specifications and model parameters.

Variable	**Description**	**Value**	**Unit**

$B_{eq,p}$	Equivalent viscous damping about pitch axis.	0.800	N/V
$B_{eq,y}$	Equivalent viscous damping about yaw axis.	0.318	N/V
$J_{eq,p}$	Total moment of inertia about pitch pivot.	0.0384	$kg \cdot m^2$
$J_{eq,y}$	Total moment of inertia about yaw pivot.	0.0384	$kg \cdot m^2$
K_{pp}	Thrust torque constant acting on pitch axis from pitch motor/propeller.	0.204	N·m/V
K_{py}	Thrust torque constant acting on pitch axis from yaw motor/propeller.	0.0068	N·m/V
K_{yp}	Thrust torque constant acting on yaw axis from pitch motor/propeller.	0.0219	N·m/V
K_{yy}	Thrust torque constant acting on yaw axis from yaw motor/propeller.	0.072	N·m/V
l_{cm}	Center-of-mass length along helicopter body from pitch axis.	0.186	Cm
m_{heli}	Total moving mass of the helicopter	1.3872	kg

A FF+LQR+I (Feed Forward + Linear Quadratic Regulator + Integrator) controller implemented in *Matlab* and *Simulink* (Figure 1) is used in the helicopter model. The FF+LQR+I regulates the pitch axis of the helicopter using an integrator in the feedback loop to improve the steady-state error. It uses a feed-forward and a proportional-integral-velocity (PIV) algorithms to regular the pitch and a PIV to control the yaw angle (Quanser, 2006).

The linear state-space model of the helicopter is given by equation (2), where after linearizing the non-linear equations of motion about the quiescent point (θ_0=0, ψ_0=0, $\dot{\theta}_0$=0, $\dot{\psi}_0$=0) and substituting the state x = [θ, ψ, $\dot{\theta}$, $\dot{\psi}$] is solved for $\dot{x}$.

$$\dot{x} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{B_p}{J_{eq,p} + m_{heli} l_{cm}^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{B_y}{J_{eq,y} + m_{heli} l_{cm}^2} & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{K_{pp}}{J_{eq,p} + m_{heli} l_{cm}^2} & \frac{K_{py}}{J_{eq,p} + m_{heli} l_{cm}^2} \\ \frac{K_{yp}}{J_{eq,y} + m_{heli} l_{cm}^2} & \frac{K_{yy}}{J_{eq,y} + m_{heli} l_{cm}^2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} u \qquad y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} x \quad (1)$$

The FF+LQR+I control that converges $(\theta, \psi, \dot{\theta}, \dot{\psi}) \rightarrow (\theta d, \psi d, 0, 0)$, where θd is the desired pitch angle and ψd is the desired yaw angle, is defined in equation (3), where k_{ff} is the feed-forward control gain set to: 1 V/V:

$$\begin{bmatrix} u_p \\ u_y \end{bmatrix} = \begin{bmatrix} k_{ff} \frac{m_{heli} g l_{cm} \cos\theta_d}{k_{pp}} \\ 0 \end{bmatrix} - \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \end{bmatrix} \begin{bmatrix} \theta - \theta_d \\ \psi - \psi_d \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} - \begin{bmatrix} \int_{t_0}^{t_f} k_{15}(\theta - \theta_d) + \int_{t_o}^{t_f} k_{16}(\psi - \psi_d) \\ \int_{t_0}^{t_f} k_{25}(\theta - \theta_d) + \int_{t_o}^{t_f} k_{26}(\psi - \psi_d) \end{bmatrix} \quad (2)$$

This FF+LQR+I equation uses the integral values of the errors in the feedback loop to improve the steady-state error, with the following control gain according to the *Matlab* model and the Quanser (2006) manual:

$$k = \begin{bmatrix} k_{11} \ldots k_{16} \\ k_{21} \ldots k_{26} \end{bmatrix} = \begin{bmatrix} 18.9 & 1.98 & 7.48 & 1.53 & 7.03 & 0.770 \\ -2.22 & 19.4 & -0.450 & 11.9 & -0.770 & 7.03 \end{bmatrix}$$

3.2 RTDS

The RTDS implementation consists of eight nodes, being everyone an independent Real-time kernel. Four of these nodes are sensor nodes, 2 nodes are actuators, one is the controller node and the eighth node is the scheduler node. As it can be seen in Figure 2 the eight nodes that build the RTDS communicate between them through a Real-time "CAN" network. The model is implemented in *Matlab* using the tool *TrueTime* from Lund University by Cervin et al. (2007) to simulate the nodes with a real-time kernel and the real-time network. The outputs of the system can be connected to both, the simulated helicopter model or the actual helicopter, being therefore capable of calculating or actually "seeing" the RTDS performance.

