The Effects of Initial Offset and Clock Drift Errors on Clock Synchronization of Networked Control Systems

Eloy M. Oliveira Junior[1], **Marcelo L. O. Souza**[1]

[1]Space Mechanics and Control Division (DMC)
National Institute for Space Research (INPE) – São José dos Campos, SP – Brazil

eloy@dem.inpe.br, marcelo@dem.inpe.br

Abstract*. Control systems of satellites, aircrafts, automobiles, traffic controls, etc., are becoming increasingly complex and/or highly integrated due to their use of computers networked via communication devices and protocols working in real time. In these systems, the time requirements should be followed strictly, with great precision and synchronization; otherwise the controls degrade until instability. This paper analyzes the effects of initial offset and clock drifts errors on clock synchronization of Networked Control Systems (NCS). To do so, it simulates a typical NCS in the TrueTime/Matlab/ Simulink environment using a FTM correction algorithm and a TDMA communication protocol. The preliminary results suggests that: 1) the TDMA protocol is more susceptible to errors in the initial offset than to errors in the clock drift; 2) the FTM algorithm corrects the clock drift error better than the initial offset error; 3) In a NCS with a TDMA protocol, a fault in the time management can turn the control laws temporarily unstable.*

1. Introduction

Control systems of satellites, aircrafts, automobiles, traffic controls, etc., are becoming increasingly complex and/or highly integrated as defined by the SAE-ARP-4754 Standard. Such systems use, among other key technologies, computers networked via communications devices and protocols working in real time to form Networked Control Systems (NCS). In these systems, the time requirements should be followed strictly, with great precision and synchronization; otherwise the controls degrade until instability. This creates the need to work with high-precision clocks corrected by periodical algorithms to achieve a good time management.

This paper analyzes the effects of initial offset and clock drifts errors on clock synchronization of Networked Control Systems (NCS). To do so, it simulates a typical NCS in the TrueTime/Matlab/Simulink environment using a Fault-Tolerant Mid-Point (FTM) correction algorithm and a Time Division Multiple Access (TDMA) communication protocol. In the first simulation, one of the nodes of a NCS is with an initial offset error in its clock. The initial offset error generates an initial delay that affects the clock synchronization with the FTM algorithm, and then, the communication and control. In the second simulation, one of the nodes of a NCS is with a drift error in its clock. The drift error generates a delay that affects the computing, communication and control. The objective is to analyze: 1) how the TDMA protocol is affected by such errors; 2) the efficiency of the FTM algorithm in correcting such errors; 3) how the NCS is affected by such errors.

The FTM algorithm is fault tolerant: each node reads the value of the clock of the other nodes in the network and estimates the drift of the clocks by a convergence function. In this paper, two control loops sharing the same databus in a network were simulated. Each control loop has a sensor, a controller and an actuator/plant. The nodes of the control loops use the FTM algorithm to synchronize the clocks of the nodes on the network. We used a TDMA communication protocol. The plant used is an electrical/hydraulic actuator of second order controlled by a Proportional, Integrative, and Derivative (PID) controller. We simulated it using the TrueTime toolbox, based on Matlab/Simulink, and we synchronized the nodes using the FTM algorithm.

2. Clock Model

There are many models to represent a physical clock: for example, Varnum (1983) proposed a simple stochastic model of a physical clock. In this paper, we used the geometric model of Figure 1, where the clocks are represented by straight lines. In this model the effects of fluctuation (jitter) will be discarded. More information about this model can be found in Oliveira Junior (2010).

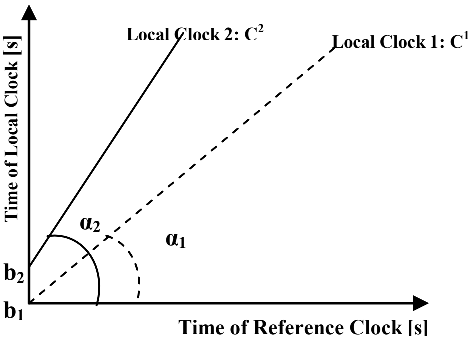

Figure 1. Geometric Clock Model.

