A Real-time Web-based Networked Control System Education Platform

J. Cavalcanti, L.F.C. Figureredo, J.Y. Ishihara, M.C. Bernardes, P.H.R.Q.A. Santana, A.N. Vargas, G.A. Borges

March 10, 2018

Abstract

Networked control systems (NCSs) have recently experienced a surge in fundamental theoretical results ignited by numerous advantages of introducing shared multipurpose communication networks in control systems. Regarding developments on the practical side, however, NCS is still wanting when it comes to experimental platforms suitable for research and educational purposes, which contributes to most of NCS theory being validated by means of numerical examples and simulations. This paper addresses this issue by presenting a low-cost real-time NCS platform, based on custom hardware and software solutions that can be readily explored with the sole use of a web browser connected to the Internet. The technical decisions made during development represent a fundamentally novel take on NCS experimental platforms that can potentially be reproduced by several universities. The platform provides the user with multiple controller and input reference options, network configurations, delay statistics, and even a downloadable file containing advanced experiment data. A survey conducted with students located over 1200 km away from the platform who used it during laboratory assignments highlight the system's usability and interactivity, and supports the platformissuitable for educational purposes.

1 Introduction

Communication networks have been responsible for fundamental paradigm shifts in control systems by allowing their elements to communicate remotely, enabling distributed and centralized control. From a technological standpoint, networks also contribute to reducing constructive costs and improving maintainability. Altogether, these features made Networked Control Systems (NCSs) a priority topic within the control community (Ref. 6), which is reflected by the soaring number of published articles related to NCS in the past ten years, shown in Fig. 1.

The introduction of shared multipurpose communication networks in control systems, however, is not entirely beneficial. Counter-effects—among which

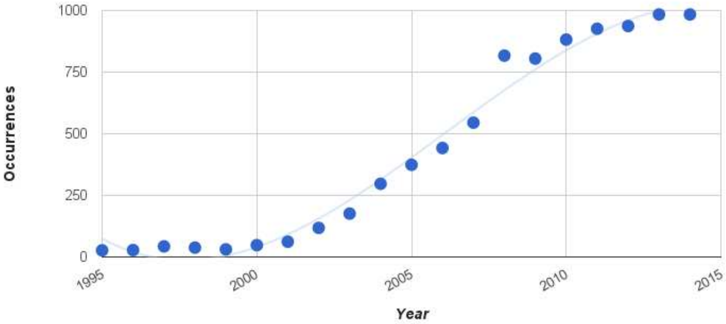


Figure 1: NCS-related papers occurrences in Scopus during the past ten years.

communication delays and packet dropouts are the most problematic---can degrade system performance and even destabilize the NCS. The past two decades have witnessed remarkable theoretical contributions to cope with such issues, and established novel analyses and design frameworks, which can be found in excellent tutorials (Fridman, Hespanha, Zhang,).

NCS theory, however, outpaced practice when it comes to experimental validation. The lack of suitable platforms for that purpose contributes to most NCS theory be validated either numerically or by simulation. Indeed, NCSLab (Ref. 12) is, to the author's best knowledge, the sole existing NCS remote laboratory. It allows users to design their own controllers, and experiment with devices located in China and the UK. Nevertheless, using NCSLab presupposes having access to MATLAB on the user's machine, for controller design takes place on MATLAB environment, followed by uploading the design, which is then compiled by a MATLAB server prior to be implemented in the real NCS. Although this gives users more freedom in the design process---any compiling design is allowed---it also excludes users who do not have MATLAB, and the design/upload/compile procedure can make the whole NCSLab experience time-consuming and complex, as reported by the very authors of Ref 12. Moreover, NCSLab demands extensive infrastructure, which---along with mandatory MATLAB controller design---can render its approach to NCS experimental platform prohibitive to many universities, especially in developing countries.

The lack of experimental NCS platforms is in sharp contrast with other control, and engineering areas. Indeed, remote control laboratories have been available even before Automatic Control Telelab (Ref 3.)---one of the most influential remote control laboratories. Now, control students are able to experiment with several types of processes and controllers (Refs. ?), whereas in robotics, for example, initiatives such as PR2 (^Ref. 14, Pitzer) promote research by offering validation resources to scientists.

rtNCS addresses the gap in low-cost and user-friendly NCS laboratories.

