2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009



# Fractional Order Networked Control Systems and Random Delay Dynamics: A Hardware-In-The-Loop Simulation Study

Shayok Mukhopadhyay, Yiding Han and YangQuan Chen
Center for Self-Organizing and Intelligent Systems (CSOIS),
Electrical and Computer Engineering Department,
Utah State University, Logan, UT 84322 – 4160, USA
E-mails: {shayok.mukhopadhyay, yiding.h}@aggiemail.usu.edu, yqchen@ieee.org

***Abstract*— In networked control systems (NCS), the spiky nature of the random delays makes us wonder about the benefits we can expect if the "spikiness", or what we call "delay dynamics" are considered in the NCS controller design. It turns out that the "spikiness" of the network induced random delays can be better characterized by the so-called $\alpha$-stable processes, or processes with fractional lower-order statistics (FLOS) which are linked to fractional calculus. Using a real world networked control system platform called the CSOIS Smart Wheel, the effect of modeling the network delay dynamics using non-Gaussian distributions, and compensating for such a delay in closed-loop systems using a FO-PI (fractional order proportional and integral) controller has been experimentally studied. The cases studied include the case when the delay compensated is exactly the same as the actual delay. Other scenarios are the ones when the nature of the estimated delay is similar to the actual delay, but the magnitude is slightly smaller. The effect of phase shifting between the estimated and the original delay is also considered. Finally the order of the fractional order proportional integral controller which gives least ITAE, ISE for a particular distribution of the delay is presented. The conclusion is strikingly stimulating: in NCS, when the random delay is spiky, we should consider to model the delay dynamics using $\alpha$-stable distributions and using fractional order controller whose best fractional order has shown to be related to the FLOS parameter $\alpha$ as evidenced by our extensive experimental results on a real NCS platform.**

***Index Terms*— Fractional order control, fractional calculus, fractional lower-order statistics, $\alpha$-stable processes, spiky, delay dynamics, networked control systems, hardware-in-the-loop simulation.**

## I. INTRODUCTION

A networked control system (NCS) is a system in which the control loops are closed over the network [1]. The feedback and control signals flow between the system components in the form of packets of information over the network. Network control systems are cost effective, hence their research has received impetus in the recent past [2], [3], [4]. The goal of networked control systems is similar to other kinds of control systems, i.e. to achieve stability, and provide good closed loop performance. Rise time, overshoot and various other design criteria may be used to optimize the system [1], [5].

Our previous work [6] showed the application of a jitter-robust fractional order proportional integral (FO-PI) controller based on the tuning rules in [7]. Here we propose to compensate network delays in a setup similar to [6], where the network delays are modeled by $\alpha$-stable distributions. This work also shows that with the presence of spiky random network delays with different distributions, there is no single order for the controller which guarantees best performance.

The remainder of this paper is organized as follows. Section II provides basic information about $\alpha$-stable distributions, their parameter estimation and the application as used in this work. Section III talks about the architecture of the CSOIS Smart Wheel system. Section IV very briefly discusses the system identification process for the plant. It also covers the basics about the hardware operation and entire flow of the hardware-in-the-loop simulation. Section V goes into great detail about the implementation of the simulation model for this particular work. Information about tuning rules developed in [7], the generation of the appropriate distributions simulating the delay and the methods used to estimate the delay are included in this section. In section VI various plots from real-time hardware-in-the-loop experimentation have been provided. This section further discusses the results obtained, and their possible causes. Sections VII and VIII conclude the paper with our inferences from the experimental activity and also some ideas for future work in this direction.

## II. STABLE DISTRIBUTION AND ITS PARAMETER ESTIMATION

### A. Overview of $\alpha$-stable distributions.

*Definition 2.1:* A random variable X is said to have an $\alpha$-stable distribution iff its characteristic function has the following form:

$$\varphi(\upsilon) = exp\left\{j\delta\varphi - \gamma|\upsilon|^{\alpha}\left[1 + j\beta sign(\upsilon)\omega(\upsilon,\alpha)\right]\right\} \quad (1)$$

where

$$\omega(\upsilon,\alpha) = \begin{cases} tan\frac{\alpha\pi}{2} & for\,\alpha \neq 1 \\ \frac{2}{\pi}log|\upsilon| & for\,\alpha = 1 \end{cases} \quad (2)$$

$$sign(\upsilon) = \begin{cases} 1 & for\,\upsilon > 0 \\ 0 & for\,\upsilon = 0 \\ -1 & for\,\upsilon < 0 \end{cases} \quad (3)$$

and

$$0 < \alpha \leq 2, -1 \leq \beta \leq 1, \gamma > 0, -\infty < \delta < \infty \quad (4)$$