In Figure 1, two lines are drawn, one representing the local clock 2, C^2, and other representing the local clock 1, C^1. Clock 2 has an initial offset and drift in relation to clock 1. The horizontal axis represents the time of the reference clock used to measure the time of both clocks. The vertical axis represents the time of local clocks. The clock model of Figure 1 is expressed by:

$$\Delta C_k(t) = C_k^2 - C_k^1 = (a_k^2 - a_k^1)(t_k^R) + (b_k^2 - b_k^1) = \Delta a_k t_k^R + \Delta b_k \qquad (1)$$

In Equation 1, C^1 represents clock 1 and C^2 represents clock 2. The coefficient 'b' is the initial offset and the coefficient 'a' represents the inclination of a straight line. Deriving Equation 1 with respect to time, we have:

$$\Delta \dot{C}_k(t) = \Delta a_k \qquad (2)$$

Equation 2 shows the rate of change of the difference between straight lines 2 and 1; we can conclude that Δa is the drift of clock 2 with respect to clock 1. With the discrete geometric model of the clock, we will set the model measures. This is given by:

$$\Delta a_k^{2,1} = \frac{b_k^2 - b_{k-1}^2}{b_k^1 - b_{k-1}^1} \qquad (3)$$

$$\Delta b_k = b_k^2 - b_k^1 \tag{4}$$

$$t_k^R = kg^R \tag{5}$$

$$y_k = \Delta b_k + \Delta a_k^{2,1} t_k^R \tag{6}$$

Where Equation 3 represents the measurement of drift of clock 2 with respect to clock 1, Equation 4 represents the measurement of the instantaneous offset, Equation 5 represents the measurement of the reference clock where the g^r is the resolution of such clock, and Equation 6 represents the model of discrete measures.

3. FTM Algorithm

The FTM (Fault-Tolerant Mid-Point) algorithm, also known as Welch-Lynch algorithm provides fault tolerance for Byzantine clock synchronization of distributed systems (Lundelius & Lynch, 1984). To ensure that all nodes have a consistent view of time, we need to re-synchronize the clocks regularly (periodically). For this the algorithm follows a logical sequence shown in the flowchart of Figure 2. Each node applies this sequence with the objective of reaching a correction term. With this correction term, the deviations caused by the drift of the clocks are adjusted so that all system clocks are within a certain precision. In such flowchart:

Number 1 in Figure 2 indicates a loop condition. This condition means that if the local clock time of the node is equal to the time of re-synchronization, then synchronization has to start somewhere. R_{int} is the predetermined period of re-synchronization and k is its instant. Number 2 in Figure 2 indicates where one has to read data of the databus, meaning that the local clocks exchange information among themselves, that is, all clocks send a broadcast with the timestamp of its own clock. Number 3 in Figure 2 indicates the ordering of data. Each node sees only its row of the matrix with the timestamps values forming a vector of values A. This vector is sorted in ascending order. At Number 4 of Figure 2 the Welch-Lynch algorithm calculates the convergence function, after ordering the data in ascending order. It discards the highest and lowest value of A; so, it is the arithmetic mean of the highest and lowest value of the remaining elements in the vector, according to Equation 7. Equation 8 calculates de adjustment function with the value of the convergence function.

$$cfn(A) = \frac{A[f+1] + A[n-f]}{2} \tag{7}$$

$$Adj = kR_{int} + \delta - cfn(A) \tag{8}$$

And finally at Number 5 of Figure 2, the virtual clock fixes its value by adding their adjustment function.

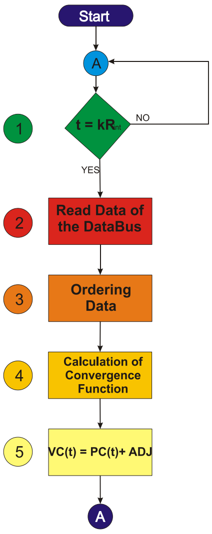

Figure 2. Flowchart of FTM Algorithm.

