Hybrid Online Labs: Making Remote Laboratories Usable Under Unfavorable Bandwidth Conditions

http://dx.doi.org/10.3991/ijoe.v8i4.2158

K.P. Ayodele, L.O. Kehinde and O.A. Komolafe
Obafemi Awolowo University, Ile-Ife, Nigeria

***Abstract*—One of few limitations of remote laboratory technology is the fact that access and usability of such laboratories depend largely on the existence of favorable bandwidth conditions between the remote user and the system under test. This dependence is regrettable because some of the institutions likely to find remote laboratories attractive are also those most likely to have severe bandwidth limitations. Also, a typical remote laboratory will be completely unusable to remote students in the event of an outright network downtime. In this paper, we propose a hybrid online laboratory architecture that allows the automatic generation of accurate software models of remote laboratories. Such models can be hosted closer to the student and during periods of unfavorable bandwidth conditions, students can successfully interact with such models in lieu of the real hardware. We identify the challenges that need to be resolved for such a scheme to be useful and discuss the process by which suitable modeling bases were chosen. Finally we present and discuss data from a first test of the system and conclude that such a scheme holds considerable promise in changing the way remote laboratories are used and viewed.**

***Index Terms*—bandwidth, hybrid, online, remote laboratory, simulation, system identification**

I. INTRODUCTION

Remote laboratories allow students to carry out laboratory work even when they are located thousands of miles from laboratory equipment [1], [2]. While this has many benefits [3], [4], the separation of student from equipment leads to a number of challenges. Notably, the usability and effectiveness of remote laboratories depend heavily on the conditions of the network link between the student's computer and the remote hardware. When available bandwidth is low, the laboratory experience is adversely affected [5] and if any segment of the link experiences outright failure, the student is unable to carry out his experiment.

Consequently, current remote laboratory technology is something of an all-or-nothing technology. With excellent bandwidth conditions, students using a remote laboratory can benefit from all the advantages attributed to the class of resources. However, when there are less-than-ideal bandwidth conditions between the student and the remote hardware, problems may arise. During a full network downtime, a remote laboratory is completely unusable. This is regrettable because some of the institutions that are most likely to find remote laboratories useful are also those most prone to bandwidth and connectivity problems.

Considering the widely-held belief that educators are largely resistant to change [7], it should not be surprising if they fail to fully embrace a solution whose functions may be limited by prevailing internet connectivity conditions.

It can be argued that a ready-made solution to this problem exists in the form of simulations, either on the student's local computer, or througha virtual laboratory environment. Indeed, use of a simulation provides its own set of unique advantages [8]. Nevertheless, there is evidence that many instructors would take a well-designed remote laboratory (if such is available) over a simulation.There are justifications for this preference. For example, [8] concluded that "*the interaction with real equipment and real data in remote labs affords for more authentic inquiry, trustworthiness of data, a greater personal investment, a sense of presence and reality, and a stronger preference for remote labs than performing a similar experiment with simulated data*".

Consequently, it appears that for most instructors, the three popular student experimentation paradigms can be arranged in order of preference as shown in Figure 1. In the absence of a real laboratory, instructors would generally prefer to work with a remote laboratory if possible,but would alsoaccept using a simulation.

In this paper, we present anonline laboratory architecture, the "hybrid online laboratory",that makes use of this information. Under normal operation, the hybrid online laboratoryis a remote laboratory, providing remote access to real hardware. However, when the link to the remote hardware is downor excessively slow, the architecture allows a student to interact with a software model of the hardware instead. In other words, the new architecture is not just a remote laboratory or a virtual laboratory, but a hybrid of the two.

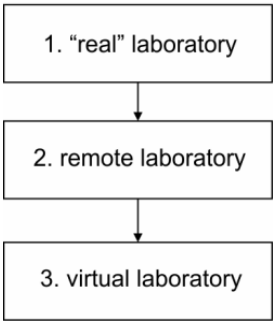


Figure 1. Three common laboratory experiment paradigms arranged in the order of preference shown by most instructors

In a hybrid online laboratory, a software model of the remote laboratory system under test (SUT) is generated. Since the internet link is the usual source of bandwidth-related problems, the generated model is domiciled somewhere on the user's intranet (either on a client-side server or on the student's machine). Whenever there are problems with the Internet link, the remote laboratory front-end or client can route the student's requests to the model (after notifying the student and obtaining the student's assent), which can generate results that, in a properly-designed system, would be in agreement with what would have been generated by the real hardware.