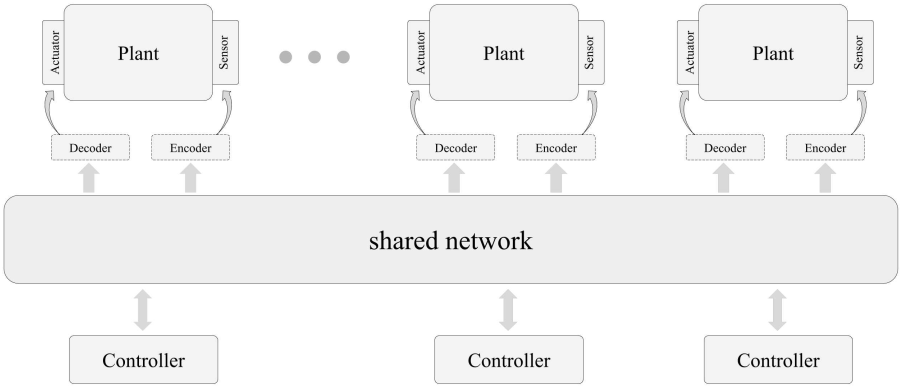


Figure 2: General architecture for a networked control system.

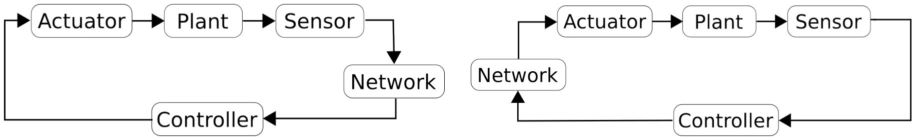
It is available at https://www.lara.unb.br/rtncs, and based at the Automation and Robotics Laboratory (LARA), University of Brasilia. The platform's features include modifying controllers, input references, and network configurations, but the various choices in NCS settings do not hamper affordability and user-friendliness, providing the control community with a competent experimental platform aimed to foster research and education in NCS. Hardware accounted for US$ 2,000.00---neglecting the server---which is the outcome of adopting inexpensive, commonly available electronic components and represents a novel approach to remote NCS experimental platforms that can be followed by many other universities and institutions.

The remainder of the paper describes how rtNCS works by presenting its architecture and implementation in more detail, illustrates website usage, and discusses results of a survey conducted with students located over 1200 km away from the experiment.

2 rtNCS

NCSs distinguish themselves from traditional control systems in that their elements are connected via multipurpose shared communication networks where information is exchanged in form of data packets. Figure 2 depicts a general NCS, where sensor data concerning the plant's state are sent via network to be processed by the controller, which in turn sends control signals to the actuator via the same—or even a different—shared network. Particular cases of this general-form NCS are shown in Fig. 3(a) and Fig. 3(b), where the NCS is only partly connected through networks.

rtNCS comprises two independent networks, one of which is an actual NCS that can assume either the general or particular forms illustrated in Figs 2-3, whereas the other network represents the front-end application communicating with the website server, which interfaces the two networks and is also part of



(a) Sensor-controller network only.

(b) Controller-actuator network only.

Figure 3: Particular NCS architectures with single loop delays.

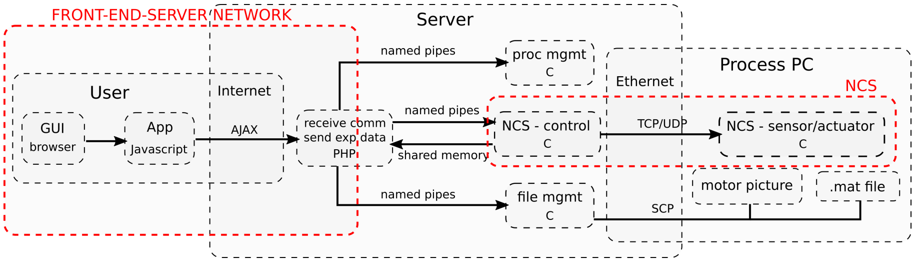


Figure 4: rtNCS's internal processes, interfaces, and networks.

the NCS. Both networks, plus processes and interfaces are shown in Fig. 4.