Therefore, an $\alpha$-stable distribution can be completely determined by four parameters: **1.** The *characteristic exponent* $\alpha$. It is a parameter which specifies the thickness of the tail (or shape) of the probability density function. In other words, $\alpha$ changes the level of spikiness in the distribution, the larger the value of $\alpha$, the less likely it is to observe any random variable distant from its central location. **2.** The skewness index $\beta$. Positive values for $\beta$ make the distribution skewed towards the right hand tail and negative values make it skewed towards the tail on the left hand side. **3.** The variable $\gamma$ is called the scale parameter and it expresses the dispersion of the distribution. **4.** The variable $\delta$ is called the location parameter and it is an expression of the mean or median of the entire distribution. Details about $\alpha$ - stable distributions and their parameters can be found in [8], [9], [10].

### *B. Application to network delay modeling*

$\alpha$-stable distributions obey two major properties. **1.** The stability property, which states that the sum of weighted independent $\alpha$-stable random variables is still stable with the same characteristic exponent $\alpha$. **2.** The generalized central limit theorem, which states that the sum of a number of independent and identically distributed (i.i.d) random variables can only be a stable distribution. The generalized central limit theorem defines the randomness as a result of cumulative effects and these effects are distributed with a heavy-tailed probability density. The same effects as above

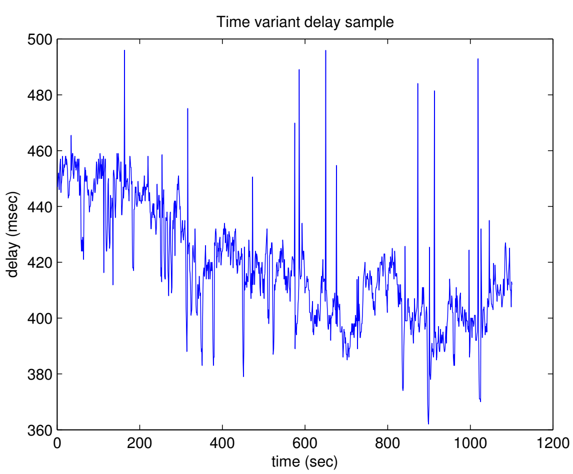


Fig. 1. Network delay samples

cause the randomness of the time variant network delay. It implies that the distribution of such stochastic processes follow the stable model. Figure 1 shows a series of network delay samples measured from the internet. The observed spikes in the figure imply a heavy-tailed distribution. $\alpha$-stable distributions are different from Gaussian distributions. Only moments of order less than the characteristic exponent $\alpha$ exist in a $\alpha$-stable distribution. Hence the variance of a stable (non-Gaussian) distribution is infinite unless $\alpha = 2$, in which case the stable distribution is a Gaussian. This is

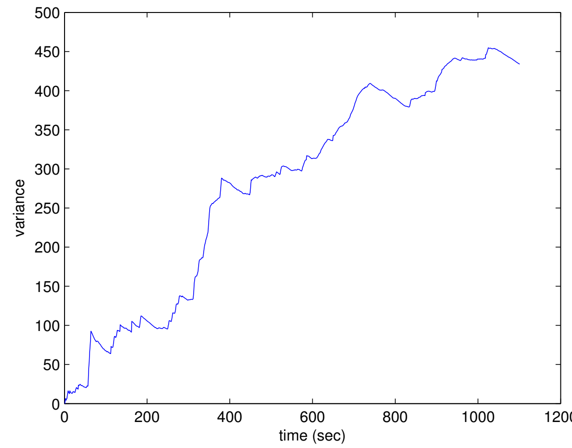


Fig. 2. Infinite variance of network delay

called the *Fractional Lower Order Moment* property. The run-time variance of the random network delay shown in Fig. 1 is calculated and plotted in Fig. 2. It is obvious that rather than converging to a finite value, the variance keeps increasing with time, which means the model of the random network delay distribution should have *Fractional Lower Order Moment* statistics. Therefore, the $\alpha$-stable distribution is chosen to be a statistical model of time variant network delay.