While the idea of using a simulation for experimentation is certainly not new, the novelty in our work lies in the fact that our architecture provides for decision logic that automatically decides whether to link students to a real hardware system or a simulation. Furthermore, we describe how the software model can be developed automatically without any human intervention, using subspace state space system identification.

The rest of the paper is arranged as follows: in Section II, a discussion of relevant system identification concepts is undertaken, concluding with the selection of a suitable class of algorithms. Section III presents the architecture of the hybrid online laboratory,and the testing of a sample implementation is described in Section IV. The results and issues arising are discussed in Section V.

II. FRAMEWORK FOR AUTOMATIC MODEL GENERATION

Although there is some appeal in the notion of an architecture that generates a model to replace the SUT of a remote laboratory when necessary, there are two challenges that must first be addressed. First, the generated model must be an accurate replica of the real system, otherwise results obtained with it would not be considered pedagogically useful. Secondly, unless some means is found to automate the model-generation process, the scheme might only find limited application as it would be unrealistic to expect that a modeling expert could be found to develop models for each and every SUT. In summary therefore, a hybrid laboratory should be able to automatically generate models for its SUT(s), and these models must be very accurate.

There are two broad approaches to modeling. In modeling from first principles or white box modeling, a set of mathematical or logical structures is developed from a thorough understanding of the scientific principles underpinning the operation of the real (or "target") system. In the other approach, called system identification, a model is generated from input and output data from the system [9]. When the development of models from system behavioral data benefits from insight (called a priori information) into the behavior of the corresponding real life system, the process is called gray-box modeling. Conversely, black-box modeling is said to have taken place when no prior insight into the target system is available.

Models developed by system identification can be parametric or non-parametric. Whereas nonparametric models represent the target system as curves and functions, parametric models employ finite-dimensional parameter vectors. Although nonparametric models are easy to employ, they are very sensitive to noise and are generally not very accurate. Very accurate parametric representations can usually be generated for system data, but the process is often involved and requires experience [10].

A very important aspect of the system identification problem is the selection of the proper mathematical, logical or graphical structures to serve as basis for the representation of the target system. Commonly used bases include regressive structures, artificial neural networks, radial basis functions, and state space representation.

In recent times, state space representations have found increased application, especially for the class of linear, time invariant (LTI) systems. Application of state space structures to modeling is based on realization theory and is usually traced back to the work of Ho and Kalman [11].

In the combined discrete-time realization problem, an unknown system with inputs $u_k \in \mathbf{R}^m$ and outputs $y_k \in \mathbf{R}^l$ is represented as the state space structure:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned} \quad (1)$$

where $A \in \mathbf{R}^{n \times n}$, $B \in \mathbf{R}^{n \times m}$, $C \in \mathbf{R}^{p \times n}$ and $D \in \mathbf{R}^{p \times m}$ are unknown system parameters to be determined and correspond to the state matrix, the input matrix, the output matrix, and the direct transmission matrix respectively. Also, u and v are zero-mean, white noise processes with covariance matrix:

$$E\left[\begin{pmatrix} w_p \\ v_p \end{pmatrix}\begin{pmatrix} w_q^T & v_q^T \end{pmatrix}\right] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}\delta_{pq}. \quad (2)$$

The objective is to identify the order, n, of the system, and system matrices A, B, C and D to within a similarity transformation, as well as the matrices $Q \in \mathbf{R}^{n \times n}$, $S \in \mathbf{R}^{n \times l}$ and $R \in \mathbf{R}^{l \times l}$. Techniques by which realization theory is applied to system identification of an unknown system are collective referred to as subspace state space techniques [12].

Subspace techniques possess a number of advantages when compared to other parametric techniques/structures. They do not require non-linear optimization techniques used in classical prediction error method (PEM)/instrumental variable (IM) approaches. They also do not require the imposition of particular canonical forms, a process that is primarily responsible for making classical approaches so difficult to master [13]. From the perspective of the user, another important advantage of subspace approaches is that they can be used for MIMO systems with no added complexities. In fact, the user only needs to specify a single parameter, the order of the system.