2.1 The NCS

When experimenting with rtNCS, the user will be controlling the angular position of a MAXON DC motor---shown in Fig. 5---with a dedicated PC. In general-form NCS mode, this computer runs a C program---called CP, denoted as NCS-sensor/actuator in Fig. 4---that samples motor angular position, and that communicates via Ethernet with another C program running on the server--called CS, denoted as NCS-control in Fig. 4---that computes control signals to be sent back to CP, which applies these signals to the plant. CS acts as a master of the NCS, since it sets the controller type and parameters, input reference, and network configuration. Angle measurements are taken from a multi-turn potentiometer attached to the motor shaft, and go through a low-pass filter before being sampled to prevent signal aliasing. The filtered potentiometer signals are retrieved by CP via an ICP DAS PIO-821 data acquisition board through ICP interface. This is performed periodically with high precision frequency because CP runs on Xenomai, a real-time Linux framework, which also guarantees task-scheduling delays are constrained, preventing these delays from outgrowing the network's, what would mischaracterize the NCS we seek to analyze. It must be highlighted that NCSLab (ref 12) does not ensure real-time task execution, because it does not run on a real-time operating system; task-scheduling delays are neglected.

If the user chooses alternative NCS configurations from Fig. 3, CS will only mirror incoming data packets rather than calculate control signals. If sensor-

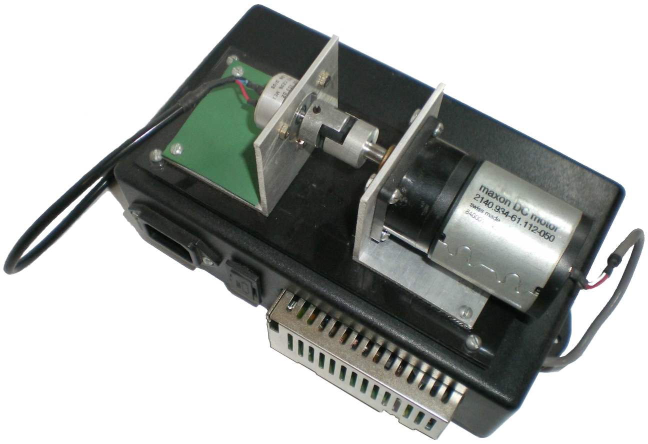


Figure 5: Brushed DC motor; NCS plant.

controller network only---see Fig. 3(a)---is selected, CS sends back sensor data coming from CP, which will compute the control law. In controller-actuator network only mode, however, CP pre-calculates control signals before sending them to CS, which mirrors the signals back to CP. Thus, the overall behavior of the NCS matches that of the corresponding NCS configuration. This kind of flexibility, it must be said, is not available on NCSLab either (CHECAR!). (IMPORTANTE AQUI NÉ)

2.2 Front-end application and server interface

The two rtNCS internal networks interface at the server, primarily responsible for allocating and controlling the NCS process. An AJAX/PHP mechanism enables the front-end application to communicate with the NCS: the front-end application transforms user actions into AJAX requests that call PHP scripts on the server side---see Fig. 4. These scripts exchange information with three C programs, two supervisory processes and CS, using FIFO named pipes and shared memory segments. Once the PHP scripts finish processing, they answer the AJAX requests that are on hold, which are then treated by front-end Javascript, rendering the dynamic behavior of rtNCS webpage.

Supervisory applications manage the NCS process, the visual feedback displayed on the website page, and the data file the user may download after finishing the experiment. Centralizing process management was decided with the expansion of rtNCS in mind, since it allows standardization of communication interfaces between processes and the supervisory, and makes it easier to synchronize process allocation across the entire platform. The other supervisory application handles the files coming from process PC to the user, namely, the picture that provides visual feedback from the DC motor, and the data file from the last experiment run by the user. When PHP scripts request these files, the file supervisory triggers scp transfers, and once the files are available, they are transferred to the user. This is an advantage of decoupling NCS from the rest of rtNCS, because auxiliary functionalities such as file management can be added and performed independently of the experiment.

User-friendliness is key to attract interest of students and researchers, which is why it is, along with low-cost, the focus of rtNCS. User interface was conceived so it would be possible to experiment with NCS in the first contact with the platform without any training. The buttons and text inputs are self-explanatory, but help icons and hovering the mouse over buttons also provide essential information on how to use the platform. Software requirements are also regarded in terms of user-friendliness. Thus, a web browser capable or running Javascript is the only requirement to use rtNCS. Javascript enables data exchange with the server to be performed on the background via AJAX requests, which avoids page reloading to refresh page information. Also, users can exploit experiment data through a data file compatible with mainstream numerical environments such as Octave---available for free---or MATLAB.