The "base" period is the operational period for the controller periodic task in the corresponding controller node, this "base" period and the *dispersion* factor are used to calculate the actual sampling periods of the four sensor nodes according to the following equations:

$$\begin{aligned} sensor1 &= base \times (1 + dispersion) \\ sensor2 &= base \times (1 - dispersion) \\ sensor3 &= base \times (1 + (dispersion \times 1.1)) \\ sensor4 &= base \times (1 - (dispersion \times 1.1)) \end{aligned} \qquad (3)$$

where *sensorN* = period for sensor N, *base* is the "base" period and $0 \leqslant dispersion \leqslant 0.2$ is the dispersion factor.

The value of the dispersion factor means that in the tightest case, when *dispersion*=0, the four sensors have the same period (i.e. "base" times 1 + 0) as the controller. On the other extreme, when "base"=0.020, the sensors have a period 20% or 22% above or bellow the "base" period. Dispersion greater than 25% causes the system to have an unacceptable performance.

The first node in the model (leftmost node in Figure 2) is the controller node, which uses the values provided by the sensors and equation (2) to calculate the control outputs (u_p and u_y) that correspond to the respective "pitch" and "yaw" motor voltages. The four sensor nodes are correspondingly: theta, psi, theta-dot and psi-dot from equation (2). The two actuators (2 lower right nodes in Figure 2) have the control outputs u_p and u_y been the "pitch" and "yaw" motor voltages. The last node is the scheduler node (rightmost node in Figure 2), which as it name implies, schedules the RTDS activities. It is also responsible, as part of the scheduling strategy, to periodically assign network bandwidth the to the nodes and it defines and communicates the other nodes a common operational period named "base" period.

In order to have a performance metric of the system, an error of the RTDS is calculated. The RTDS error is the difference between the obtained or real angle θ and desired angle θd (Figure 3), this error is added during the whole simulation as expressed by equation (4):

$$quadratic\ error = \sum \frac{(\theta_t - \theta d_t)^2}{2} \qquad (4)$$

The logarithmic chart in Figure 4 shows the *quadratic error* graphed against the "base" period. The dotted horizontal line marks the level of acceptable and unacceptable system performance. As it can be seen in this chart the function appears not to have a continuous slope as it is expected from a NCS, nevertheless the points of over sampling and under sampling periods (0.001 and 0.027 seconds respectively) can be located.

The next consideration is the bandwidth allocated to each node of the RTDS. When all the nodes are competing to obtain the network and get bandwidth according to the network access control algorithm (i.e. *CAN* network priority protocol), the system trends to an unstable state as it can be inferred from Figure 5, in which the error increases through time (i.e. The difference amongst real angle θ and desired angle θd). In this case study, the scheduler node avoids this problem by assigning each node a time window when to transmit.

4 Results

A series of tests were conducted, having the following characteristics: the "base" period runs above 0.001 and below 0.027 seconds (the points of over and under sampling in Figure 4) with increments of 0.001 seconds. The dispersion factor runs from 0.0 to 0.2 (i.e. 0% to 20% of "base") with increments of 0.05. The total time to run each test is 50 seconds, which is time enough to determine an unacceptable performance.

The chart in Figure 6 shows the total number of transmissions (i.e. all the transmissions made by all the nodes during the test) graphed against the different "base" periods and percentages of dispersion. It can be seen that the total number of transmissions depends as it was expected on the "base" sampling period, being evenly distributed amongst the dispersion factors.

As the total network activity, the network activity per node (i.e. number of transmissions per node) depends basically on the "base" period. As can be seen in Figure 7, the number of transmissions is inversely proportional to the "base" period, since the dispersion factor practically does not have an impact in the network activity per node, it is neglected in the chart.