Subspace techniques are therefore ideally suited to be applied for automatic model generation in a hybrid online lab.

III. ARCHITECTURE OF A HYBRID ONLINE LAB

A. Overview

A hybrid online laboratory is primarily a remote laboratory but becomes a virtual laboratory when the remote hardware is unavailable or otherwise unusable. This specification leads to two important considerations about

the architecture of a hybrid online laboratory. First, a hybrid laboratory is best viewed as an extension of a remote laboratory,with the new features constituted by the addition of a "lower functionality" software model mode. Second, the central operational requirement in a hybrid online laboratory architecture is the capability to select one of the two possible modes of operation (Figure 2). In the architecture of a sample hybrid online laboratory in Figure 3, the selection logic is performed by the "bandwidth check and decision logic" block.

A hybrid laboratory should ideally use a multi-tiered architecture. Problems with the network link between student and hardware resource usually affects the external link to the Internet, rather than segments of the student's local network. Therefore, for the hybrid laboratory to function properly, the backup software model to be used instead of the remote hardware has to be hosted on the student's side of the internet link. It would be possible to design the student's client (front end) software module to incorporate the bandwidth check and decision logic. It would be less convenient to incorporate the ability to generate software models, and it would be impractical to give the client full access to all previous hardware runs that would be needed by the model generation module. All three functions above would be better performed if the architecture had a third tier.

B. Sample Implementation

The current implementation of the hybrid online laboratory architecture was based on the MIT iLab Batched Architecture (ISA), a three tiered architecture with the topology shown in Figure 4 [1], [5], [16]. A number of modifications were made to the base ISA. Two web service methods were added to the server tier. The first method, *connectionFavorable()*, allows the determination of bandwidth conditions between service broker and server, while a second one, *populateModelData()*, allows the extraction of data from previous experiment runs on the hardware.

No modification was made to the client tier, other than to note that for each laboratory, two distinct clients might now be required: one to be used for the remote laboratory, and another one to be used if the simulation mode is selected. In the sample implementation, a single client was used.

More significant changes were made to the service broker tier. Normally, the ISA service broker includes a database table, containing experiment records of students whose user accounts are managed by the particular service broker. To allow for the possibility of generating models from as large a set of user data as possible, the service broker tier was modified by adding a new database table, *allData,*which contains every experiment record ever run on the server, even those for students managed on other service brokers. This database is populated by invoking *populateModelData()*.

The bandwidth check and decision logic was implemented as a separate module. In the normal service broker, to launch a laboratory client, the student navigates to the appropriate page, and clicks a button labeled "launch". For the hybrid online laboratory, the operation of the "launch" button was modified so that it first invoked the bandwidth check module. This module invokes the *connectionFavorable()* web service. By timing how long it takes to transfer a known amount of data between the two tiers, the bandwidth check module determines the state of the connection between service broker and server. Depending on the outcome, the bandwidth check module launches either the client for the remote laboratory or the alternative client to interact with a simulation. Before doing the latter though, the student is presented with a dialog box informing her of the options (Figure 5).

An automatic model generation (AMG) module was also added to the service broker. Originally implemented in Matlab, the AMG was re-implemented in C# to allow better integration into the service broker. The AMG generated a software model from the data in *allData*. The model was stored in a text file in form of the system matrices *A, B,C* and *D*, and the matrices *w* and *v* (1).

