POLLACK PERIODICA
An International Journal for Engineering and Information Sciences
DOI: 10.1556/606.2018.13.3.9
Vol. 13, No. 3, pp. 83–93 (2018)
www.akademiai.com

DEVELOPMENT OF A VR CAPABLE VIRTUAL LABORATORY FRAMEWORK

[1] Tamás BUDAI, [2] Miklós KUCZMANN

[1,2] Department of Automation, Faculty of Mechanical Engineering, Informatics and Electrical Engineering, Széchenyi Isván University, Egyetem Tér 1., H-9026 Győr, Hungary
e-mail: [1]budai.tamas@sze.hu, [2]kuczmann@sze.hu

Received 3 January 2018; accepted 16 May 2018

Abstract: The aim of this paper is to introduce the design steps and implementation details of a system, which combines modern web-technologies and open-source simulation software to create a virtual laboratory framework. In order to validate the functionality of this framework and to demonstrate its capabilities, a classic experiment from the field of control theory was implemented; the inverted pendulum. In this experiment, a simulated controller keeps the rod of the pendulum in an upright position. Users can change different parameters of the model and then test the impact of these changes in a very intuitive and interactive way, by applying force to the pendulum model in 3D and observing the behavior of the controller.

Keywords: Virtual laboratory, e-Learning, web, Control theory, Inverted pendulum

1. Introduction

In the past few decades, online education or *e-Learning* has matured and now it is a vital part of higher education worldwide. Early e-Learning systems were simple online libraries, where static content composed from course materials were available to students. These systems were and are mostly useful for the transfer of pure lexical knowledge e.g. history or literature.

In other disciplines like engineering sciences, where most of the courses discuss and experiment on complex mathematical or physical phenomena, interactive demonstration techniques yield greater didactic power than static content. As the media-consumption habits of the newer generations are shifting towards interactive online content from classic mediums like books, articles and television, education must keep up with this transition. It must incorporate new technologies to enrich the learning experience of students.

The rapid development of InfoCommunication Technology (ICT) and specifically web-technologies enabled the possibility to design interactive online learning systems [1], [2]. In the field of engineering sciences like electronics, mechanics and chemistry where laboratory exercises and measurements are a vital part of the learning process, different institutes and universities with the common goal to transform classic hands-on experiments to interactive online materials developed many systems.

Virtual and remote laboratories in engineering disciplines have numerous advantages over classic physical experimentation and laboratory courses:

- *Accessibility*: students can log in and use the virtual laboratory anytime from anywhere, there are no lab open hours and there is no need to travel to the laboratory in person;
- *Self-paced learning*: Students can go over and repeat parts of the exercise as many times as necessary to better understand the subject;
- *Easy evaluation and homework assignment*: students can take quizzes and answer questions about the discussed topic in the same environment and teachers can assign grades based on quiz results.

There are two common types of these systems; virtual and remote laboratories [3]. These share the common concept of an online service that is available to students either via a website or via an application anytime from anywhere, but has an important difference in the structure. Sometimes in literature the two terms are used inconsistently, therefore it is important to describe the differences between them.

1.1. Remote laboratories

In a *remote laboratory* an existing physical setup of the experiment is available (usually it is housed in the corresponding laboratory of the university) and it is observed and controlled remotely by the students with the help of networking elements like webcams and network-enabled measurement equipment. *Fig. 1* shows an example remote laboratory setup.

The main advantage of these systems that the laboratory is backed by actual measurement equipment, often similar to the ones that students encounter during hands-on laboratory courses. Therefore, all real-life positive and negative side effects of the equipment are present during the exercise e.g. non-linear response to room temperature changes. This yields important practical knowledge as it could help students to comprehend the importance of the factors that could influence the measurement results. The biggest disadvantage of a remote system is that it is *single-user* i.e. only one student can experiment on one setup at a given time. This requires scheduling and makes the system less flexible.