Also [9] shows that $\alpha$-stable distributions can be used to model network traffic. However, as stated above we observed that spikes are found in the network delay too. Thus we decided to use $\alpha$-stable distributions to model random network delays which have a spiky nature. [11] provides insight into the process of simulating stable random variables. The method used by us for estimation/generation of the $\alpha$-stable distributions which simulate the delay in the networked control system are done by the method of maximum likelihood estimation. This technique is very well documented in [12]. The author of [12], has a computer program named 'stable.exe' [13] which we have used to generate the stable-distributions simulating network delay.

## III. Architecture of the Smart Wheel

The Smart Wheel shown in Fig. 3 is a self contained robotic wheel. It is equipped with a steering axis, drive axis and a z-axis, which are capable of independent actuation [14]. The system contains appropriate power and drive circuitry for the motors and encoders for the feedback. It has a microcontroller which sends commands to the motors and actuators and collects data from the encoders. This microcontroller is connected to a serial server, thus enabling the Smart Wheel to be controlled from anywhere over the network.

## IV. System overview

In order to be able to control the Smart Wheel, system identification is absolutely necessary. The system identification is also done through a hardware-in-the-loop simulation.

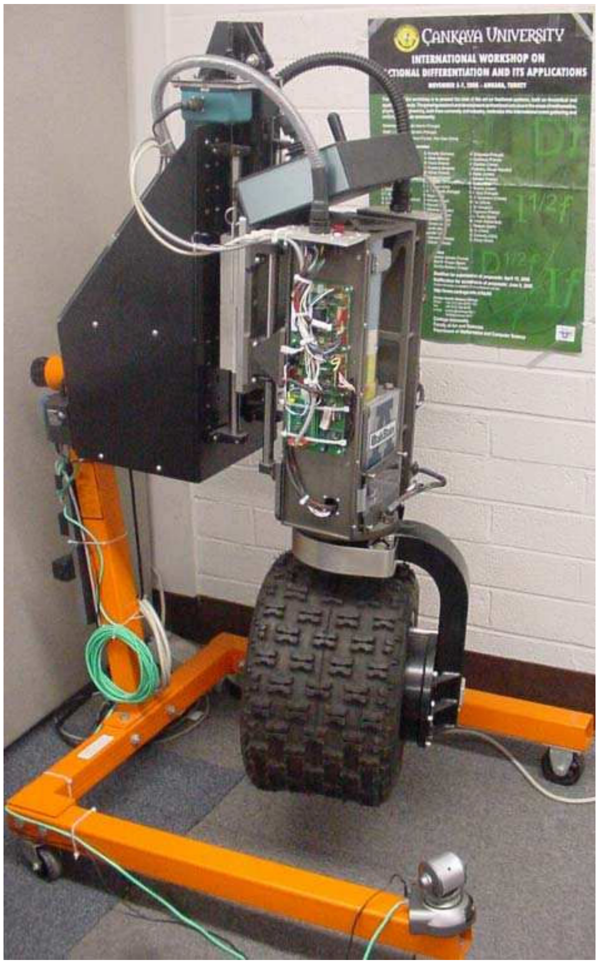


Fig. 3. Stand-alone Smart Wheel on a mobile rig

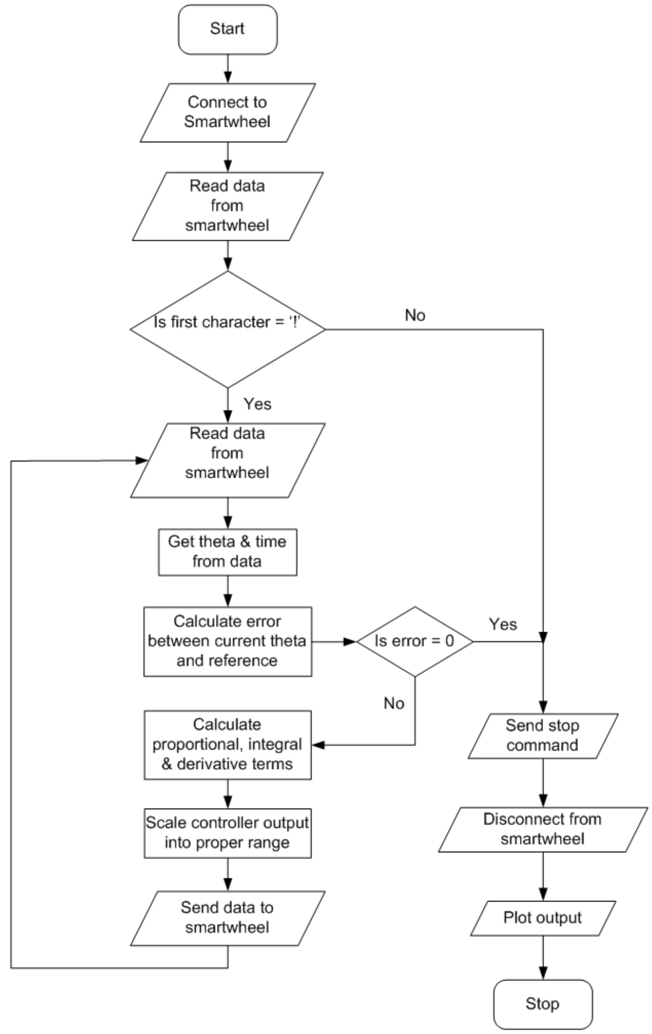


Fig. 4. Flowchart describing the working of the NCS

The entire details of the system identification process can be found in [6]. The first order plus time delay model of the Smart Wheel is obtained as:

$$P(s) = \frac{K}{Ts+1}e^{-Ls} = \frac{0.1484}{0.045s+1}e^{-0.592s} \tag{5}$$