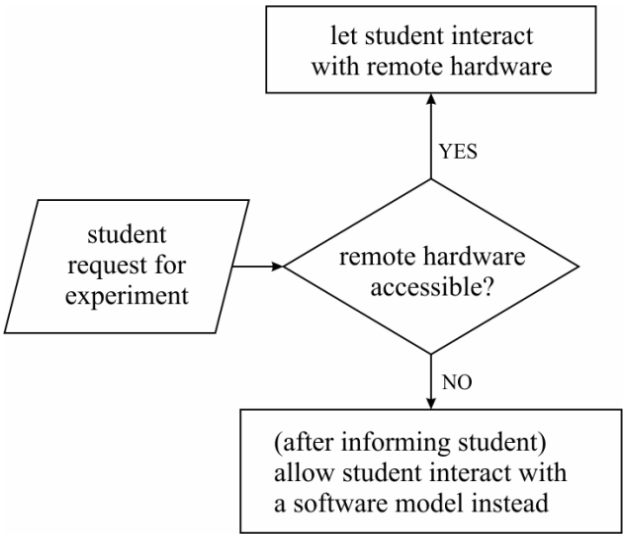


Figure 2. Selection logic at the heart of hybrid online laboratory architecture

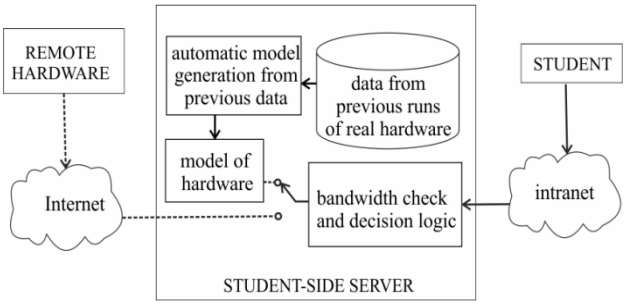


Figure 3. Architecture of a hybrid online laboratory

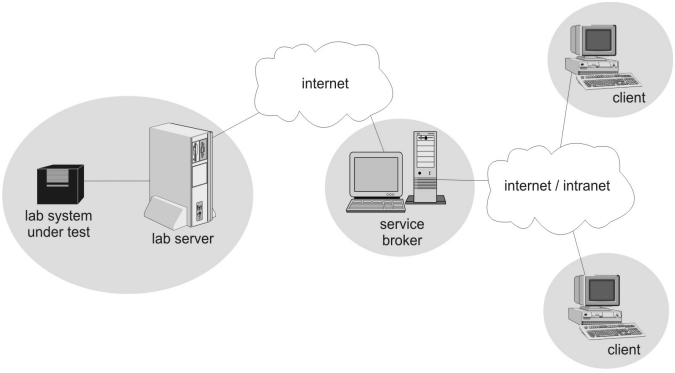


Figure 4. MIT iLab Batched Architecture

Three different subspace algorithms were considered as basis for the AMG: the ISID algorithm of Van Overschee and De Moor[12], the Matlab Central MOESP algorithm of Yi Cao [14] and the Matlab subspace state space system identification(N4SID) function [15]. System matrices for 1250 systems were randomly generated using a Matlab script, and assuming the noise model shown in Figure 6. All terms other than J1 and J2 were defined in (1) and (2). J1 and J2 are summing junctions. Input and output sequences were generated for each system, with the length of each sequence set at 50,000 samples. White noise corresponding to a signal-to-noise ratio (SNR) of 5dB was injected at the output. All 1250 systems were then identified in turn by the three candidate subspace algorithms.

The entire process was repeated for simulated systems with 10dB, 15 dB, and 20 dB, as well as another set with no injected noise (hence, ∞ dB). In all, a total of 7250 systems were used for the assessment, which consisted simply of determination of the simulation errors when each system was identified using each of the three algorithms. Table I provides a breakdown of the simulation errors for each algorithm, per system order, and per SNR. The difference in the performance of the three algorithms is statistically insignificant. The MOESP algorithm was selected, ultimately not because it appears to provide the best results (the differences are insignificant) but because its implementation is very simple and computationally efficient, making it easy to port it to the C# programming language.

IV. TESTING

The sample hybrid laboratory implementation was tested piecemeal. First, the accuracies of automatically-generated models were tested. Secondly, the bandwidth check and decision logic module was tested. All tests were carried out with a hybrid online laboratory based on the Obafemi Awolowo University (OAU) Op-Amp iLab [1], [16].

The OAU Op-Amp laboratory is an iLab built around a simple operational amplifier circuit that can be remotely configured in any of a small number of standard circuits. A simplified version of the SUT is shown in Figure 7. Students interact with the SUT through a simple interfacein a browser.

By utilizing a National Instruments SCXI 1169 switch arraywith 100 rhodium-tipped electromechanical channels, various combinations of switches can be closed remotely to achieve different test circuits. For our tests, two configurations, theinverting amplifier and non-inverting amplifier, were utilized.

