Influence of the Sampling Period in the Performance of a Real-time Distributed System under Jitter Conditions

ANA ANTUNES[1], FERNANDO MORGADO DIAS[1], ALEXANDRE MANUEL MOTA[2]
[1]Escola Superior de Tecnologia de Setúbal do Instituto Politécnico de Setúbal
Campus do IPS, Estefanilha, 2914-508 Setúbal
[2] Departamento de Electrónica e Telecomunicações
Universidade de Aveiro, 3810 Aveiro
PORTUGAL

Abstract: - In this paper an analysis is made of the influence of the sampling period in the control performance of a pole-placement adaptive controller for a real-time distributed control system under output jitter conditions. Results show that the lower the sampling frequency the better control performance is obtained for the same jitter conditions.

Key-Words: - jitter, real-time systems, distributed control, adaptive controller, control performance.

1 Introduction

Real-time distributed control systems are nowadays widely used in embedded applications. Distributed control architectures induce jitter, either in the sampling period (called input jitter, sampling jitter or read-in jitter) or in the actuation moment (called sampling to actuation jitter, output jitter or read-out jitter) eventually leading to performance degradation [1],[2],[3],[4],[5]. In this paper an evaluation is made of the control performance of a real-time distributed system with a pole-placement controller under several jitter conditions for different sampling period values.

2 The real-time distributed system

The block diagram of the real-time distributed control system is shown in figure 1. The system has three nodes: the sensor node, the controller node and the actuator node. The nodes are connected using the CAN bus.

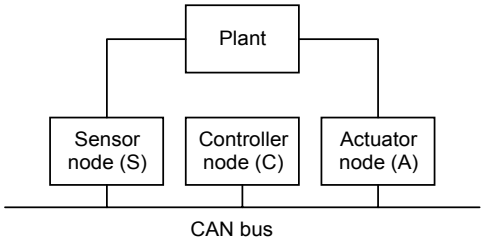

Fig.1- Block diagram of the distributed control system.

The communication scheme is shown in figure 2. The sensor node sends a message (M1) with the sampled value of the output of the plant to the controller node that computes the actuation value for the next sample and sends that value and the actuation order (M2) to the actuation node.

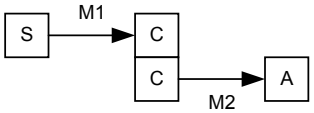

Fig.2- Communication model scheme for the system.

The system was simulated using TrueTime a MATLAB/Simulink based simulator for real-time control systems [6],[7],[8]. The simulator facilitates co-simulation of controller task execution in real-time kernels, network transmissions and continuous plant dynamics. Each node of the system was implemented using a TrueTime Kernel block and the CAN bus was implemented with a TrueTime Network block [9].

The plant models a cruise control system and its transfer function is given by equation (1).

$$\frac{Y(s)}{U(s)} = \frac{0.05}{s + 0.05} \quad (1)$$

This model was taken from [10].

3 The adaptive controller

The system controller used was an adaptive controller [11]. The adaptive controller has two loops. The inner loop includes the variable dynamics controller and the process. The outer loop is composed by the recursive process parameter estimator and a design calculator block and is responsible for the adjustment of the parameters of the controller. The controller was implemented in MATLAB inside the TrueTime Kernel of the controller node.

3.1 System identification

The generic form of the discrete time transfer function is given by equation (2).

$$G(q^{-1}) = z^{-K}\frac{B(q^{-1})}{A(q^{-1})} \quad (2)$$

where A and B are given by equations (3) and (4) and K is the discrete dead time.

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_n q^{-n} \quad (3)$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \ldots + b_m z^{-m} \quad (4)$$

The model for the SISO system is given by equations (5) and (6)

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \quad (5)$$

$$y(t) = x(t) \quad (6)$$

The discrete model can be represented by equation (7).

$$y(kh + h) = \Phi y(kh) + \Gamma u(kh) \quad (7)$$

where

$$\Phi = e^{Ah} \quad (8)$$

$$\Gamma = \int_0^h e^{As} ds B \quad (9)$$

The discrete transfer function is given by equation (10).

$$G(q) = \begin{bmatrix} 1 & 0 \end{bmatrix}(qI - \Phi)^{-1}(\Gamma q^{-1}) \quad (10)$$

This leads to equation (11).

$$G_1(q) = \frac{bq^{-1}}{1 - aq^{-1}} \quad (11)$$

The discrete function for the model was obtained using the parametric-type model ARX [12]. System parameters were estimated using the least squares criterion and a recursive implementation for this method was adopted to run online during the simulation. The sampling period was chosen to be 2.75s, 1.925s and 1.1s as will be explained.