1.2. Virtual laboratories

In a *virtual laboratory* the physical behavior that is the subject of the experiment or measurement is entirely simulated by software, therefore no actual hardware is required beside the computer that runs the simulation. This has numerous advantages over classic hands-on and remote laboratories, including:

- *Significantly reduced operational cost*: a virtual laboratory system does not require any laboratory equipment, which in itself yields numerous advantages: there are no parts that could wear out or get damaged during use and there is no physical space required to store equipment;
- *Concurrency*: many students can use the virtual laboratory at the same time;
- *Flexibility*: since the virtual laboratory is based on simulation, physical constants like gravity can easily be modified and it is possible to conduct examination under extreme circumstances like high temperature or pressure.

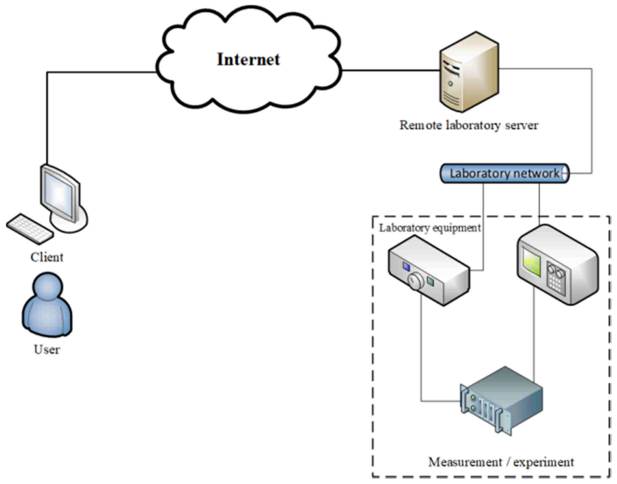

Fig. 1. A remote laboratory setup

The only real disadvantage is that all factors that are important in the scope of the experiment must be taken into account during the simulation, which is often hard to implement and requires complex processing [4], therefore virtual laboratory solutions tend to have high system requirements. *Fig. 2* shows an example of a virtual laboratory setup.

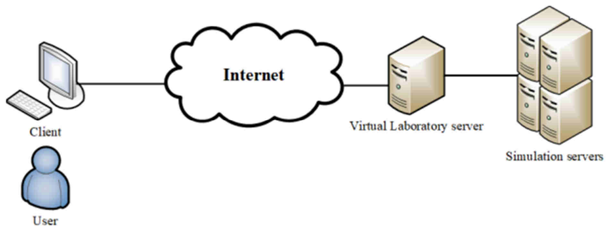

Fig. 2. A virtual laboratory setup

2. Current solutions

In the past years, several research projects were born with the aim to implement remote and virtual laboratories, but these have numerous disadvantages. Many of these are relying on very outdated and often proprietary software or hardware components, thus reducing the feasibility of these systems in education. As [3] shows many universities and institutes have designed their own systems, but there is a lack of unity, the re-use of software components are minimal, therefore a great amount of development work is wasted because of the repeated effort on implementing the same functions among these solutions.

As [5] shows there are numerous combinations of programming languages used both on the server and the client side, but the best way to create easy-to-use virtual laboratories is to use a combination of web technologies, like Asynchronous JavaScript and eXtensible Markup Language XML (AJAX) and HyperText Markup Language 5 (HTML). Since web technologies are a constantly evolving topic, revisiting the currently available implementations was necessary to discover possible ways of improvement.

2.1. Virtual labs

The virtual labs [6] project funded and backed by the Indian government, implements a series of virtual laboratories in the field of engineering sciences. Multiple universities and institutes are participants of the program, which aims to provide distance-learning possibilities for engineering students in the developing country. This project aims to develop a framework that enables the usage of diverse software and hardware technologies to create remote and virtual laboratories. In practice, most of the laboratories that are currently available were implemented either with the help of Adobe Flash or with the LabVIEW runtime, where the application executable must be downloaded and used in conjunction with the web-materials.

The platform features a well-defined structure of the end-user interface of a virtual laboratory, including theory notes, quizzes, workbooks, video lectures and modeling. I would like to define a similar structure for the new system.