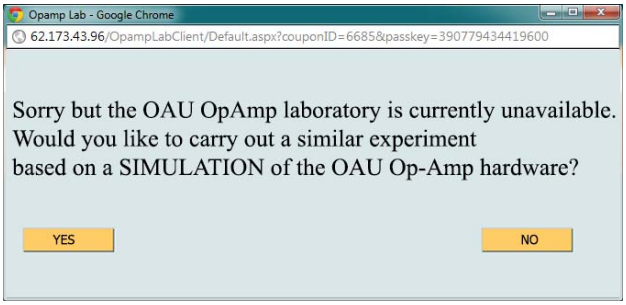


Figure 5. Dialog box showing a message informing student that a simulation-based experiment is being resorted to rather than the original hardware-based one requested

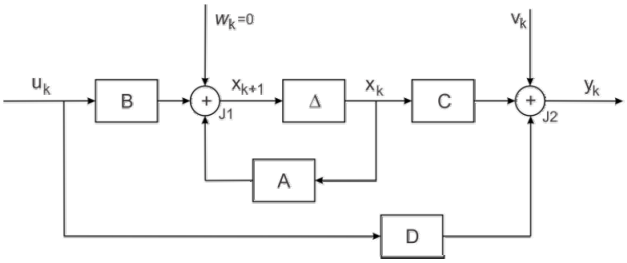


Figure 6. Representation of state space innovative form showing error model

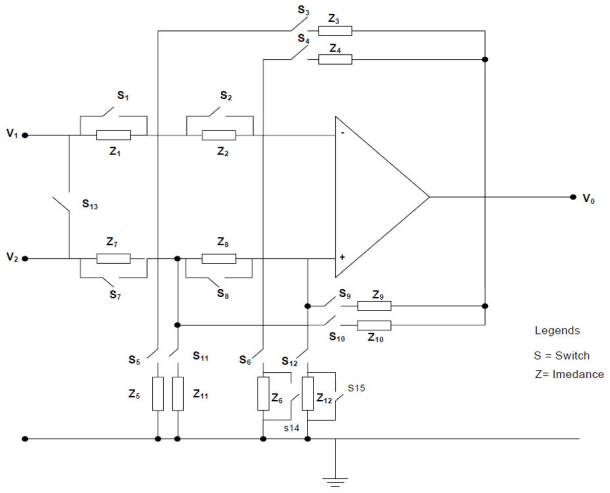


Figure 7. OAU Op-Amp lab system under test

TABLE I.
SIMULATION ERRORS OF ISID, MOESP AND N4SID-CVA SUBSPACE ALGORITHMS FOR FIRST ORDER TO FIFTH ORDER SYSTEMS AND VARYING AMOUNTS OF NOISE

System Order	Simulation Errors (%)														
	ISID					MOESP					N4SID-CVA				
	∞ dB	20 dB	15 dB	10 dB	5 dB	∞ dB	20 dB	15 dB	10 dB	5 dB	∞ dB	20 dB	15 dB	10 dB	5 dB
1st Order	0.4	7.3	11.5	19.1	30.2	0.4	7.3	11.1	19.6	30.2	0.4	7.3	11.1	19.6	30.6
2nd Order	0.0	3.9	7.4	11.9	19.0	0.0	3.9	7.4	11.9	19.1	0.8	4.2	7.4	12.2	19.4
3rd Order	0.0	3.2	4.4	7.5	12.2	0.0	3.2	4.7	7.5	12.2	0.0	3.2	5.2	7.5	12.2
4th Order	0.0	2.0	3.6	4.1	9.0	0.0	2.0	3.6	4.1	8.7	0.0	2.0	4.0	4.5	9.5
5th order	0.0	0.9	3.2	3.0	6.2	0.0	0.9	3.2	3.0	6.2	0.4	0.9	3.4	3.4	7.8

The Op-Amp iLab was selected for this work because it is the simplest multi-tiered remote laboratory to which we had full access. Its simplicity allowed us to identify and focus on the core issues surrounding the implementation of a hybrid online laboratory.

Eight sets of data were collected from the Op-Amp lab. The first four sets were for two inverting amplifier experiments (Datasets 1 and 2) and two non-inverting amplifier experiments (Datasets 3 and 4) run by students of Electronic and Electrical Engineering, OAU during the 2005 academic session. For all four experiments, the input signals were single-tone sinusoids.

For the second set of four experiments, the experiment execution engine was modified to accept a swept-sine input. This was fed into an inverting amplifier configuration for two tests (Datasets 5 and 6) and a non-inverting amplifier configuration for two more tests (Datasets 7 and 8).