The next graph (Figure 8) shows the accumulated quadratic error of several simulations, of 25 seconds each, in which the base period goes from 0.002 to 0.026 seconds, showing that the acceptable performance of the RTDS depends not only on the sampling rate, but also in the time dependencies along its components.

In the Quadratic error graph (Figure 8) some periods of operation reveal a higher than acceptable error, for instance when the "base" period is 15 milliseconds and the dispersion factor is 0 the accumulated error for 25 seconds is 80. The Quadratic error graph (Figure 8) and the "System Performance" chart (Figure 4) show that the acceptable performance and the sampling rate of a RTDS do not have a linear relation.

As it can be seen in Figure 8, the quadratic error does not present a smooth slope curve, as it is expected form a NCS, in a RTDS the boundaries for an acceptable performance not necessarily guarantee it in all the intermediate points,

proving that a stable RTDS depends not only on the individual sampling period, but also on the relations of the sampling periods of the nodes conforming the RTDS.

In Figure 9 the behavior of the system using a "base" period of 0.015 seconds can be appreciated, the system is unstable after approximately 75 seconds.

The RTDS scheduler node allocates a bandwidth share to every node by means of assigning a time-window when to transmit, taken as an heuristic solution, the controller gets 1/3 of the bandwidth, while the rest (2/3) is shared equally amongst the sensors. The Network utilization graph (Figure 10) clearly shows the network is not the constraining resource of the RTDS, since it still has spare capacity to provide. Again emphasis should be made in the time relations among the systems components.

The graph in Figure 10 shows five horizontal lines which represent the controller (node 1) and the four sensors (nodes 2 to 5) transmissions during a one second sample. Each one of the five lines has three possible states: 0, 0.25 and 0.5, meaning respectively: not transmitting, waiting for the network and transmitting (Figure 11). It is easy to appreciate in the Network utilization graph (Figure 10) that the network throughput is not the restraining component on this RTDS, since most of the time the nodes are not transmitting (i.e. the node state is 0).

In the extreme over-sampling case, the network is used 100% of the time, and in several occasions there are nodes waiting to transmit (Figure 12). This lack of network availability results in deadline losses at all the nodes, causing the information that the sensors or the controller send, if any, to be obsolete. So as the system has to operate with practically no data, it reaches instability almost immediately.

5 **Conclusions**

The global scheduling strategy for real-time distributed systems proposed here, helps to significantly reduce the complexity of scheduling a system of this characteristics. Further research must be done following this idea, in order to be able to demonstrate that the postulate made by Liu (2000) declaring that 'Hard Real-Time Distributed Systems do not have robust and accurate schedulability analysis algorithms to be validated as dynamic multiprocessor systems' is close be proven wrong.

This scheduling strategy also allows evaluating a RTDS performance as a NCS by changing one single (base) parameter for the whole system, it also serves to increase the acceptable performance sampling range of a NCS.

6 References

Almeida L., Pedreiras P., (2004), "Scheduling within Temporal Partitions: Response-time Analysis and Server Design", EMSOFT'04, September 27--29, ACM 1-58113-860-1/04/0009.

Anand M., Easwaran A., Fischmeister S., Lee I., (2008), "Compositional Feasibility Analysis of Conditional Real-Time Task Models", 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing (ISORC), IEEE, pp.391-398.

Baruah S:, Baker T. (2008), "Global EDF schedulability analysis of arbitrary sporadic task systems", Euromicro Conference on Real-Time Systems, IEEE, pp.3-12.

Cervin A., Ohlin M., Henriksson D., (2007), “True-Time 1.5 Reference Manual”, Dep. Of Atomatic Control, Lund University, Sweden.

Chakraborty S., Priya S. (2006), “A Framework for Compositional and Hierarchical Real-Time Scheduling”, Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06).

Easwaran A., Shin I., Sokolsky O., Lee I., (2006-1), ”Associative Composition of Hierarchical Real-Time Systems, Technical Report: MS-CIS-06-06”, University of Pennsylvania, Philadelphia, PA.

Easwaran A., Shin I., Sokolsky O., Lee I., (2006-2), “Incremental Schedulability Analysis of Hierarchical Real-Time Components”, Proceedings of the 6th Annual ACM Conference on Embedded Software (EMSOFT), October, pp.272-281.