The flowchart shown in Fig. 4 describes the details of the hardware-in-the-loop system. The details provided hereunder are similar to those in [6]. From the flowchart, it is observed that the first step is to connect the Smart Wheel's port and obtain access to it. The connection process is done at the start of the simulation. A count is maintained & connection is performed only on the first ever operation. In the event of timeout or no receipt of connection code, the simulation shuts down. When connected, data is read in from the buffer on the Smart Wheel.

Size of the data available is limited by the size of the buffer, longer data produces greater computational delay and noise when differentiated to get velocity, hence a proper data length selection is very important. The Smart Wheel reports data in a particular format, hence data is parsed to get the value of steering angle $(\theta)$ and time$(t)$ from it. The 's-function' block properly sizes the $\theta$ and $t$ vectors so they are the same length and are synchronized. The current velocity and the error between the setpoint and the velocity is obtained next. If the error is zero then the 's-function' block issues a *REMOTE_REQ* command, *ASCII* character 92, which sends a stop command and the wheel stops spinning. For non-zero error, the controller produces some output. The steering motor rotates in one direction when it receives *ASCII* values 70 to 90 (70 - Stopped, 90 Maximum Speed). It rotates in the opposite direction when it receives *ASCII* values 70 to 50 (70 - Stopped, 50 Maximum Speed). The control input lies in the range of $0-20$ for either direction of rotation.

As seen in Fig. 5, a saturation block is placed in the loop which modulates the controller output to acceptable values. This necessitates the use of an anti-windup feedback block to the integrator to prevent integrator saturation. The correct values are fed into the 's-function' block which communicates with the Smart Wheel and sends appropriate commands. In each cycle, the 's-function' block reads data, parses data, calculates error, takes an input from the controller and sends appropriate commands to the Smart Wheel. The plotting of the data is done at the end of the simulation once the computer has disconnected from the Smart Wheel.