2.2. Weblab Deusto

The aim of the WebLab Deusto [7] project is to provide a framework for remote laboratories. However, all of the implemented laboratories require hardware; the system is worth evaluation because it uses AJAX to present a web-based Graphical User Interface (GUI) to the end-user. It also supports interoperation with LMSes like Moodle. It is open-source and freely available, therefore different implementation details can be examined. One of the goals of my work was to design a system with similar features like Weblab Deusto, without the hardware requirement.

2.3. Summary

Most of the currently available solutions have their own user authentication and authorization schemes to manage users and laboratories. As [2] points out this is redundant because there are multiple Learning Management Systems (LMS), which are mature and already implemented this functionality. A good solution would be to integrate the virtual laboratory system into one of these and let the LMS handle management tasks.

After test-driving all of the above-mentioned solutions, it was apparent that only the virtual labs project feature purely virtual laboratories, but all of these are realized in LabVIEW and require the LabVIEW runtime engine to run. Although LabVIEW is a great tool for building simulations, it is proprietary and these stand-alone executable are unusable on mobile platforms. All of the other solutions are remote laboratories, realized with the help of actual hardware components.

3. Requirements of the new system

The aim of this work was to design a new, general, multifunctional virtual laboratory system, based on simulations implemented in Scilab [8]. It must be able to be used for demonstration purposes in the classroom to enhance the lectures held by colleagues at the department and function as a base of online virtual laboratory courses. It must be designed in a way that enables easy adoption of currently implemented Scilab codebase.

Although the main goal was to design a system that will be used in electrical engineering courses, there was a motivation to generalize it enough to be applicable in different fields like mechanical engineering and every other field where Scilab (or Matlab) simulations are applicable in classes.

After revisiting current solutions, in order to enumerate the possible use-cases and to define the required features of the new system, colleagues at the department were interviewed. Based on the current solutions and taking into account the requests and comments from colleagues, the requirements of the new system was determined and defined:

- *Secure*: the communication between the server and the client should be encrypted;
- *Accessible*: the web User Interface (UI) must use modern web technologies (HTML5 and JavaScript) and must work without additional browser plugins. This requirement also ensures that the client is Operating System agnostic;
- *Scilab support*: the simulations powering the virtual laboratory are going to be written in Scilab;
- *Credible*: the simulations must produce the exact same results to the same input, regardless of the client and it must be assured that there is no way to tamper with the simulation process from the client side. In addition, the activity of the clients must be logged;
- *Flexible*: target students on the go and allow them to use their smartphones and tablets for simple tasks on a web user interface and provide a more

sophisticated, Virtual reality (VR)-compatible 3D environment for laptops and desktop machines;
- *Modular*: built from elements that could be re-used in different laboratory courses;
- *Scalable*: the system must be able to support a large number of concurrent students;
- *Possible LMS integration*: interface with existing e-Learning solutions to handle tasks like user and laboratory management.

4. System design

After the establishment of requirements, the new system design was proposed. The most important design decisions based on currently available implementations, the requirements described in Section 3 and the building blocks of the new system are shortly described below.

The proposed system is based on the *Client-Server model*, with the following roles on the two sides.

Roles of the client:

- simulation control: input parameters, commands;
- data display: numerical listing, graphing and interactive object manipulation.

Roles of the server:

- request processing: listen and wait for client connections and commands;
- execution of simulation scripts;
- gateway: forwarding of input commands and output results between scilab instances and clients.

The first and most important design choice was to keep the simulation processes on the server side. This has two important advantages. First, since the server carries out the actual calculations, the system requirements of the client are minimal; therefore, students can use the new system even on a low-end smartphone. Second there is no way to tamper with the calculation process from the client side i.e. it is impossible to alter simulation results, therefore it will be possible to assign grades to students based on their simulation results.