The regressors are given by equation (12).

$$y_1(k) = \begin{bmatrix} y_1(k-1) & u_1(k-1) \end{bmatrix} \quad (12)$$

3.2 The control function

The control function uses the pole-placement technique. The closed loop behaviour was determined by choosing adequate values for the pole of the closed loop system. An observer polynomial, with faster dynamics, was also chosen. The closed loop pole was chosen as α_m=0.2 and the observer pole as α_0=0.4. The parameters of the control function were obtained by directly solving the resultant Diophantine's equation. The resulting equation is given in equation (13).

$$u(k) = t(r_{ef}(k) - a_{obs} r_{ef}(k-1)) - s_0 y(k) - s_1 y(k-1) \ldots + u(k-1) \quad (13)$$

4 Test description

The choice of the sampling period was made following the rule of thumb presented by [13] according to what, for first order systems, the number of samples per rise time (N_r) should be chosen between 4 and 10. The chosen values were 2.75s and 1.1s corresponding to N_r= 4 and N_r=10, respectively and a middle point of 1.925s.

Several tests were made with different output jitter conditions. In all the tests the sampling jitter is equal to zero. The output jitter was introduced by the change in the value of the parameter exectime of the controller task according with the task models defined by the TrueTime simulator [9].

A first test was made with exectime = 0. The output jitter measured for that situation equals $7.56*10^{-4}$s and is only due to the system architecture (processing delays inside the nodes and network access delays).

The output jitter introduced is of two kinds: constant or variable following a pre-defined sequence. In the constant case the values used correspond to 25%, 50% and 75% of the sampling period. The jitter sequences are of two kinds: random and modified gamma. Two random sequences were generated using the MATLAB rand function, distributed over 75% and 100% of the sampling period. The modified gamma sequences were generated using MATLAB gamrnd function and were then modified to allow jitter to be concentrated in the upper half of the sampling period. The histograms of the jitter sequences for h=1.925s, are presented in figures 3 to 6.

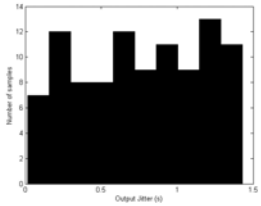

Fig.3- Output jitter distribution for sequence rand75.

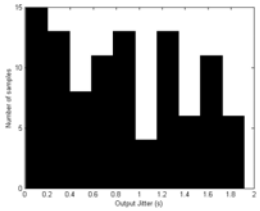

Fig.4- Output jitter distribution for sequence rand100.

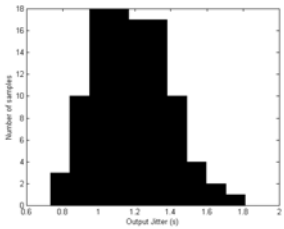

Fig.5- Output jitter distribution for sequence gam1.

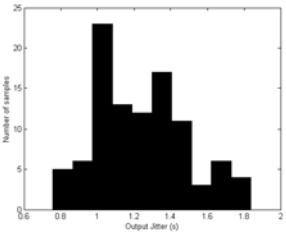

Fig.6- Output jitter distribution for sequence gam2.

The sequences for the other values of h were obtained in a similar way.
In all the tests a perturbation was introduced in the system through the control signal.

5 Test Results

The results for the control performance are presented in table 1 that reports the quadratic mean square error obtained comparing the reference signal with the output signal of the system.

Test number	Exectime	MSE
1	0	0.1867
2	25%h	0.1277
3	50%h	0.1755
4	75%h	46.5918
5	rand75	0.1961
6	rand100	0.2250
7	gam1	18.2614
8	gam2	24.7575

Table 1- Mean square error obtained for h=2.75s between samples 25 and 80.