## V. EXPERIMENTAL SETUP AND SIMULATION

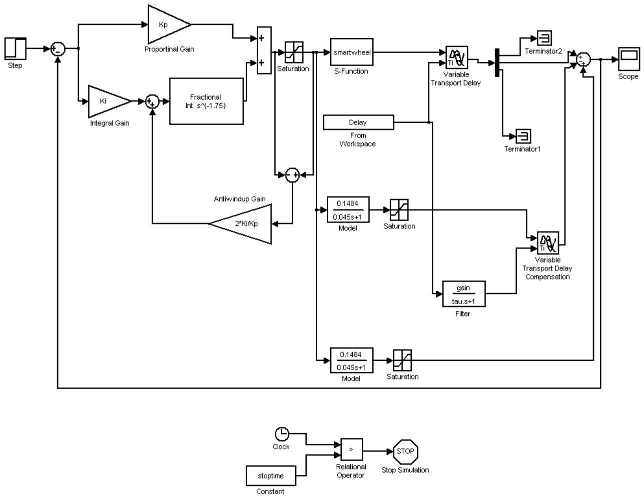


Fig. 5. The hardware-in-the-loop setup

From Fig. 5, it is seen that the closed loop system consists of the fractional order proportional integral (FO-PI) controller followed by the anti-windup blocks and the Smart Wheel 's-function' block. However the major difference between the system presented in [6] and Fig. 5 is that, this model has a Smith's predictor embedded in it. In [6], the delay block used data obtained by pinging certain universities over the network. However here, we consider the effects of delays in the system which have non-Gaussian distributions. The data supplied to the variable transport delay block is therefore strictly non-Gaussian in nature, thus it simulates the randomly occurring spikes in the delay time over the network. It must be noted that since the Smart Wheel is on a dedicated network and also the computer used to control the Smart Wheel is not very far away from it, hence, the real network delay between the two is insignificant and hence additional delay has been added to simulate network conditions.

The FO-PI controller used in this system uses tuning rules developed in [7]. The rules can be stated as:

$$\begin{aligned} K_p^o &= \frac{0.2T}{L} + 0.16 \\ K_i^o &= \frac{0.25}{TL} + \frac{0.19833}{L} + 0.09 \\ \alpha^o &= \tau - 0.04L + 1.2399 \\ \tau &= \frac{L}{L+T} \end{aligned} \tag{6}$$

In the set of equations (6) the superscript $^o$ signifies that the proportional and integral gains and the fractional order $\alpha$ obtained for the controller are optimal. Which is to say that they provide optimal performance for the Smart Wheel system when modeled and used as in [6].

However, the main purpose of this paper is to experimentally study and find out what fractional order $\alpha$ would give the least ITAE or ISE when the gains provided by (6) are used, and the distribution of the network delay is non-Gaussian (or spiky) assuming that such a network delay can be estimated with reasonable accuracy. Simulations and real time hardware-in-the-loop experiments have thus been been conducted for various different scenarios.

Firstly, a set of random vectors is generated to simulate the network delay. These random vectors are so generated that they do not have a Gaussian distribution and their parameter $\alpha_{noise} \in [1.1, 1.2, ....1.8]$, values greater than 1.8 are exempted since beyond this the distribution tends towards a Gaussian and the magnitude and the frequency of spikes in the generated delay vector decreases. The next step is to use a proper estimate of the delay. Assuming that such estimation is possible, the following three scenarios are considered:

**1.** The delay is estimated with full accuracy so that the estimated delay fed to the Smith's predictors variable transport delay compensation block in Fig. 5 is the same as the original delay.

**2.** The delay is estimated accurately in nature but there is some discrepancy in the magnitude. This case is simulated by multiplying the original delay by a gain of 0.9 and using that as the estimate.

**3.** The delay is estimated with marginal error in both nature and magnitude. This case is simulated by passing the original network delay through a filter given by $\frac{0.95}{0.15s+1}$, and using the output as the estimate. In this case the estimated delay is shifted in phase by a small amount with respect to the original delay, and also is slightly lesser in magnitude as compared to it.

Figure 6 shows how the estimated delay looks when estimated by passing it through the filter. The phase shift and difference in magnitude is clearly visible from it. It is important to take care that no negative values result in the compensated delay seen by the closed loop system, because that would make the system non-causal. Also for the sake of clarity Fig. 6 shows only a small part of the original delay vector.

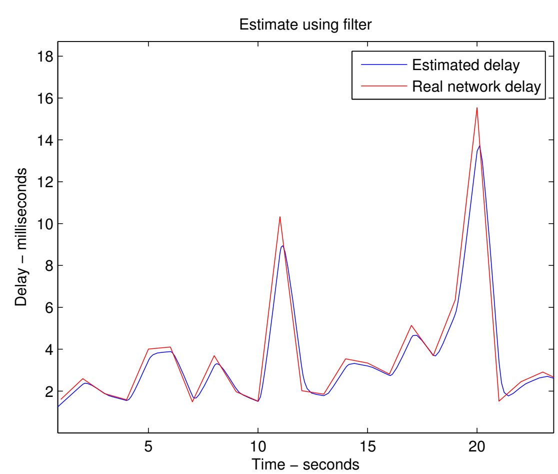


Fig. 6. Delay estimation by passing it through a filter

The experiment thus consists of running the hardware-in-the-loop simulation for different values of the delay having non-Gaussian distributions with $\alpha_{noise} \in [1.1, 1.2, ....1.8]$, for each value of $\alpha_{noise}$ three different estimates are used for the delay as described above. And further for each different combination of $\alpha_{noise}$ and the delay estimate, the experiment is run for fractional orders of the controller ranging from $\alpha_{controller} \in [0.1, 0.2..., 1.95]$. For further comparison the optimal PID and FO-PI controllers designed in [6] are tested with a square wave reference and with delays having Gaussian, and non-Gaussian distributions present in the loop.