Feng-Li Lian, James Moyne, and Dawn Tilbury (2002), “Network Design Consideration for Distributed Control Systems”, IEEE Transactions on Control Systems Technology, Vol. 10, No. 2, March.

Feng M., Mok A., (2002) “A model of hierarchical realtime virtual resources”, Proc. of IEEE Real-TimeSystems Symposium, pages 26–35, December.

Guan N., Gu Z., Lv M., Deng Q., and Yu G., (2008), “Schedulability Analysis of Global Fixed-Priority or EDF Multiprocessor Scheduling with Symbolic Model-Checking”, 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing (ISORC), pp.556-560.

Liu W.S. J, (2000), “ Real-Time Systems”, Prentice Hall, USA.

Mok A., Wang W. (2004), “On the Composition of Real-Time Schedulers”, RTCSA 2003, Springer-Verlag Berlin Heidelberg, LNCS 2968, pp. 18–37.

Quanser Inc. (2006), “Quanser 2 DOD Helicopter User and Control Manual”, rev 1.0, February 10.

Regehr J., Stankovic J. (2001), “HLS: A framework for composing soft real-time schedulers”, Proc. of IEEE Real-Time Systems Symposium, December, pp.3–14.

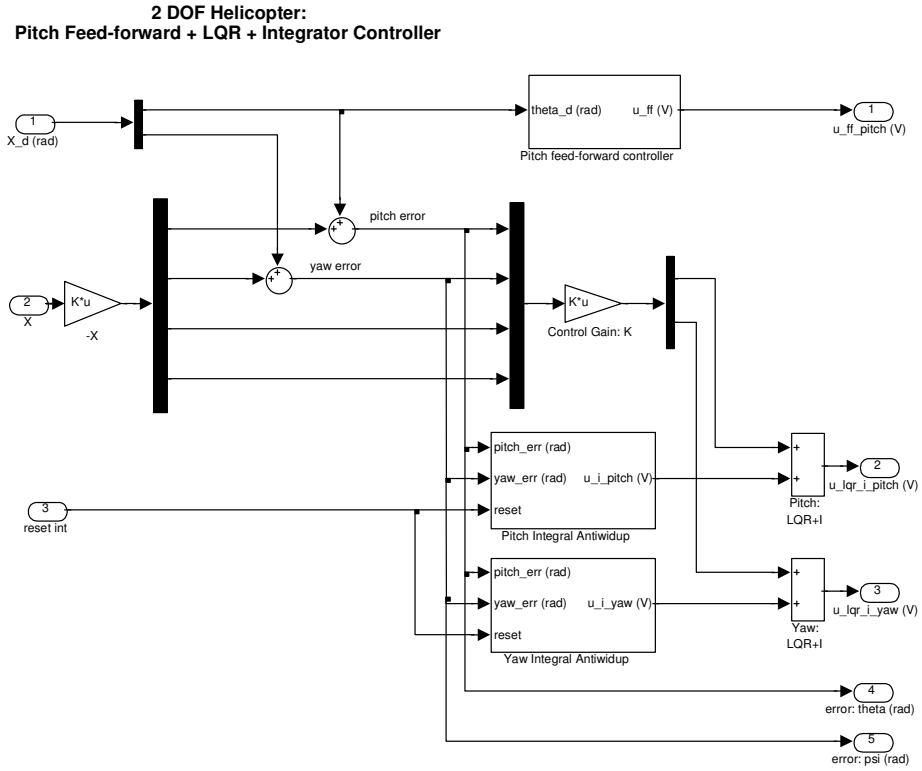

Figure 1 Helicopter Close-Loop system simulation model Quanser (2006).

