Low production cost virtual modelling and control laboratories for chemical engineering students

J. A. Rossiter *

* *Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, S1 3JD, UK (e-mail:j.a.rossiter@sheffield.ac.uk*

Abstract: This paper gives a brief review of work on virtual and remote laboratories along with a critique of their strengths and weaknesses. This is used as a motivation for a proposed method for producing virtual laboratories which, while relatively crude in comparison with professional alternatives, are much cheaper and faster to produce and thus can be created using the skill set and time of normal academics. Some examples and the coding processes are demonstrated.

Keywords: Virtual laboratories, staff efficiency, student engagement, independent learning.

1. INTRODUCTION

For the sake of brevity, this paper will assume the high value of laboratory activities within a chemical engineering education and move immediately to consideration of how this might be delivered. It is well recognised that the infrastructure costs (space, equipment and technical support) combined with student timetabling challenges (that is ensuring timing close to corresponding lectures) are some of the reasons why, in practice, engineering undergraduates have lower exposure to hardware than would be desirable. As a consequence, Universities (e.g. Abdulwahed 2010, Qiao et al. 2012, Rossiter et al. 2011) have sought to improve student access to authentic activities by introducing pseudo laboratory activities whereby students interact with a realistic simulator or even with real hardware via a web interface. Moreover, a further advantage of software/computer based activities is that they enable students more freedom for independent learning activities.

1.1 Background on remote and virtual laboratories

The focus of this paper will be on so called virtual laboratories, that is laboratories which are based on simulations, perhaps of high fidelity process models, rather than on hardware. The focus on virtual laboratories as opposed to say remote laboratories (RL) (Dormido et al 2012) is for a few simple reasons:

- Remote laboratories are known to be costly and time consuming to produce and maintain and moreover require staff with significant expertise in many areas such as database use and web scripting which are outside their normal knowledge (Chen et al. 2010, Vargas et al. 2011). Most departments do not have the expertise or resource to support this activity effectively.
- Remote laboratories have limited accessibility in practice due to the queueing required by students combined with the possibly slow time constants of chemical processes. This mitigates against the intended benefits of 24/7 accessibility for students, especially with large classes.

Nevertheless, the design of an accessible virtual laboratory (VL) is fraught with equally many challenges and in particular the fact that web accessibility requires significant software skills from the author, in addition of course to understanding and implementing any pedagogical requirements. Some of the examples in the literature such as (Cameron 2009, Goodwin et al 2011) are excellent virtual environments on which to study chemical engineering, but the creation of such artefacts is not achievable for most academics indeed the authors of those environments assumed that departments would pay a substantial annual license fee for students to access their simulators. Even what might be considered an accessible (essentially free) web based system and well used software environment (Easy Java Simulation (EJS)) is non trivial to code accept for elementary scenarios (de la Torre 2013, Fabregas 2011, Perez et al. 2011, Guzman 2006).

1.2 Paper motivation and ethos

It is well accepted that high quality virtual/remote laboratory activities are a significant enhancement to the student learning environment. One obvious and perhaps less well publicised use is for laboratory preparation and post activities (e.g Abdulwahed 2010, Rossiter et al. 2014) which reinforce key learning outcomes because the environment is an effective emulator (Memoli 2011) of the real hardware set up in the laboratory.

- Students can use RL/VL in order to anticipate the activities and concepts they will encounter with the hardware and thus to support preparation of key computations, notes, algorithms and concepts they enable them to make the most effective use of their time on the equipment.
- After the hardware laboratory, students can use the RL/VL to test any hypothesis not completed success-

fully, forgotten or recognised during the write up and reflection phase.

Consequently, this paper takes the motivation for RL/VL as a given and instead focuses on a different issue. Specifically, this paper takes the following premise:

(1) Most academic staff do not have the time, support or departmental infrastructure to develop robust web accessible remote or virtual laboratories.
(2) Where funding is available and there is a tight synergy with the course learning outcomes, departments may choose to purchase licenses for commercial simulators (indeed the authors department used the Goodwin 2010 resource for a few years).
(3) In practice, the bespoke nature of each departments course/module design and learning outcomes mean that the requirements for laboratory activities are rarely met closely by off the shelf resources and thus there is a need to do some in-house development.