## VI. Experimental Results

When the estimated network delay used is the same as the original delay, the $\alpha_{controller}$ which gives the minimum values for ISE and ITAE do not differ too much for different values of $\alpha_{noise}$. From Fig. 7 and Fig. 8 it can be seen that the values for $\alpha_{controller}$ which give a minimum value for ISE are widely different from those which give a minimum ITAE for different $\alpha_{noise}$ when a gain of 0.9 is used to estimate the noise. However from Fig. 9 and Fig. 10 it can be seen that the values for $\alpha_{controller}$ which give a minimum value for ISE and ITAE are not greatly different (as compared to the previous case) for different $\alpha_{noise}$ when the noise is estimated using a filter.

Figure 11 shows that the optimal FO-PI controller performs better than the optimal PID controller in the presence of non-Gaussian network delays. The FO-PI controller has a much faster response and a comparable overshoot. In some parts the overshoot is lesser than the optimal PID controller. From Fig. 12 it is seen that the optimal PID controller does a better job than the FO-PI controller in the presence of Gaussian network delays as the FO-PI controller has a much higher overshoot. Figure 13 compares the performance of the FO-PI controller for different delays, it is amply clear that there is a lot to be gained using it in the presence of non-Gaussian delays. It must be noted that $\alpha_{noise} = 1.2$ has been selected as a representative case for carrying out the experiments with non-gaussian delays, and $\alpha_{controller}$ is selected as in [6] since it was desired to test the performance of such a controller in the presence of varying delays.

## VII. Conclusion

From the above discussion, it can be concluded that, no single value of the order $\alpha_{controller}$ will guarantee that the ITAE or ISE achieved by the controller is minimum with the presence of varying non-Gaussian, alpha-stable, random spiky network delays. Moreover, for a certain $\alpha_{noise}$ there is optimal $\alpha_{controller}$ which gives best performance.

It is also clear that an optimal FO-PI controller performs much better than an optimal PID controller in the presence of non-Gaussian delays.

## VIII. Future Scope

Definite future directions of work include finding a mathematical relation between the characteristic exponent $\alpha_{noise}$ and the order $\alpha_{controller}$ which gives optimal controller performance. The application of "fractional order - (proportional