Using Dataset 1, the AMG was used to generate a subspace state space model without user intervention. The accuracy of the generated model (for ease of reference, we will refer to the model generated from Dataset X as "Model X") was tested by feeding in the input sequence from Dataset 2 into the model. The resultant output sequence was compared with the actual output signal of Dataset 2. A similar process was repeated for the 2005 non-inverting sets as well as both inverting and non-inverting amplifier swept-sine experiments, resulting in "Model 3", "Model 5" and "Model 7".

Simulation errors were calculated from the differences between the Model 1-generated output sequence (given Dataset 2 input sequence) and actual output sequence of Dataset 2; Model 3-generated output sequence (given Dataset4 input sequence) and actual output sequence of Dataset 4; Model 5-generated output sequence (given Dataset 6 input sequence) and actual output sequence of Dataset 6; Model 7-generated output sequence (given Dataset8 input sequence) and actual output sequence of Dataset 8. The simulation errors are shown in Table II. While the simulation errors for Model 5/Dataset 6 and Model 7/Dataset 8 are seen to be low, a better feel for the implication in time-domain signal characteristics is shown in Figures 8a and b. Figure 8a shows the waveform of the output sequence predicted by Model 5, given Dataset 6 (solid line) along with the signal generated by the real Op-Amp circuit (dotted line). Figure 8b has similar waveforms for Model 7-generated output for Dataset 8 and actual Dataset 8 output.

The bandwidth check and decision logic module was tested by artificially throttling the connection to the Op-Amp server machine using a bandwidth manager. At different bandwidth settings of the bandwidth manager, the bandwidth check module was tested by clicking on the "launch" button and seeing whether the student was prompted that a simulation would be run instead. Table III shows a summary of the test.

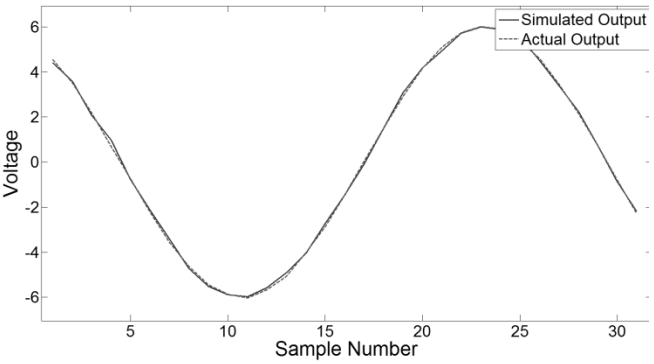
V. DISCUSSION

The results generated by Models 5 and 7 (both swept-sine experiments) are seen to be very close to results generated from real hardware while the 2005 experimental records led to less accurate models.

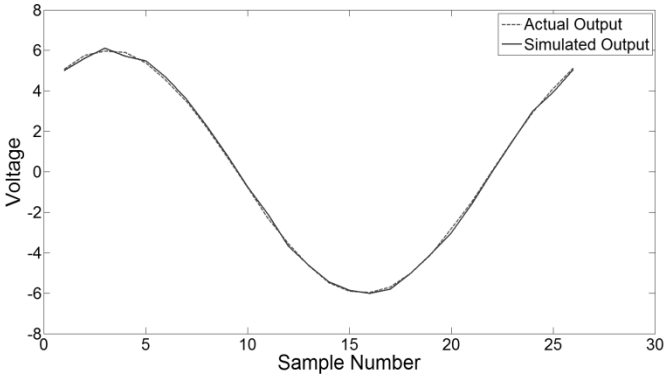
There are two likely reasons for the inaccuracies of the 2005 datasets. First, the input/test signals were monotone sinusoids, which run counter to a well-established tenet in system identification: systems must be excited with signals that can expose a good cross-section of system dynamics [10].

TABLE II.
SIMULATION ERRORS FOR REAL OUTPUTS OF DATASETS 2, 4, 6 AND 8 VERSUS OUTPUTS PREDICTED BY MODELS 1, 3, 5 AND 7

	Model 1 (Dataset 2)	**Model 3 (Dataset 4)**	**Model 5 (Dataset 6)**	**Model 7 (Dataset 8)**
Simulation Error (%)	11.21	8.13	0.35	0.41



(a)



(b)

Figure 8. (a) Output sequence predicted by Model 5, given Dataset 6 (solid line) along with the signal generated by the real Op-Amp inverting amplifier circuit (dotted line). (b) Output sequence predicted by Model 7, given Dataset 8 (solid line) along with the signal generated by the real Op-Amp non-inverting amplifier circuit (dotted line).