Herein lies a major challenge. Academic staff may wish to develop RL/VL activities to support student engagement and independent learning, but they lack the expertise or support required to produce a high quality and fully web accessible resource. Consequently an alternative solution is required.

The author believes in pragmatic solutions, that is, better a simple solution that can be implemented tomorrow than a perfect solution in 2-3 years (if ever). Moreover, simple and cheap solutions often have the advantages of being equally cheap and easy to modify should the departmental requirements change whereas expensive resources are often equally expensive and difficult to modify. The reality of most student learning, lectures, tutorial classes and indeed real industrial processes are that they are not manicured environments. Rather lecturers often mumble, make mistakes in lectures and correct themselves (or not), write illegibly and so forth, and despite all this students may still comment that the lecture course was well presented, clear, enjoyable, etc. In summary, a VL/RL does not need to be coded and presented to commercial standards in order to be an effective learning tool.

1.3 Proposal for virtual laboratory development

In summary, this paper proposes a pragmatic approach to virtual laboratory development, that is an approach with a typical academic could achieve with relatively little coding expertise and, more importantly, relatively little time. The sacrifice of being able to produce learning resources quickly is a reduction in accessibility, that is the resources may no longer be web accessible. However, this need not be an impediment in that the real requirement for accessibility is that the students can access and use the resources 24/7, that is, as and when they need too; being on the web is rather secondary and could even be an impediment where wireless or broadband is unreliable. The author favours the use of MATLAB software for the development of VLs for 3 major reasons.

(1) Within his University (and indeed many Universities) there is a site license so students can guarantee access to the software and indeed get a version for their own laptops should they prefer.
(2) Students can easily be provided with the MATLAB source code and thus as many students as you like can use the VL simultaneously, asynchronously or indeed however they wish. The only impediment to accessibility is access to a suitable computer and the assumption that the student has downloaded the relevant files.
(3) MATLAB is easy to code and thus one can produce an effective VL using the GUI environment in about half a day with minimal expertise.

This paper will make 2 brief contributions: first it will demonstrate some of the VL the author has produced for chemical engineers to support learning of modelling and control and second it will give an introduction to the coding requirements in the hope that readers will be reassured that this is indeed a skill they could easily and quickly acquire.

2. EXAMPLES OF MATLAB BASED VIRTUAL LABORATORIES

This section will illustrate 4 examples of virtual laboratories that have been produced for chemical engineering students to help them relate their module in modelling and control to real scenarios and also to reinforce key concepts. Currently the author embeds the use of these into a quiz assessment to ensure students make use of them, but a long term plan when hardware becomes available (a new teaching building is nearly complete), is to make a closer link with a real laboratory using a tri-lab design (Abdulwahed 2010).

2.1 Dynamics and GUI for a mixing tank

A simple mixing tank can be modelled by an equation of the following form:

$$\frac{V}{F}\frac{dC_A}{dt} + C_A = C_{A0} \tag{1}$$

where V is the tank volume, F the flow rate through the tank, C_A the concentration coming out of the tank and C_{A0} the concentration of the inflow. A typical set of learning outcomes are for students to understand the impact on behaviours of changes in any of the parameters (V, F) and the input (input flow concentration).

The remainder of this section describes the GUI created by the author for this scenario. A short video demonstrating how to run and use this file is available on this link [http://controleducation.group.shef.ac.uk/matlabguis.html].

A screen dump of the VL (or GUI interface) is shown in figure 1. In this case the student has used 3 different values for input flow from which it is clear that the time constant, but not the gain, depends upon the input flow. It is also clear that asymptotically, the output concentration matches the input concentration.

- The GUI will overlay lines each time the push to update button is selected. Hence students should plan which variants of parameters they wish to overlay before beginning.
- Students can change 4 different values and thus explore how each of these affects the dynamics.