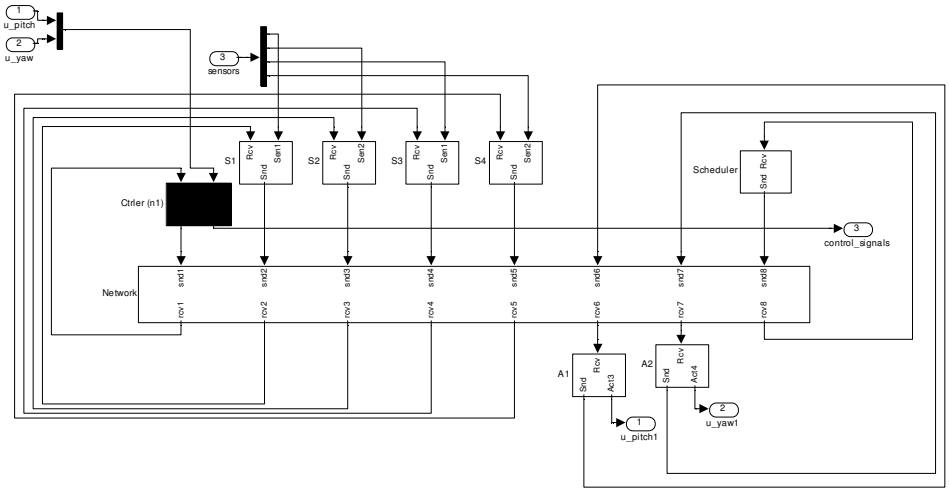

Figure 2 RTDS Case study implementation.

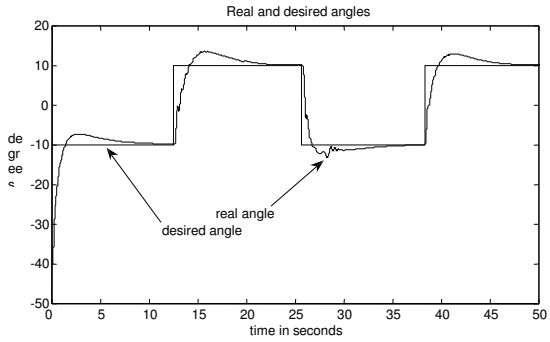

Figure 3 Real angle θ and desired angle θd.

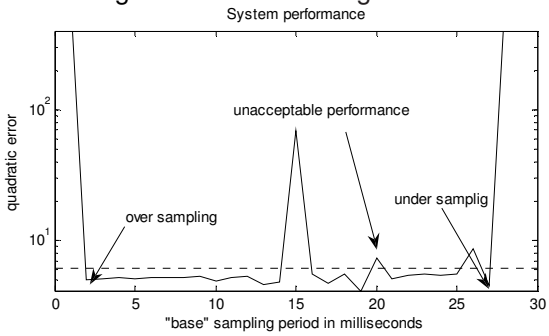

Figure 4 System performance. Sampling period (in 10^{-3} seconds) against logarithmic quadratic error.

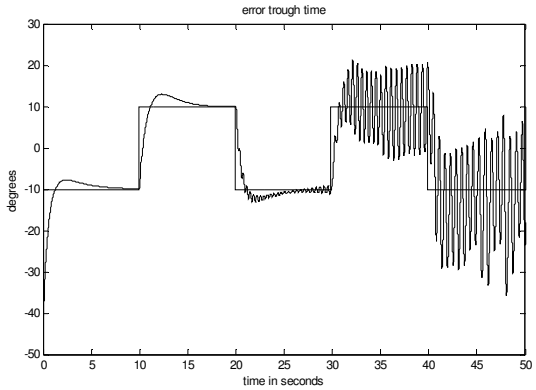

Figure 5 Error through time (Difference amongst real angle θ and desired angle θd).

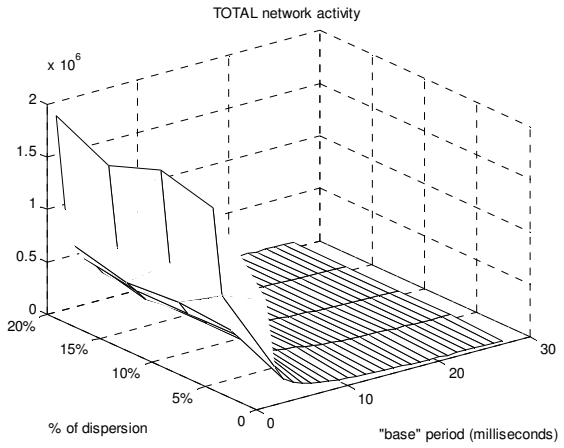

Figure 6 Total number of transmissions varying the dispersion and the “base” period.