4. Design of Simulations

In the TrueTime/Matlab/Simulink environment, we simulated two sets of controls, i.e., a system with two control loops connected by a common databus. The controller used was a PID. The actuator/plant is described as continuous in time, according to the following transfer function:

$$G(s) = \frac{1000}{s(s+1)} \tag{9}$$

The controller and sensor nodes had logical clocks given by the virtual computer of the TrueTime Kernel; and they used the databus to exchange data among them and from their clocks. The actuators/plants used the databus only to receive the control data. Each control node implemented a periodic control task and a clock synchronization task. Each sensor node implemented a task for sending the measured data to the controller; and a task for clock synchronization. Each actuator/plant was activated by events when the control task arrived in the actuator by the databus. All nodes had an interruption caused by data arrived from the databus. The model of the simulated control system is given at Figure 3.

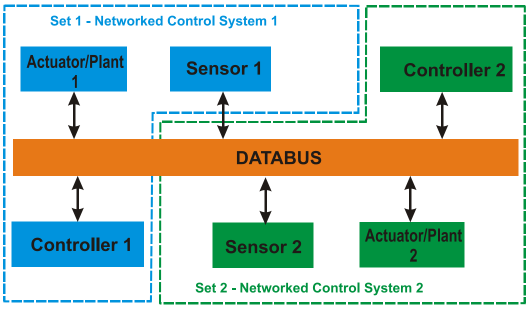

Figure 3. Model of the Simulated Control System.

For these simulations we used the TDMA philosophy for the communication protocols. The communication network is configured by the TrueTime Toolbox.

5. Results

We simulated 2 control subsystems sharing the same databus with a TDMA protocol. The sensors, actuators/plants and controllers are connected via the databus. Errors of clock were inserted in Sensor 1. The objective was to synchronize Controller 1, Sensor 1, Controller 2 and Sensor 2 together; and to compare the cases. In these simulations we varied: the clock drift in Figure 4; and the initial offset of node 2 (Sensor 1) in Figure 5. The databus used a communication protocol with TDMA. The drift rate applied was 1%. This may seem enormous in terrestrial and controlled environments. But, in this paper, we are interested in space environments, where variations of temperature of -10 a 50 degrees, can cause drift rates of up to 1% due to the extreme sensitivity of the quartz clock to temperature, according to Henderson et. al. (2000). In Figure 4, we observe that the FTM algorithm efficiently synchronized the NCS. In Figure 5, the synchronization algorithm did not correct the effects of clock and the system became temporarily unstable. This occurred because the FTM algorithm supposes that all nodes are initially synchronized.

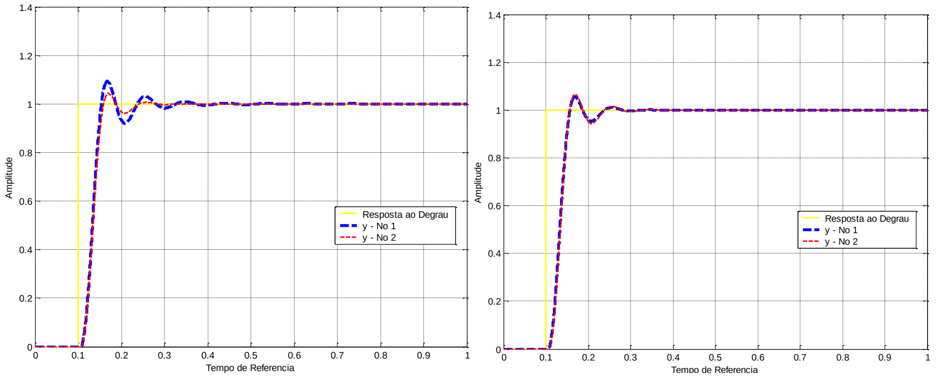

Figure 4. a) Step Response with 1% of drift rate. b) Step Response Synchronized.

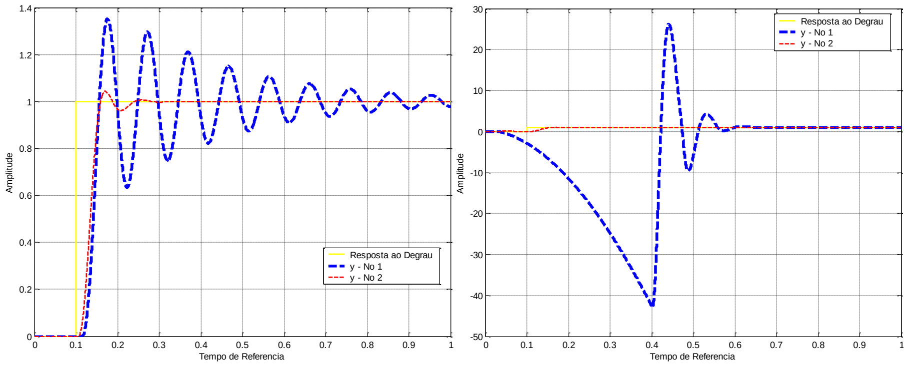

Figure 5. a) Step Response with 0.39 s initial offset. b) Step Response Synchronized.