Test number	Exectime	MSE
1	0	0.1509
2	25%h	0.1514
3	50%h	0.1490
4	75%h	8.3867
5	rand75	0.1439
6	rand100	0.1228
7	gam1	0.1794
8	gam2	0.2576

Table 2- Mean square error obtained for h=1.925s between samples 20 and 120.

Test number	Exectime	MSE
1	0	0.1418
2	25%h	0.1179
3	50%h	0.1159
4	75%h	0.0834
5	rand75	0.1338
6	rand100	0.1021
7	gam1	0.1124
8	gam2	0.0909

Table 3- Mean square error obtained for h=1.1s between samples 40 and 200.

Figures 7 to 21 present the control and error signals for tests 2, 3, 4, 6 and 8 for the different sampling periods. Tests 2, 3 and 4 were obtained under constant jitter of 25%, 50% and 75% of h, respectively. Tests 6 and 8 were obtained under variable jitter with the sequences obtained as stated before. These test were chosen because they present the worst jitter conditions.

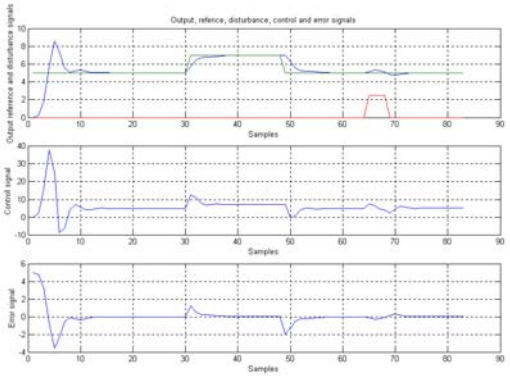

Fig.7- Signals for test 2 with h=2.75s.

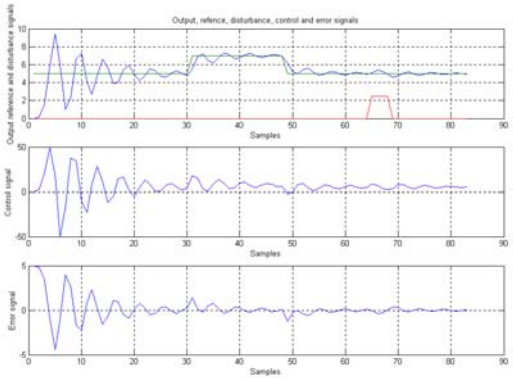

Fig.8- Signals for test 3 with h=2.75s.

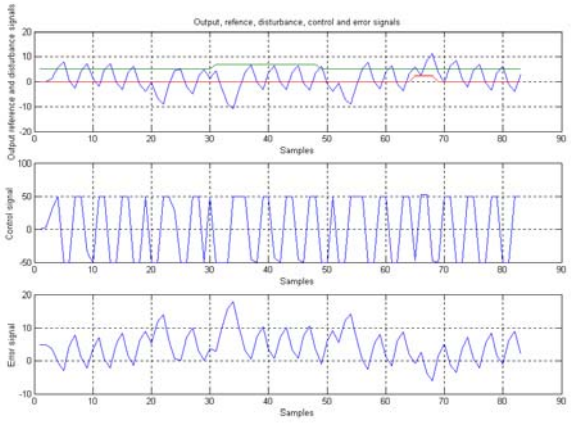

Fig.9- Signals for test 4 with h=2.75s.

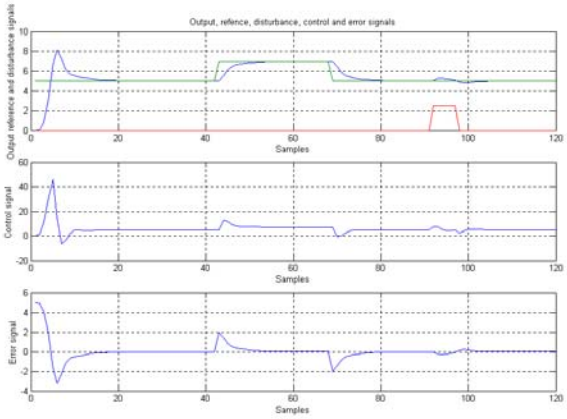

Fig.12- Signals for test 2 with h=1.925s.