3 Experimenting with rtNCS

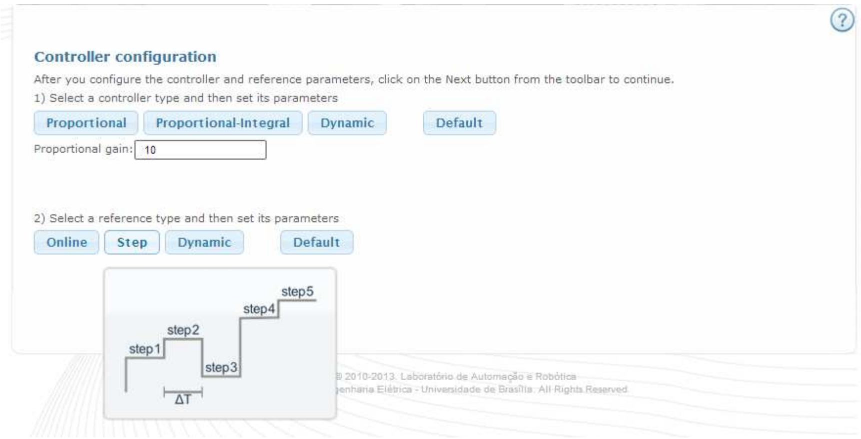
The homepage of rtNCS briefly introduces the platform and provides a button to access the NCS experiment. Once the user clicks on it, running the actual experiment is only three-steps away:

1. Define controller structure (Proportional, Proportional-Integral, or Dynamic) and select the gains, as Fig. 6 shows.

2. Set the reference (Online, Step sequence, or Dynamic) and its parameters, as in Fig. 6. Online reference enables the user to alter input reference while the process is running; tep reference allows two to five different reference levels to be set, which will be periodically executed, and also the execution period to be chosen; if dynamic reference is selected, state transition and output matrices of a discrete dynamic system are set, and the system's output provides the reference-input.

3. Pick one over four different network configurations, shown in Fig. 7: local control, sensor-controller delay only, controller-actuator delay only, or networked control. It is also possible to add an average communication delay that will sum to the network's intrinsic delay.

Whenever a parameter must be configured (e.g., controller gain), the user is also provided with default values that can be loaded, enabling a newcomer to quickly start familiarizing with the system. Once setup is complete, the experiment launches, numerical and visual feedback are provided, as Fig. 8 depicts. Numerical data are displayed in form of two charts of time-varying signals: network delay and angular position, which can also be assessed from a picture periodically updated at the bottom of the page. In addition to delay charts, network delay statistics---minimum, maximum, and mean delay---can be obtained by clicking on the 'Get Delay' button.



Controller configuration

After you configure the controller and reference parameters, click on the Next button from the toolbar to continue.

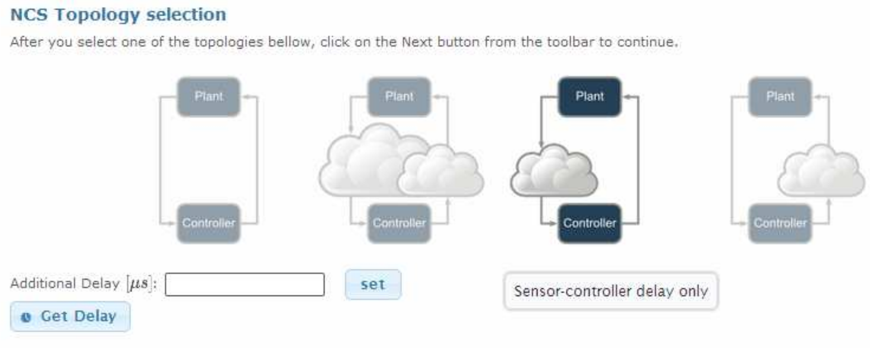
1) Select a controller type and then set its parameters

Proportional gain: 10

2) Select a reference type and then set its parameters

© 2010-2013. Laboratório de Automação e Robótica
genharia Elétrica - Universidade de Brasília. All Rights Reserved

Figure 6: Controller and reference selection.