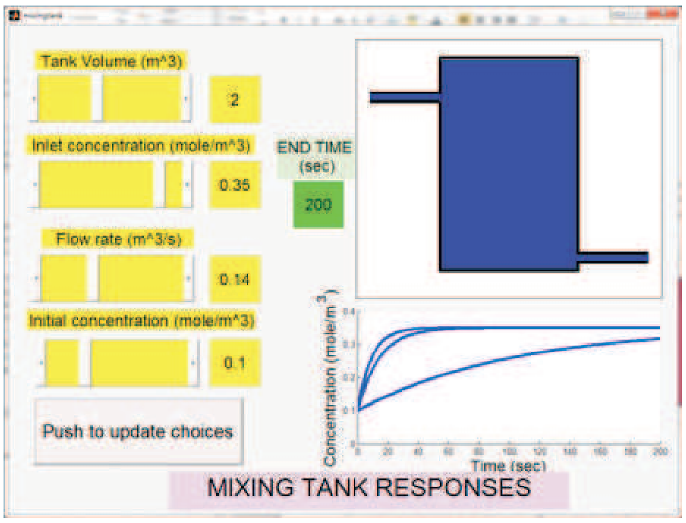

Fig. 1. Screen dump of mixing tank GUI with 3 different choices of input flow rate.

- Although not clear on this figure, the colour of the fluid in the tank changes smoothly to represent the concentration (red for low C_A and blue for large C_A); this animation effect is to help students visualise the changing concentration.
- The animation runs at 20x normal speed (about 10sec for a single run); slow enough to see the animation but not so slow to be tedious.
- To reset, simply close the window and re-open.
- It is possible to augment this GUI such that each line plot has a different colour or style and to include labels however, this is an extra programming task and thus involves academic time which needs to be judged against the benefits or needs for the anticipated users. In the authors case this was not merited.

2.2 *Dynamics and GUI for a simple heat exchanger*

The example taken here is of a heat exchanger where the heat is supplied by condensing steam and this is used to heat up a flow of water. A simplified model of this is given as:

$$\frac{V}{T}\frac{dT}{dt} + T = T_{in} + \frac{\lambda}{\rho F C_p} Q \tag{2}$$

where V is the volume of the tank, F the flow rate of water, C_p the specific heat of water, λ the latent heat of steam and ρ the density of water. In this case the system has 2 inputs, the inlet temperature T_{in} and the steam flow Q. The remainder of this section describes the GUI created by the author for this scenario. A short video demonstrating how to run and use this file is available on this link (http://controleducation.group.shef.ac.uk/matlabguis.html). Students can explore the impact on gain, time constant and steady-state of changes in 4 parameters. The GUI is shown in figure 2 and has the following animation attributes:

- The animation runs continuously rather than for a fixed time.
- When a parameter is changed, the impact is immediate as demonstrated in figure 2.

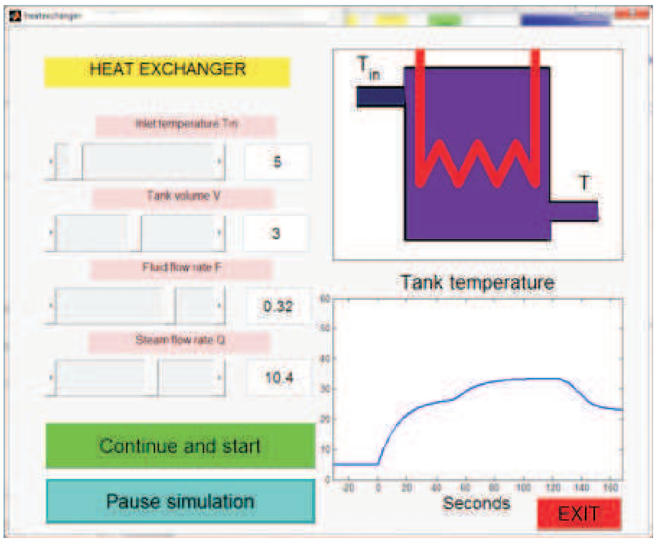

Fig. 2. Simulation of a heat exchanger with a change in steam flow Q at about 0 sec, 60 sec and a change in fluid flow rate F at 130 sec.