The second important decision was to provide two different user interfaces to support a wide range of clients: a 3D, VR-enabled one for Personal Computers (PCs) and laptops with decent graphic cards (*desktop client*) and - as a fallback solution - a lightweight, web-browser based one for devices with low graphical capabilities like mobile devices and tablets (*mobile client*). The two interfaces must share a similar graphical design for elements like buttons and input controls to eliminate confusion and reduce the learning curve of the system.

The programming language used for the web-client must be JavaScript, because it is supported out of the box in all modern web browsers and the developer community provides a wide variety of frameworks for it. This can significantly reduce the amount of work necessary to implement the planned features. However the client-server architecture allows the usage of completely different programming languages, it would be beneficial to use the same JavaScript on the server-side if possible.

The desktop client is based on MaxWhere [9], a unique VR framework (engine) that is programmable through JavaScript and allows for building VR applications in which the conventional web 2D content is fully associative with the VR world. MaxWhere provides a JavaScript Application Programming Interface (JS API) called Where Object Model (WOM) that implements similar functionality for VR as the Document Object Model (DOM) does in the web browsers.

It provides the user with a rich, VR-ready 3D environment with powerful presentation tools and an embedded browser engine. The latter allows the user to load arbitrary webpages into the floating browser windows placed across the virtual environment. The embedded chromium engine allows the user to use all modern web technologies like HTML5 and JavaScript in these floating browser windows. This feature could be very useful for presentations and demonstrations, because the user can use these browser windows in a variety of ways. It is possible to display static content for materials like syllabuses and lecture notes or rich multimedia materials like video and audio notes and animations or to interact with the user with dynamic webpages and Real Time video-Chat (RTC). By using these features together with the numerical simulation, a self-contained or encapsulated learning material can be produced.

The Scilab [8] software package and programming language is a free and open-source alternative of Matlab [10] with the aim to provide a similar tool-set for free. Scilab is gaining popularity in recent years because its feature-set matured enough to be practically usable. The department where I work is also committed to use it instead of Matlab in our courses. This means that colleagues have some expertise with it already. Plus it is a requested feature to enable interactive Scilab script editing in the browser to encourage students to use Scilab and to be able to hand out homework that involves Scilab programming in the future. Also there are successful examples [11] on usage of Matlab in virtual laboratories. These are the main reasons why I wanted to use it in my work.

Node.js [12] is an asynchronous, event driven JavaScript runtime, a building block of modern web applications. Node handles concurrent networking in a different way than other modern languages; while most of these are using threads or sub-processes, node uses the *event loop* model. This means that after starting a node application, it enters into an *event loop* and waits for events to occur. Each time a new connection is established a so-called *event callback* is fired and the actual work what was requested is handled by this callback.

Because of this, node applications require very few execution cycles on the Central Processing Unit (CPU-time), and only generate load when there is a callback to process i.e. on user generated events, therefore it is very straightforward to develop scalable web applications [13] in node.js.

It was set up as a requirement to implement the client in a web browser without any additional plugins. The only problem was that the system relies on duplex, real-time data communication and the transport technology of the web, HTTP was not designed to handle this. Originally, it was designed to support the *request-response* model, where the client sends a request and the server answers with a response. There is no way to the server to initiate communication on its own. During the last few years, several techniques were developed to overcome this limitation. The most commonly used is a technique called HTTP long polling. The client initiates a request, but the server only responds to it when data is available. After the response, the client initiates another request immediately and the whole procedure repeats itself, emulating a two-way communication channel. Although this works well in most cases, a new protocol called WebSocket [14] was developed to support real-time communication.

Deepstream.io [15] is 'the open real-time server', an open-source, scalable and secure data-streaming server written in JavaScript, using the Node.js runtime. It supports both HTTP long polling and WebSockets via socket.io and it has built-in support for authentication, encryption and a client also written in JavaScript that could be used in a browser so it fits well into the proposed system.

With the building blocks described above, the new system could be detailed and interfaces could be determined. *Fig. 3* shows the detailed architecture of the new system.