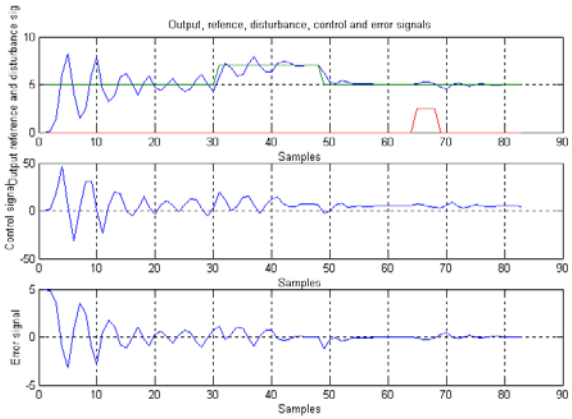

Fig.10- Signals for test 6 with h=2.75s.

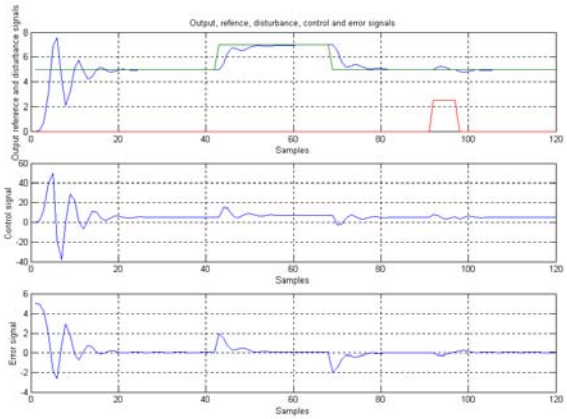

Fig.13- Signals for test 3 with h=1.925s.

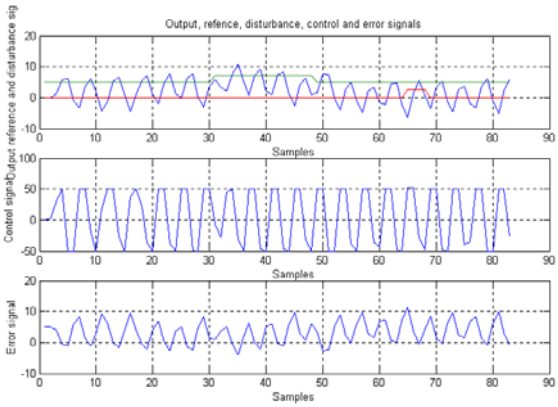

Fig.11- Signals for test 8 with h=2.75s.

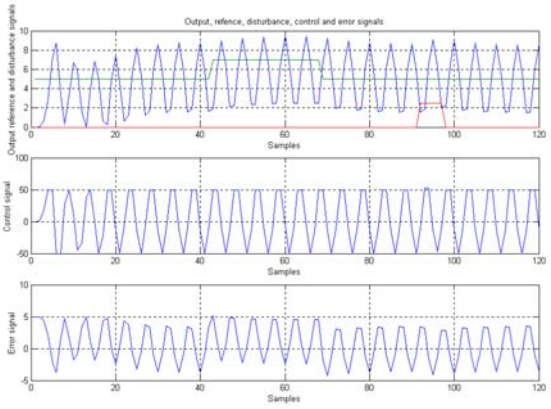

Fig.14- Signals for test 4 with h=1.925s.

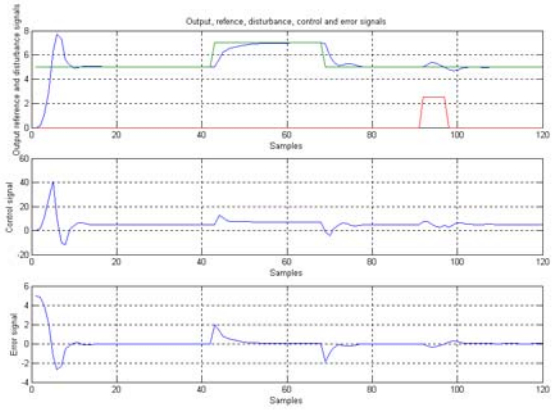

Fig.15- Signals for test 6 with h=1.925s.