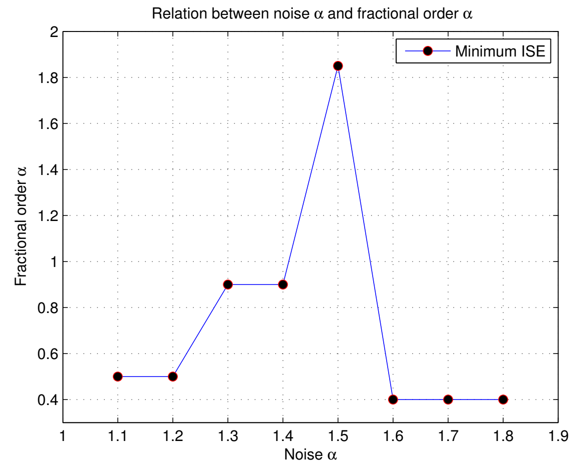


Fig. 7. $\alpha_{noise}$ vs $\alpha_{controller}$ for gain $= 0.9$

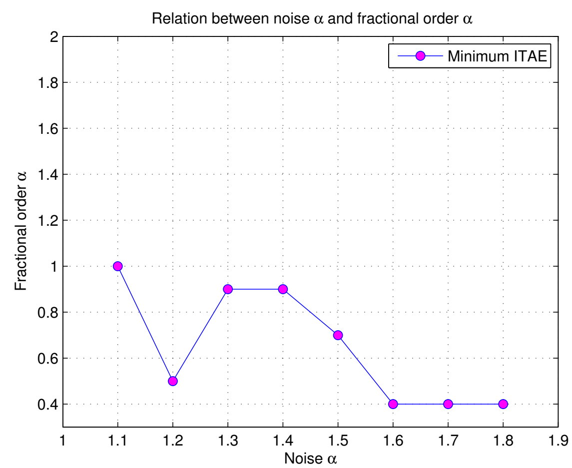


Fig. 8. $\alpha_{noise}$ vs $\alpha_{controller}$ for gain $= 0.9$

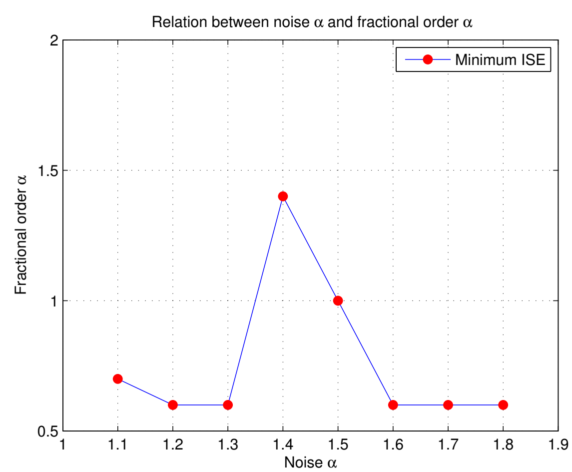


Fig. 9. $\alpha_{noise}$ vs $\alpha_{controller}$ with filter

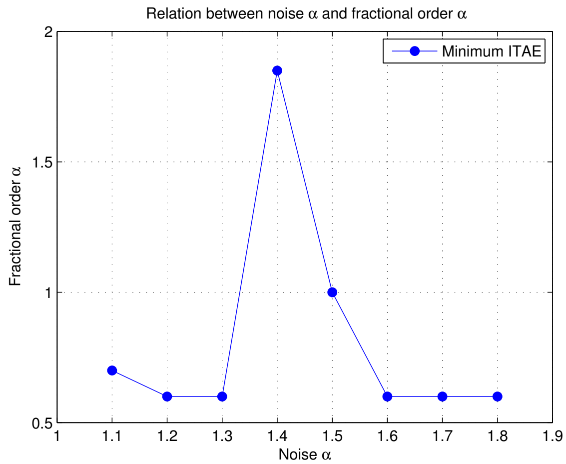


Fig. 10. $\alpha_{noise}$ vs $\alpha_{controller}$ with filter

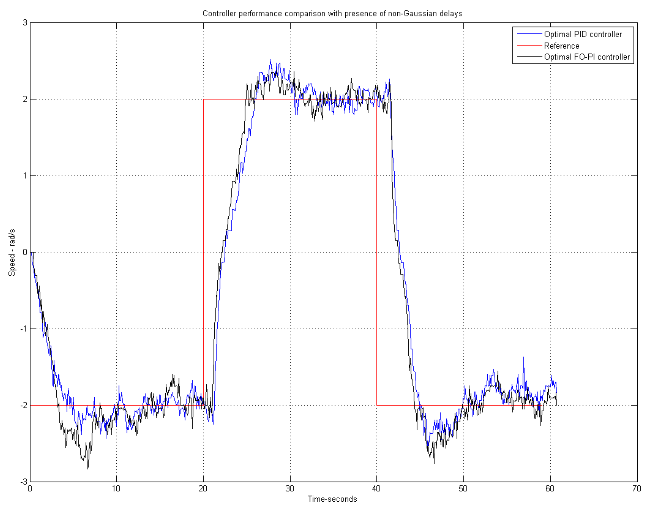


Fig. 11. Controller performance with non-Gaussian delays

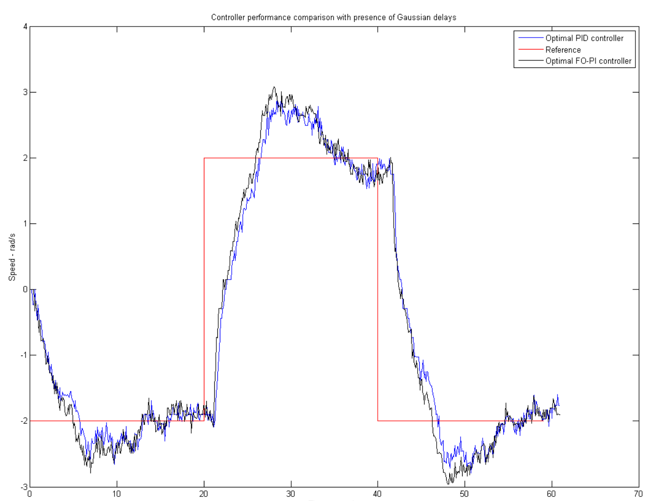


Fig. 12. Controller performance with Gaussian delays

integral)” FO-(PI) controllers of the type $\{K_p(1+\frac{K_i}{s})\}^{\alpha}$ to