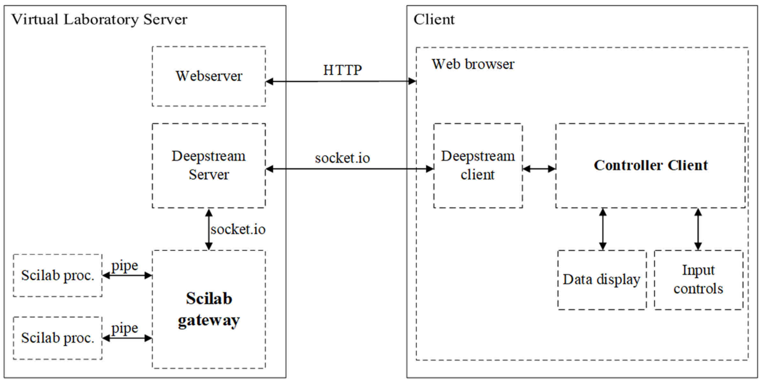

Fig. 3. Architecture of the new system

5. A practical example: The inverted pendulum

After the design was complete an example virtual laboratory course was developed to ensure the usability and demonstrate the capabilities of the new system.

A classic problem in control theory education is the control of an inverted pendulum. The most common realization of the system consists of a cart that can move along a fixed path, with a rod attached to it by a rotary joint. In the simplest setup the rod is only allowed to rotate along one axis, limiting the pendulum to one degree of freedom. Since the center of mass is above the pivot point of the rod, the inverted pendulum is inherently unstable and the controller must balance it continuously by moving the cart to keep the rod in an upright position. *Fig. 4* shows the model of the cart and rod system. The cart has a mass *M*, the attached rod has a length *l*, and a mass of *m*. Often the mass of the rod is represented as a concentrated point, but it could be modeled as a homogenous mass distributed evenly along the whole rod. The external force *F* and the gravitational force *g* cause the rod to deviate from the ideal vertical position and the controller must influence the velocity of the cart to compensate and keep the rod from falling down. In most implementations, the controller is either output a signal that can drive electric motors on the wheels, or the body of the cart is fixed to a linear actuator, which is able to act on the cart with the desired force.

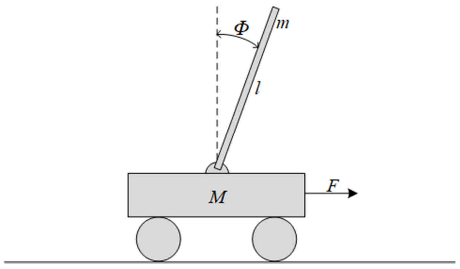

Fig. 4. Model of the inverted pendulum

To demonstrate the capabilities of the new system, an existing Scilab implementation of the pendulum was used and integrated it into a virtual laboratory for demonstration purposes.

The simulation is based on an ordinary differential equation, which is presented in discrete form and solved in time-steps.

The equation that describes the behavior of the cart and pendulum system is realized as a function. This function is called and the current state of the cart is calculated in a *while* loop for each time step, with a time resolution of 0.25 seconds. While this resolution is considered coarse in numerical simulation, it produces adequate results between the boundaries of the experiment and even lower end machines can calculate the result of the function in the current loop under 50ms, therefore even with the additional latency that is introduced by the network the simulation can be executed in a quasi-real-time manner.

The original Scilab script contains the helper function *Display*. This function handles the simple graphical display of the cart and charting of the force and rod angle variables over time. It is important to note that previous values of the displayed variables must be aggregated, because in every time step, the whole graph is redrawn. The problem with this approach is that all the values must be stored in memory; therefore simulation time is limited by the amount of available memory. This problem is circumvented in the new system, by only aggregating the results for a limited amount of previous time steps, creating a sliding time-window. *Fig. 5* shows a screenshot from the desktop client.

The user can change the following parameters before starting the simulation:

- The mass of the cart;
- The mass of the ball on the rod;
- The position of the ball on the rod;
- The starting angle of the rod.

It is important to note that the values of these parameters are limited in order to avoid situations where the simulation can became unstable. It is considered to allow the user to change more parameters in future versions.