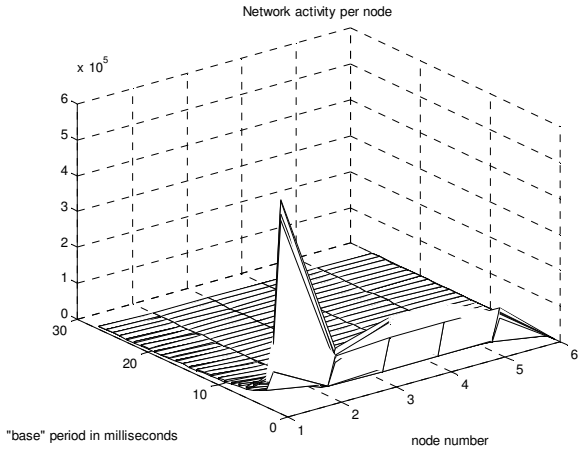

Figure 7 Relation between the total number transmissions per node and the “base” period.

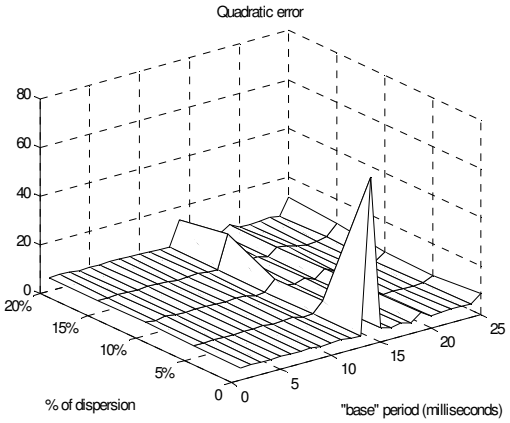

Figure 8 Quadratic error

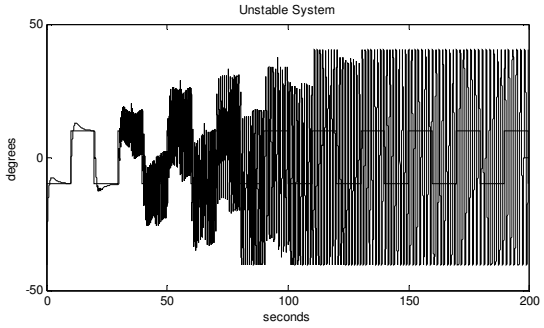

Figure 9 Unstable system with a “base” period of 0.015 secs.

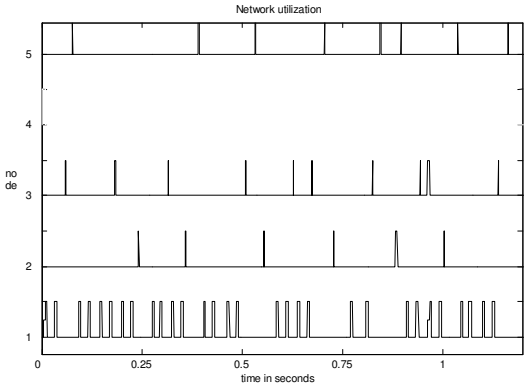

Figure 10 Network utilization

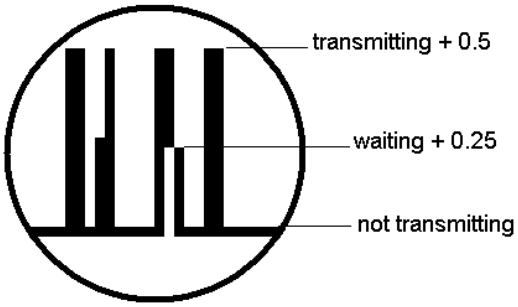

Figure 11 Network utilization zoom, the three states of a node.

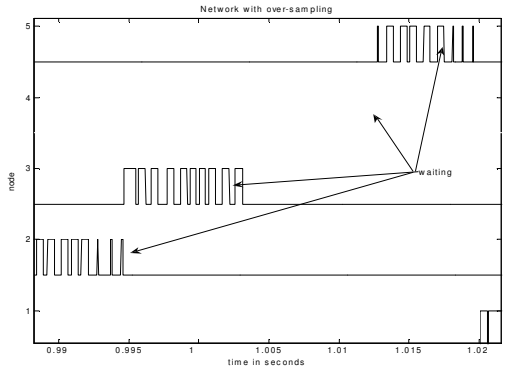

Figure 12 Network utilization with over-sampling