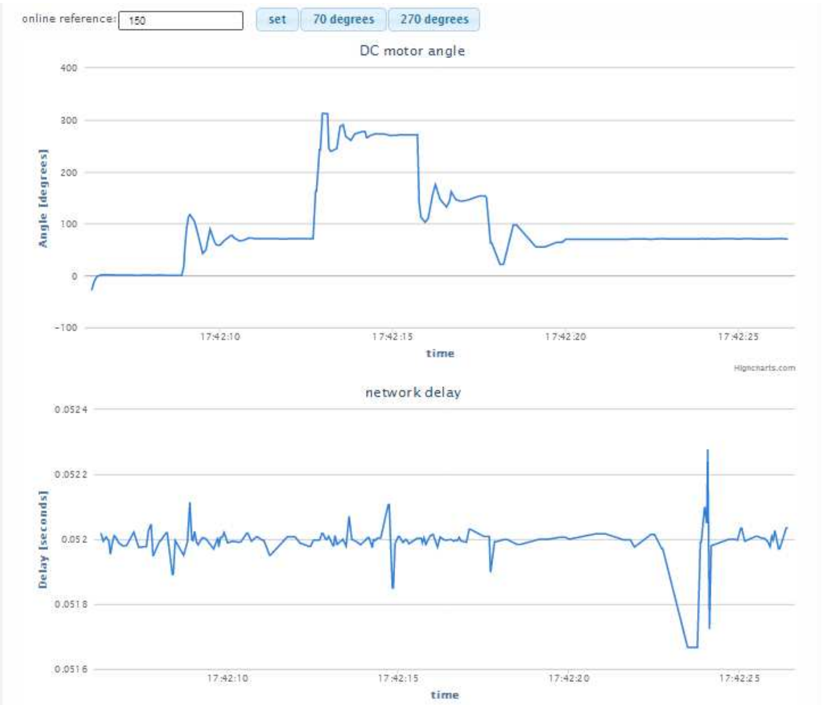


NCS Topology selection

After you select one of the topologies bellow, click on the Next button from the toolbar to continue.

Additional Delay $[\mu s]$:

Figure 7: Network configuration.



online reference: 150

Figure 8: rtNCS running experiment.

Mild software requirements and a self-sufficient dynamic webpage interface that does not require downloading (e.g., Java applets, Ref. 2) or uploading files (Ref. 12) to be operated make it possible for users to switch from one NCS setup to another real quick, altering controllers, input references, network configuration, and even average delay. This sets rtNCS apart from previous remote laboratory platforms, which generally demand users to go through a design/upload/compile process every time the experiment setup is modified.

3.1 Teaching NCS using rtNCS

In order to evaluate if the pedagogical benefits of using rtNCS to introduce NCS, and if the basic concepts that guided development of the platform could actually be noticed, twenty seven electrical engineering students from the Technological Federal University of Parana (UTFPR)---based in Cornelio Procopio, Parana, over 1200 km away from Brasilia, where rtNCS server is set---used rtNCS as part of laboratory activities between August and September 2013, without having any prior contact with the platform. Students were on the sixth and seventh semester, and had already taken a course on control systems.

After completing lab assignments, they were asked to answer an anonymous questionnaire according to a 1-5 scale, ranging from strong disagreement (1), to indifference (3), to strong agreement (5). Students were also encouraged to provide any kind of feedback they considered pertinent. The questions and answers are shown in Table 1 and Fig. 9. Averages from Q3, Q4, Q9, and Q11, confirm rtNCS is user-friendly and provides a smooth user experience, whereas Q1's corroborate the platform can help students learning NCS, and gain insights on what NCS is by experimenting with a real process. Students agreed software requirements were not strict, and enjoyed working from home. Indeed, Q2 and Q10 averages reveal this kind of activity is appreciated by students, and that remote laboratories should be more often incorporated into traditional lab activities. The two lowest-rated questions, Q7 and Q8, concern familiarity with MATLAB and using MATLAB data for further experimentation, which, in fact, suggests requiring prior knowledge of MATLAB can repel some users.