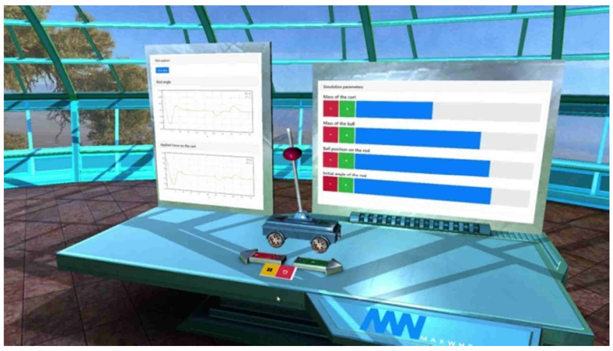

Fig. 5. The desktop client, running in MaxWhere

6. Conclusions

During the course of this work, a system was designed and implemented that could help to enrich the education of future engineers, by providing an interactive learning space, where they can experiment and learn concepts in the same way that they do in the traditional laboratories today. In order to evaluate the feasibility of the new system, a series of tests must be conducted with lecturers and students, in order to identify the key factors that can be improved in future versions.

Acknowledgements

The research leading to these results has received funding from the project EFOP-3.6.1-16-2016-00017 of the Hungarian Government.

References

[1] Berecz A., Ágoston Gy. The Hungarian adaptation of ilias web-based L(C)MS and its use in information education with a special regard to services tailoring, *Pollack Perodica*, Vol. 2, 2007, pp. 71–84.

[2] Gomes L., Garcia-Zubia J. (Eds.) Advances on remote laboratories and e-learning experiences, *IEEE Industrial Electronics Magazine,* Vol. 2, No. 2, 2008, 45–46.

[3] Ruiz E. S., Martin A. P., Orduna P., Martin S., Gil R., Larrocha E. R., Albert M. J., Diaz G., Meier R., Castro M. Virtual and remote industrial laboratory: Integration in learning management systems, *IEEE Industrial Electronics Magazine,* Vol. 8, No. 4, 2014, pp. 45–58.

[4] Kuczmann M., Budai T., Kovács G., Marcsa D., Friedl G., Prukner P., Unger T., Tomozi Gy. Application of PETSC and other useful packages in finite element simulation, *Pollack Perodica,* Vol. 8, No. 2, 2013, pp. 141–148.

[5] García-Zubia, J., Orduna P., Lopez-de-Ipina D., Alves R. G. Addressing software impact in the design of remote laboratories, *IEEE Transactions on Industrial Electronics* Vol. 56. No. 12, 2009, pp. 4757–4767.

[6] Virtual Labs, An initiative of Ministry of Human Resource Development under the national mission on education through ICT, http://vlab.co.in/ (last visited 21 December 2017).

[7] Weblab Deusto website, http://weblab.deusto.es/ (last visited 21 December 2017).

[8] About Scilab, http://www.scilab.org/scilab/about (last visited 21 December 2017).

[9] MaxWhere website, http://www.maxwhere.com/ (last visited 21 December 2017).

[10] MathWorks Matlab website, http://www.mathworks.com/ (last visited 21 December 2017).

[11] de Magistris M. A Matlab-based virtual laboratory for teaching introductory quasi-stationary electromagnetics, *IEEE Transactions on Education*, Vol. 48, No.1, 2005, pp. 81–88.

[12] About Node.js, https://nodejs.org/en/about/ (last visited 21 December 2017).

[13] Gu. X. F., Yang Le Y., Wu S. A real-time stream system based on node.js. *11th International Computer Conference on Wavelet Actiev Media Technology and Information Processing*, Chengdu, China, 19-21 December 2014, Papper 7073454.

[14] Pimentel V., Nickerson B. G. Communicating and displaying real-time data with WebSocket, *IEEE Internet Computing*, Vol. 16. No. 4, 2012, pp. 45–53.

[15] Deepstream.io, https://deepstream.io/ (last visited 21 December 2017).