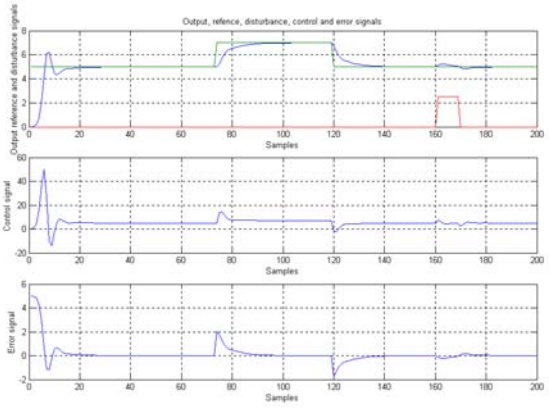

Fig.18- Signals for test 3 with h=1.1s.

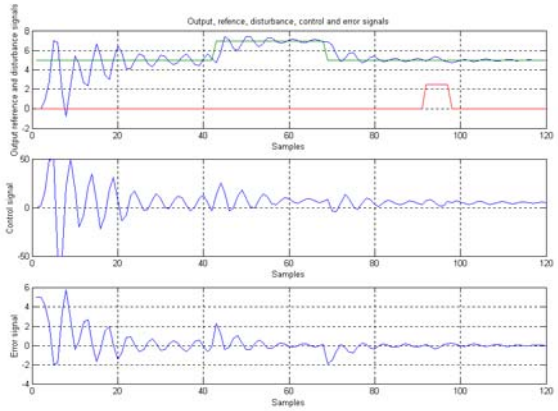

Fig.16- Signals for test 8 with h=1.925s.

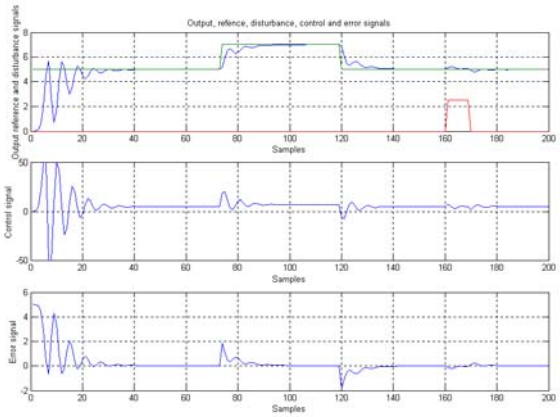

Fig.19- Signals for test 4 with h=1.1s.

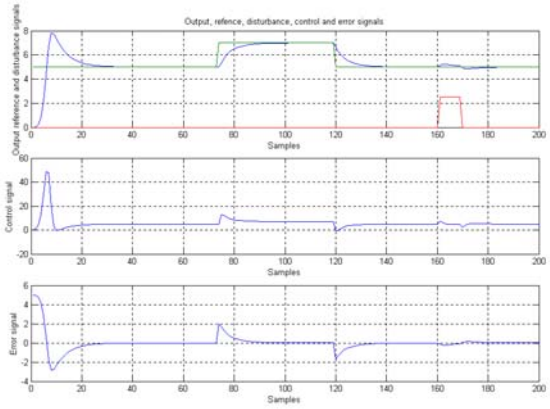

Fig.17- Signals for test 2 with h=1.1s.

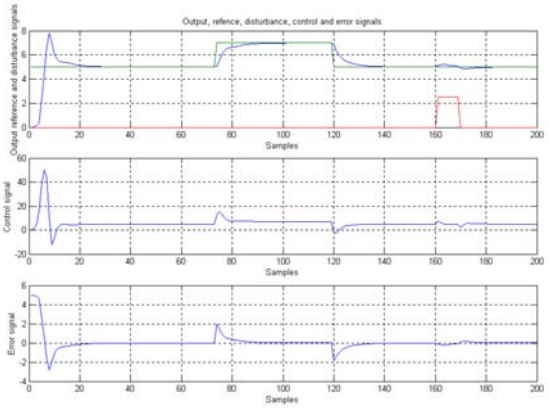

Fig.20- Signals for test 6 with h=1.1s.