- The colours of the water in the tank change to represent temperature (light for hot and dark for cold).
- The line used to represent heat supply changes in thickness to represent Q.

2.3 *Tank level control*

Controlling levels in tanks is a common process for chemical engineers and thus a good example for a VL. In this case, a simplified model (assuming flow through a restriction is approximately linear) can be given as:

$$\frac{A}{\rho g R}\frac{dh}{dt} + h = \frac{1}{\rho g R} f_{in} \tag{3}$$

Where A is the cross-sectional area of the tank, g is acceleration due to gravity, R a constant linked to the outflow resistance, ρ is the density of the fluid and h is the depth. The remainder of this section describes the GUI created by the author for this scenario. A short video demonstrating how to run and use this file is available on this link (https://youtu.be/nVu2TsiwQhY). The GUI here assumes fixed tank parameters and focuses on analysis and design of a PI compensator to control the depth. The animations and interactions include:

- Students can choose open-loop or closed-loop control.
- Students can change the PI compensator parameters and the target depth to investigate issues such as offset and effective tuning.
- Students can enter their name before attached a screen dump to an assignment submission.
- The tank is seen to fill and also the inflow pipe changes shape to represent the magnitude of the input flow. Overflow of the tank due to poor compensator design is also represented. A typical run time is around 15 seconds so that students can view the behaviour effectively.
- Figure 3 gives a view of the GUI.

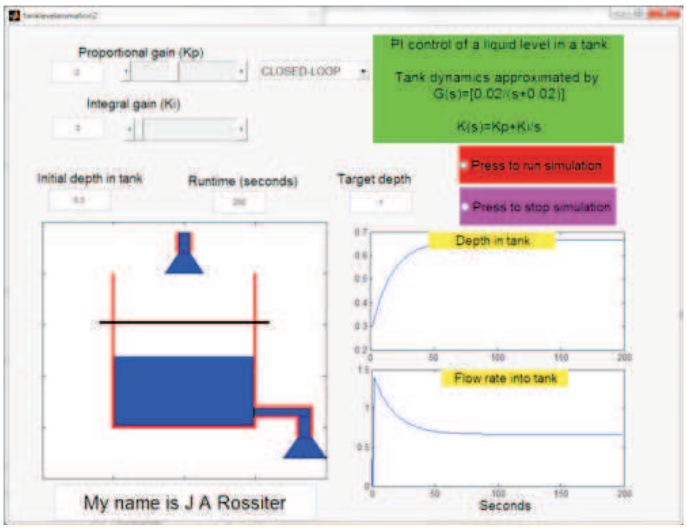

Fig. 3. MATLAB GUI for control of level in a tank with PI compensation.

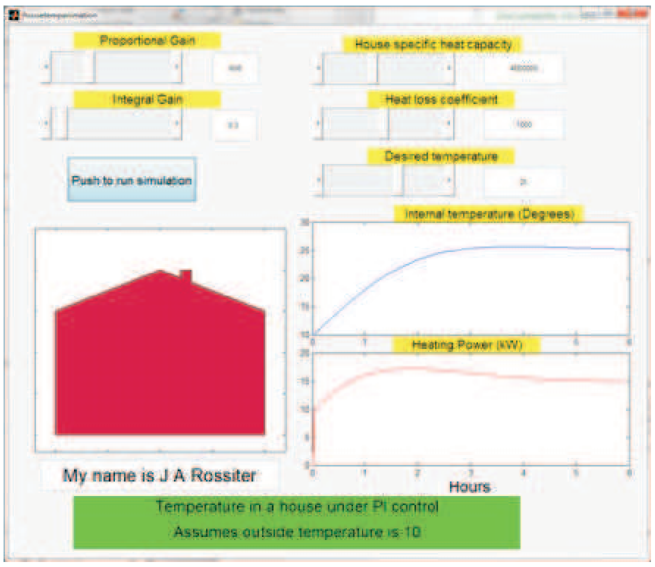

Fig. 4. MATLAB GUI to illustrate PI control of temperature in a house.