TABLE III.
CLIENT VERSION LAUNCHED FOR DIFFERENT BANDWIDTHS OF THE INTERNET LINK TO THE SERVER

Bandwidth	**Client launched by service broker: remote laboratory or simulation?**
"Unlimited" (~1Mbps)	Remote laboratory client
512 kbps	Remote laboratory client
128 kbps	Remote laboratory client
64 kbps	Remote laboratory client
32 kbps	Remote laboratory client
16 kbps	Simulation client
8kbps	Simulation client

Secondly, the Op-Amp laboratory normally stores 400 data points, meaning the number of samples used for the system identification process was far lower than the recommended range of 10^3 to 10^4 samples [6]. Datasets 5 to 8 on the other hand, all had 12,500 samples each of multi-tone sinusoids, leading to better results.

So can the results above, generated by and for such simple systems, be extended to a larger range of remote laboratory SUT? We believe the answer is yes. Doubtless, the results should extend to any single-input-single-output static, first or second order system operating in linear mode with excellent signal to noise ratio. Extension to other classes of systems would need to be preceded by a more rigorous set of tests.

It should be reiterated that there is considerable value to be derived from getting an automatic model generation system like this to work for as wide a range of remote laboratories as possible. Deployed with multi-tiered remote laboratory architecture, the generated models can be stored on a system near the student (or on the student's system itself). This allows student to interact with a software model when the hardware is unavailable.

Certainly, the aim here is not to argue that simulated results will always be identical to those from hardware; what we propose is very *much a half-bread-is-better-than-none* solution. That is, when the real hardware is unavailable, rather than sending the experiment-seeking student away, our proposed system would generate simulated results. The simulated results would ideally be very close to those generated by a real system. Even when they are not so close however, they should provide some pedagogical value (assuming there are no errors that propagate erroneous notions about the system under test). It would certainly be interesting to carry out a study where the perception and performance of made to use the simulated alternative is compared with those who interacted with the real hardware.

Whatever the outcomes of such a study though, it is clear that the hybrid online architecture would provide a gentler degradation in the laboratory experience of students when the backend hardware of a remote laboratory is unavailable.

Although the models in the hybrid online lab described in this paper were generated automatically, automatic model generation is not a requirement for every hybrid online laboratory. Indeed, a remote laboratory could be converted to a valid hybrid online laboratory simply by integrating a bandwidth check module which can redirect the student seamlessly to a virtual laboratory client offering similar learning objectives.

There is however no gainsaying the fact that an ideal online laboratory would feature automatic generation of accurate models, thereby eliminating the need to search for similar virtual laboratories to redirect students to. Further research can establish that results close enough to real results to be pedagogically useful can be obtained for certain classes of systems. This implies that some work still needs to be done on various parts of the framework. However, we believe our current results provide excellent basis for optimism.

VI. CONCLUSION

We have made a case that the over-dependence of remote laboratory systems on internet connectivity conditions is unwholesome. We propose that rather than having remote resources completely unavailable in the event of a network outage, students can be referred to an accurate model of the system under test instead. Although the experience may not be exactly the same as remote the real hardware, we have results that suggest that at least for some systems, the results may be close enough to be useful. The knowledge that a remote laboratory would offer some service of value to students irrespective of the current condition of an institution's internet connection should make rlabs even more attractive.

We confirmed our initial ideas by automatically developing models for a simple remote laboratory SUT and showed that with care taken in selecting input signals, good results can be obtained for real life systems.

Some work needs to be done to expand the idea of hybrid online laboratories further, but our results indicate that they could be very useful in improving the overall student experience with online laboratories.

ACKNOWLEDGMENT

The Authors would like to acknowledge the support of the MIT Center for Educational and Computing Initiatives. In particular, we thank J. del Alamo, Jud V. Harward and Jim Hardison for providing some data for testing our work. We also thank Andrew Watchorn of National Instruments for providing some of the data acquisition devices used.BabatundeIsola helped with some of the programming.