In terms of improvements and future work, Q5 indicates decreasing the platform's response time should be a priority. Three students mentioned they would appreciate if update rate and resolution of the motor images were increased. Furthermore, the system could be made more robust to web browsers other than mainstream ones (Chrome, Firefox, Safari and IE).

4 Conclusion

From the developer perspective, rtNCS presents a novel approach to NCS experimental platforms in that it was built out of common hardware and custom software that is license-free, and in that the process end of the NCS experiment runs on a real-time operating system, which minimizes the impact of task-scheduling delays on NCS characterization---i.e., task-scheduling delays are constrained and

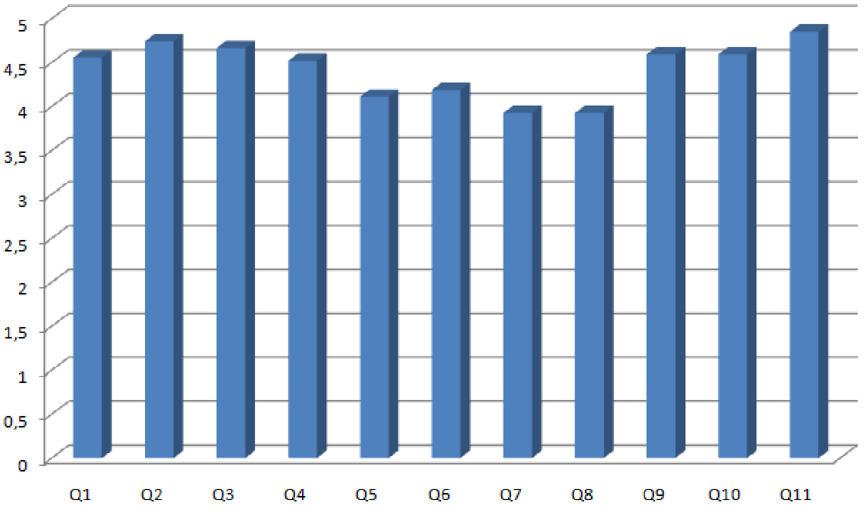


Figure 9: Survey results.

do not outgrow the network's. Although developing custom hardware and software can be time consuming, it also allows costs be kept low---US$2,000.00, precisely---which should make this take on NCS experimental platforms affordable to several universities.

Independence of proprietary software also impacts on user experience because experiment setup can be entirely carried on rtNCS's webpage, rather than on the user's PC, i.e., controllers need not be uploaded nor compiled, which enables experiment scenarios be switched real quick. Nevertheless, the platform also provides the user with a file containing experiment data that is compatible with numerical environments such as Octave and MATLAB. This allows both beginners and more advanced users to use platform for different purposes. Researchers can use rtNCS as validation platform, whereas students can familiarize themselves with NCS by playing with a real NCS, changing delays, network configurations, controllers, and input references.

The positive response from students who used rtNCS as part of their laboratory assignments confirm the concept of the platform is valid, and now focus should be given to improving response time, extending the spectrum of compatible web browsers, and adding more NCS plants. This last point should not pose a problem because scalability was kept in mind since the project started; the server-NCS interface, based on named pipes and shared memory segments, can be replicated for any plant, and only Javascript code specific to each new plant will need to be developed.

Table 1: Survey Results.

Nº	***Questionnaire***	***Response***					
	The marks correspond to 1=Strongly disagree, 2=Disagree, 3=Unsure, 4=Agree, 5=Strongly agree	1	2	3	4	5	Avg.
Q1	Does the laboratory help to understand the concept of NCS?				12	15	4.56
Q2	It has been easy to work from home				7	20	4.74
Q3	Was the web-laboratory easy to understand and use?			1	7	19	4.67
Q4	Were you able to fully use the web-laboratory by following the instructions provided?			3	7	17	4.52
Q5	How was the response time of the laboratory?		1	4	13	9	4.11
Q6	Software requirements did not pose a problem	2	1	2	7	15	4.19
Q7	Are you familiar with MATLAB?		1	8	10	8	3.93
Q8	Did you use the MATLAB data for further exploration of the results?		2	7	9	9	3.93
Q9	Did you enjoy the overall experience using the remote laboratory?			2	7	18	4.59
Q10	Would I like to do more experiments using remote laboratories in other courses?			1	9	17	4.59
Q11	Would you recommend this course to other students?			1	2	24	4.85