2.4 *Control of temperature with a simple house model*

This system is not a pure chemical engineering process but is chosen as it links well to students understanding of what constitutes reasonable behaviour and thus is a good platform to investigate the efficacy of PI compensation. A simplified model of a house with internal heating and heat loss due to temperature differentials can be given as:

$$C\frac{dT}{dt} + kT = kT_{out} + W \tag{4}$$

Where T is the internal temperature, T_{out} the external temperature, k is the heat loss coefficient, C the specific heat of the house and W the heating power. The VL developed here allows students to change parameters C, k and also the PI parameters as well as the target temperature. The main objective is to support the learning of PI compensation design rather than the dynamics of the house temperature, although students can investigate how changes in the house parameters necessitate a change in the PI parameters. As with other GUIs, the animation runs much faster than real time (about 5 sec for 6hr) and also the colour of the house changes to give a visual impact it also gives an alert if the temperature goes too high due to a failure of the control design.

2.5 *Summary of MATLAB GUIs and access*

Within the authors University students can access or copy the GUI files from a shared folder. External users can get them from an open access website (Rossiter 2015) under the MATLAB section. There are additional VLs on the site but the focus of those is more on control and modelling in general rather than specifically towards chemical engineering.

3. AUTHORING VIRTUAL LABORATORIES USING MATLAB TOOLS

This section will be brief as MATLAB itself comes with numerous guides and directions on how to use the tools. Rather the aim of this section is to illustrate to readers how quickly and easily an effective VL can be developed within the guide environment. The author has prepared a short 15min video showing the context and construction of a GUI from start to finish so that readers can see the processes for themselves (https://youtu.be/2hxfV2osjxY).

3.1 *The basic operation of guide*

GUIDE is a windows driven environment so that users build their GUI using drag/drop and resize of icons in an intuitive fashion (see figure 5). Moreover, each icon has a set of easy to access properties such as fontsize, colour, etc. so that the author can shape their GUI to the desired student view double click on an icon to access the properties. The icons can be moved around the screen to the desired position using the mouse and the screen can be resized in an intuitive manner.

Some obvious icons that an author may wish to use are: (i) sliders and edit text boxes for users to enter values; (ii) pop-up menus to allow selection from a list; (iii) push buttons to set an action going; (iv) axes for display of plots or other pictures. These are shown in figure 5.

3.2 *Basic coding of a MATLAB GUI*

A GUI consists of 2 files, a *.fig file which gives the screen layout and a *.m file which contains the implied computation and calculations required; a GUI will not run unless both files are in the same folder! After setting up the basic screen (*.fig) file, when the user saves then the associated *.m file is created automatically and indeed, anytime an extra icon is added to the fig file and saved, the *.m file is augmented as required.

- The m-file is critical as this determines the operation of the GUI. Each active icon such as a slider or push button will have a related subfunction within the m-file which will be denoted something like: function slider1_callback( )
- Whenever the user selects a particular icon, the code in the associated call-back subfunction is activated, so

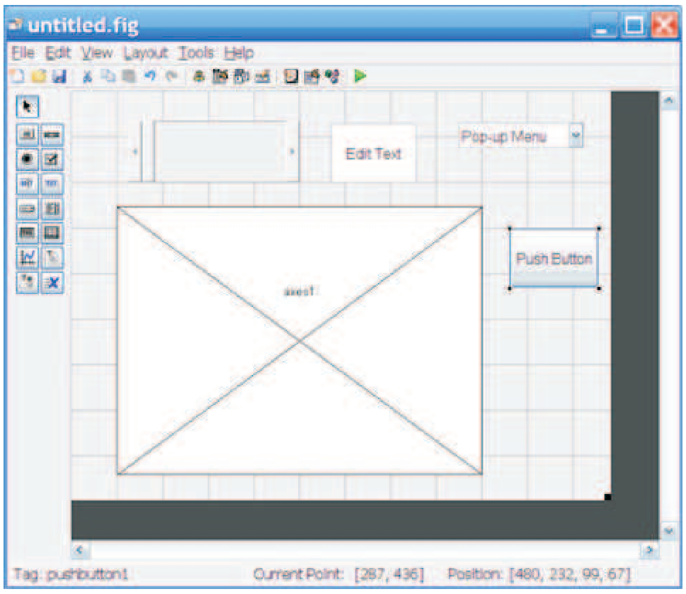
untitled.fig
File Edit View Layout Tools Help
Edit Text
Pop-up Menu
Push Button
axes1
Tag: pushbutton1
Current Point: [287, 436]
Position: [480, 232, 99, 67]