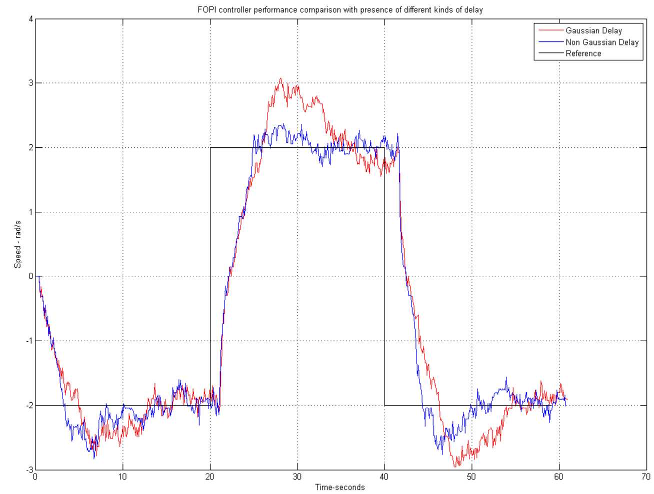


Fig. 13. FO-PI controller comparison for different delays

such systems affected by time variant delay compensated by estimated delays having non-Gaussian distributions shall provide some interesting results.

## REFERENCES


[1] O. Beldiman, *Networked Control Systems*. Ph.D. Dissertation, Dept. Electrical and Computer. Eng., Duke Univ, 2001.

[2] W. Zhang, M. S. Branicky, and S. M. Phillips, “Stability of networked control systems,” *IEEE Control Systems Magazine*, pp. 84–99, February 2001.

[3] F. L. Lian, J. Moyne, and D. Tilbury, “Anaysis and moelling of networked control systems: MIMO case with multiple time delays,” *Proceedings of American Control Conference*, June 2001.

[4] Y. Tipsuwan and M. Y. Chow, “Control methodologies in networked control systems,” *Control Engineering Practice 11*, 2003.

[5] B. Ramaswamy, Y. Q. Chen, and K. L. Moore, “Omni-directional robotic wheel - a mobile real-time control systems laboratory,” *International Journal of Engineering Education (SmartWheel web:* `http://www.neng.usu.edu/ece/csois/people/smartwheel/`*)*, vol. 24, no. 1, Jan 2008.

[6] V. Bhambhani, Y. Han, S. Mukhopadhyay, Y. Luo., and Y. Q. Chen, “Random delay effect minimization on a hardware-in-the-loop networked control system using optimal fractional order PI controllers,” *Third IFAC Workshop on Fractional Differentiation and its Applications, FDA, Ankara, Turkey*, November 2008.

[7] V. Bhambhani, Y. Q. Chen, and D. Xue, “Optimal fractional order proportional integral controller for varying time-delay systems,” *17th IFAC World Congress.*, July 2008.

[8] X. Ge, G. Zhu, and Y. Zhu, “On the testing for alpha-stable distributions of network traffic,” *Computer Communications*, vol. 27, p. 447457, November 2004.

[9] A. Karasaridis and D. Hatzinakos, “Network heavy traffic modeling using $\alpha$-stable self-similar processes,” *IEEE Transactions on Communications*, vol. 49, no. 7, pp. 1203–1214, July 2001.

[10] M. Shao and C. L. Nikias, “Signal processing with fractional lower order moments: Stable processes and their applications,” *Proceedings of the IEEE*, vol. 81, no. 7, pp. 986–1010, July 1993.

[11] J. M. Chambers, C. L. Mallows, and B. W. Stuck, “A method for simulating stable random variabless,” *Journal of the American Statistical Association*, vol. 71, no. 354, pp. 340–344, June 1976.

[12] J. P. Nolan, “Maximum likelihood estimation and diagnostics for stable distributionss.”

[13] “John Nolan’s Stable Distribution Page.” [Online]. Available: http://academic2.american.edu/ jpnolan/stable/stable.html

[14] N. Flann and K. Moore, “A six-wheeled omnidirectional autonomous mobile robot,” *IEEE Control Systems Magazine*, 2000.