REFERENCES

[1] J. Del Alamo, J. Hardison, G. Mishuris, L. Brooks, C. Mclean, V. Chan, andL. Hui, "Educational experiments with an online microelectronics characterization laboratory", *Proceedings of International Conference on Engineering Education, Manchester*, 2002.

[2] S.D. Bencomo, "Control learning: present and future", *Annual Reviews in Control*, vol. 28, 2003, pp. 115–136 http://dx.doi.org/10.1016/j.arcontrol.2003.12.002

[3] G. Carnevali and G. Buttazzo, "A Virtual Laboratory Environment for Real-Time Experiments", *Proceedings of the 5th IFAC International Symposium on Intelligent Components and Instruments for Control Applications (SICICA 2003), Aveiro, Portugal*, July 2003,pp. 39-44.

[4] L. D. Feisel and A. J. Rosa, "The Role of the Laboratory in Undergraduate Engineering Education", *Journal of Engineering Education, January 2005*, pp 121-130.

[5] D. Olowokere, K. P. Ayodele, L.O. Kehinde, O. Jonah, T.O. Ajayi, O.O. Akinwunmi, "Realistic Looking Interfaces: In Search Of The Best Ergonomic Metaphors For Remote And Virtual Laboratory Interfaces", *Proceedings of the ASEE Annual Conference and Exposition,* June 2008.

[6] M. Viberg, B. Otterson, B. Walberg, and L. Ljung, "Performance of subspace based state space identification systems", *proc. 12th World Congress International Federation of Automatic Control*, Vol. 7,1993 pp. 369-372.

[7] M.A. Flores, "Teachers' views on recent curriculum changes: tensions and challenges", *The Curriculum Journal*. Vol. 16, No. 3, September 2005, pp. 401–413 http://dx.doi.org/10.1080/09585170500256479

[8] K. Jona, R. Roque, J. Skolnik, D. Uttal, and D. Rapp,"Are Remote Labs Worth the Cost? Insights from a Study of Student Perceptions of Remote Labs" *International journal of Online Engineering,*Vol 7, No 2, 2011, pp. 48 – 53.

[9] L. Ljung, *System identification: theory for the user*, Prentice-Hall, 1987.

[10] T. Soderstrom, and P. Stoica, System *identification*, Prentice Hall: Inc., Englewood Cliffs, New Jersey, USA.1988

[11] B. L. Ho and R. E. Kalman, "Effective construction of linear state-variable models from input/output functions," *Regelungstechnik*, vol. 14, no. 12,1966, pp. 545–548.

[12] P. Van Overschee, and B. De Moor, *Subspace Identification for Linear Systems*, Kluwer Academic Publishers.1996 http://dx.doi.org/10.1007/978-1-4613-0465-4

[13] T. Katayama, *Subspace Methods for System Identification*, Springer, 2005.

[14] http://www.mathworks.com/matlabcentral/fileexchange/19728-multivariable-subspace-identification-moesp

[15] P. Van Overscheeand B. De Moor "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems", Automatica, 30, 1994, pp. 75-93. http://dx.doi.org/10.1016/0005-1098(94)90230-5

[16] K.P. Ayodele, L.O. Kehinde, O.P. Jonah, O.Ilori, E.O.B.Ajayi, and O. Osasona, "Development Of An Operational Amplifier Virtual Laboratory Based On ILab Architecture And NI Elvis", Proceedings of the ASEE Annual Conference and Exposition, June 2008.

AUTHORS

K. P. Ayodele is with the Department of Electronic and Electrical Engineering, Obafemi Awolowo University, Ile-Ife 220005, Nigeria. (e-mail: kayodele@oauife.edu.ng).

L.O. Kehinde is with the Department of Electronic and Electrical Engineering, Obafemi Awolowo University, Ile-Ife 220005, Nigeria. (e-mail: lkehinde@oauife.edu.ng).

O. A. Komolafe is with the Department of Electronic and Electrical Engineering, Obafemi Awolowo University, Ile-Ife 220005, Nigeria. (e-mail: okomolaf@oauife.edu.ng).

This work was supported in part by the Carnegie Corporation of New York under an iLab Junior Fellowship grant. Manuscript received 18 June 2012.Published as resubmitted by the authors 23 October 2012.