Fig. 5. Edit mode in GUIDE where the left hand toolbar contains the icons and the main screen indicates the shape of the final student view.

the coding for the academic is easy: what do I want to happen when a student selects that icon simply put the required code here!

- Readers should note that this subfunction can be empty if you want nothing to happen such as when the user is changing a parameter value but this change is not to be implemented yet. The author leaves many subfunctions empty and this accords with a generic aim of keeping life simple for academics.
- When MATLAB creates the subfunctions, it always gives a line showing how to access the properties of the icons and this includes things like the values from sliders, e.g. *v=get(handles.slider1,value)* will give the current value of the slider with name *slider1* . Hence it is easy to get all the required parameter values from within any subfunction and perform computations based on these.
- A demonstration can be done real-time at the conference for those who would like this and is also available on the link (https://youtu.be/2hxfV2osjxY).

3.3 Animating the GUI the easy way

Without animation, one could write a good GUI with sliders, edit textboxes and plots in about 30min (as seen in the example video provided) this is very fast and may suit the purposes of many academics. Nevertheless, a critical point for a VL is how to create animations, that is to give life to the GUI so that things are changing in real time. The author is a pragmatist and uses a very simple tool, that is the pause function within a loop. As MATLAB is effectively instantaneous for simple computations, hence a command such as pause(0.2) will stop the GUI for 0.2sec meaning a loop with 100 iterations would translate to a run time of about 20sec; readers can make their own decisions on what timescales would work for them.

The basic idea is to update any colours, line plots, tank fills and so forth every time one goes through a loop (this is done using a command such as *set(plot1,Xdata,tt)* which updates the x-coordinate data in the axes with tag *plot1*). If these updates are fast enough, but not too fast, then the user will be able to perceive and relate to the associated dynamics. The associated dynamic simulation could be discretised to run within this loop, or done in a single shot and then the values accessed as required.

It is relatively easy to write a GUI which runs for a prescribed runtime, animating as it goes, and then stops. The basic technique is to use a pushbutton to begin the animation and do not allow interruptions until this is complete. If the author wishes to allow the user to update parameters on the go, then the coding is slightly more subtle but still relatively straightforward. Nevertheless, it may be obvious that the more functionality you want from your GUI, the harder the coding will become which is one reason the GUIs or VLs illustrated in this paper are relatively simple in scope.

4. CONCLUSION

This paper has made some simple but hopefully useful contributions. It has highlighted some virtual laboratories that have been created in the guide environment of MATLAB to support learning and facilitate independent study by chemical engineers. These VLs are open access so other academics can take them and distribute them to their own students. However, more importantly, the illustrations serve to demonstrate that the creation of such learning resources is now within the reach of non-expert programmers so that most academics could create one of these in about half a day and thus provide their students with interactive learning resources such as VL whereas previously these have been too expensive or complicated to produce. Indeed, as seen in the provided video, one can do something basic in well under an hour.

APPENDIX: ILLUSTRATIVE MATLAB CODE

An indication of the coding complexity is apparent from the code snippet below linked to the mixing tank GUI. it is noted that the core part of the code is less than 20 lines long.