The temporary instability in the simulation of Figure 5 occurred because sensor 1 starts with an initial positive offset. The virtual computer of sensor 1 identifies overdue tasks and it begins to perform all tasks delayed. The task of sensor 1 is to measure and submit data via the databus to the controlller. By sending the data via the TDMA databus, the volume of tasks is much larger than the slot of transmission time of sensor 1. Thus the tasks of a sensor will be suspended when its slot of transmission time is exceeded. This suspension of tasks generates a large delay. Figure 6a shows the corresponding scheduler of the TDMA communication network, where we can observe this phenomenon: in blue we observe controller 1, in red we observe sensor 1, in green we observe controller 2 and in brown we observe sensor 2. The delay generated by the TDMA network is larger enough that the control system reverses its phase.

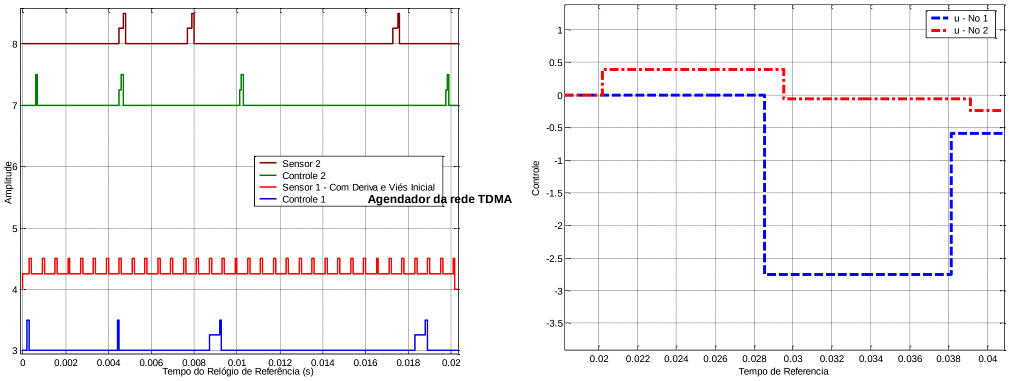

Figure 6. a) Scheduler of Databus Synchronized. b) Aproximation of Figure 5b.

By reversing the phase, the system becomes temporarily unstable, as shown in Figure 6b. For sensor 1, all tasks have already been done. Due to that, until a new task of measurement enters, the system remains unstable. When entering a new task, the system recovers from its instability.

6. Conclusions

This work still is in progress. But the preliminary results suggests that: 1) the TDMA protocol is more susceptible to errors in the initial offset than to errors in the clock drift; 2) the FTM algorithm corrects the clock drift error better than the initial offset error; 3) In a NCS with a TDMA protocol, a fault in the time management can turn the control laws temporarily unstable.

References

Henderson, W.; Kendall, D.; Robson, A. Accounting for Clock Frequency Variation in the Analysis of Distributed Factory Control Systems. Proceedings of the 2000 IEEE International Workshop on Factory Communication Systems, Porto, Portugal, 2000.

Kopetz, H. Real Time Systems: Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, Holland, 1997.

Lundelius, J.; Lynch, N. A New Fault-Tolerant Algorithm for Clock Synchronization. Proceedings of the third annual ACM Symposium on Principles of Distributed Computing, Vancouver, Canada, 1984.

Oliveira Junior, E. M. Study of FTM, FTA and Kalman Filter (KF) Algorithms for the Clock Synchronization and their Influences on a Control System. 2010. 457 p. Dissertation – INPE, São José dos Campos, Brazil, 2010.

Varnum, F. B. Kalman Filtering with a Two-State Clock Model. Proceedings of the 15th Annual Precise Time and Time Interval (PTTI) Application and Planning Meeting, Meeting, Washington, USA, 1983.