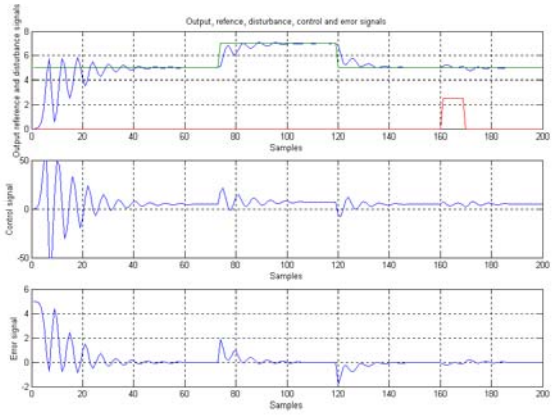

Fig.21- Signals for test 8 with h=1.1s.

As can be seen from figures 7 to 11, for h=2.75s, corresponding to N_r= 4, the system oscillates for tests 3 and 4 and for tests 6 and 8 the oscillation is so heavy that the system is not able to follow the reference signal. Test 2 is the only one where the system is controlled without oscillation. For h=1.925s, corresponding to $N_r \approx 6$, the system oscillates for test 8, test 4 presents heavy oscillation and for tests 2, 3 and 6 the system doesn't presents oscillation. Finally for h=1.1s, corresponding to N_r=10 the system oscillates slightly for test 8 and doesn't oscillates for the other situations.

Output jitter distributed over the upper half of the sampling period affects control performance more then randomly distributed jitter. Random jitter affects the system in the same way as a constant jitter with the average value of the sequence [14] so the test 6 usually presents better results than test 4.

It can clearly be seen that as the sampling period grows the system begins to oscillate first slightly then heavily.

4 Conclusion

A real-time distributed system was tested under different jitter conditions.

Comparison between the control performance of the tests made for different sampling periods shows that as the sampling period grows the system begins to oscillate slightly then heavily becoming unstable.

Results recommend that the sampling period should be chosen as small as possible taking into account the particular characteristics of the systems and its implementation.

In future work, another identification approach, using a different model to account for the jitter effect would be tested and 2nd order systems would be investigated as well.

References:

[1] A. Antunes, F. M. Dias, A. M. Mota, CAN-based real time adaptive distributed control, *Proceedings of the 8th International CAN Conference*, Las Vegas, USA, 2002.

[2] A. Stothert, I.M. MacLeod, Effect of Timing Jitter on Distributed Computer Control System Performance, *Proceedings of the 15th IFAC Workshop DDC'98- Distributed Computer Control Systems*, 1998.

[3] M. Sanfridson, Timing problems in real-time computer control systems, *Technical report*, Department of Royal Institute of Technology, 2000.

[4] P. M. Colom, Analysis and design of real-time control systems with varying control timing constraints, *PhD. Thesis*, Universitat Politècnica de Catalunya, Spain, 2002.

[5] K. G. Shin, X. Cui, Computing time delay and its effects on real-time control systems, *IEEE Transactions on control systems technology*, Vol.3,No2, 1995,pp. 218-224.

[6] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, K.-E. Årzén, How does control timing affect performance?, *IEEE Control Systems Magazine*, Vol.23, No2, 2003, pp. 16-30.

[7] A. Cervin, D. Henriksson, B. Lincoln, K.-E. Årzén, Jitterbug and Truetime: analysis tools for real-time control systems, *Proceedings of the 2nd Workshop on Real-Time Tools*", Copenhagen, Denmark, 2002.

[8] D. Henriksson, A. Cervin, K.-E. Årzén, TrueTime: Simulation of control loops under shared computer resources, *Proceedings of the 15th IFAC World Congress on Automatic Control*, Barcelona, Spain, 2002.

[9] D. Henriksson, A. Cervin, TrueTime 1.1 – Reference Manual, *Technical Report*, Department of Automatic Control, Lund Institute of Technology, Sweden, 2003.

[10]http://www.engin.umich.edu/group/ctm/index.html, *Control Tutorials for Matlab*, University of Michigan.

[11] K. J. Åström, B. Wittenmark, *Adaptive Contro*", Addison-Wesley, 2nd edition, 1995.

[12] L. Ljung, System Identification – Theory for the user, Prentice Hall, 1987.

[13] K. J. Åström, B. Wittenmark, *Computer Controlled Systems*, Prentice Hall, 3nd edition, 1997.

[14] B. Lincoln, A simple stability critetion for control systems with varying time delays, *Proceedings of the 15th IFAC World Congress*, 2002.