- The first few lines collect the user selected information from the sliders.
- next, the appropriate response is computed. Here, as that has a known algebraic form, it can be done in a single line.
- The *for loop* is used to animate the presentation of the plot, thus slowing down the presentation of the response and illustrating the dynamics through a dynamic change of colours as the concentration changes.

```
% Change in input data leads to a change in concentration
% C(t) = (C(0)-Cin)exp(-Ft/V) + Cin
endtime=str2num(get(handles.edit6,'string'));
tt=linspace(0,endtime,200);
c0=get(handles.slider4,'value'); % initial concentration
Cin=get(handles.slider2,'value'); % inlet concentration
vol=get(handles.slider1,'value'); % tank volume
flow=get(handles.slider3,'value'); % flow rate
conc=(c0-Cin)*exp(-flow*tt/vol) + Cin;
```

```
% Plotting using pause function to animate
axes(handles.axes2);xlim([0,endtime])
linec=plot(0,Cin,'linewidth',3);
for kk=1:length(tt);
% Update colour [1-conc/.4,0,conc/.4]
axes(handles.axes1);
set(handles.inhandle,'Facecolor',[1-Cin/0.4,0,Cin/0.4]);
set(handles.outhandle,'Facecolor',[1-conc(kk)/0.4,... ;
% Update line plot
axes(handles.axes2);
set(linec,'Xdata',tt(1:kk),'Ydata',conc(1:kk));
pause(0.03);
end
```

REFERENCES

Abdulwahed, M, 2010, Towards enhancing laboratory education by the development and evaluation of the trilab concept, PhD Thesis, University of Loughborough

Cameron, I., 2009, Pedagogy and immersive environments in the curriculum, Blended Learning conference, 290-294.

Chen, X., G. Song, and Y. Zhang, 2010, Virtual and remote laboratory development: A review. In Earth and Space, Engineering, Science, Construction, and Operations in Challenging Environments, 3843-3852. doi: 10.1061/41096(366)368.

de la Torre, L., R. Heradio, C. A. Jara, J. Sanchez, S. Dormido, F. Torres, and F. Candelas, 2013, Providing Collaborative Support to Virtual and Remote Laboratories. IEEE Transactions on Learning Technologies.

Dormido, S., H. Vargas, J. Sanchez, 2012, AutomatL@bs Consortium: A Spanish Network of Web-Based Labs for Control Engineering Education, Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Discipline, 11, 206-225, A. Azad, M. E. Auer, V. J. Harward (Ed), IGI Global.

Fabregas, E., G. Farias, S. Dormido-Canto, S. Dormido, and F. Esquembre, 2011, Developing a remote laboratory for engineering education. Computers & Education 57:1686-1697.

Goodwin, G. , 2010, Virtual laboratories for control systems design. http://www.virtual-laboratories.com/(last checked 1/9/10).

Goodwin G.C., A. M. Medioli, W. Sher, L. B. Vlacic, and J. S. Welsh, 2011, Emulation-based virtual laboratories: A low-cost alternative to physical experiments in control engineering education, IEEE Transactions on Education, 54:48-55.

Guzman, J., K. Astrom, S. Dormido, T. Hagglund and Y. Piguet, 2006, Interactive learning modules for pid Control, IFAC symposium on Advances in Control Education.

Memoli, P., 2011, Virtual experiments, http://www.edshare.soton.ac.uk/6589/1/preloader-diode.html. Project funded by HESTEM.

Perez, J., S. Dormido and L. Vlacic, 2011, Enhancing student learning: On-line interactive laboratory for modelling of real world control system applications, IFAC world congress, pp.7268-7273.

Y. Qiao, G. Liu, G. Zheng and C. Luo, 2012, Design and realization of networked control experiments in a web-based laboratory, Proc. UKACC.

Rossiter, J.A., Y. Baradaranshokouhi, I. Lilley and C. Bacon, 2011, Developing web accessible laboratories for introductory systems and control using student projects, IFAC world congress.

Rossiter, J.A., S. Dormido, L. Vlacic, B. Ll. Jones, R.M. Murray, 2014, Opportunities and good practice in control education: a survey, IFAC world congress.

Rossiter, J.A., 2015, Lectures and resources in modelling and control, http://controleducation.group.shef.ac.uk/indexwebbook.html

Vargas, H., J. Sanchez, C. A. Jara, F. A. Candelas, F. Torres, S. Dormido, 2011, A Network of Automatic Control Web-based Laboratories, IEEE Transactions on Learning Technologies, 4, 3, 197-208.