*Int. J. Engng Ed.* Vol. 21, No. 6, pp. 1134–1147, 2005
Printed in Great Britain.

0949-149X/91 $3.00+0.00


# The Virtual Control Laboratory Paradigm: Architectural Design Requirements and Realization Through a DC-Motor Example*

C. S. PEEK, O. D. CRISALLE
*Chemical Engineering Department, University of Florida, Gainesville FL 32611.*
*E-mail: crisalle@che.ufl.edu*

S. DÉPRAZ AND D. GILLET
*Automatic Control Laboratory, École Polytechnique Federale De Lausanne (Epfl), Ch 1015, Switzerland*

*The architectural requirements of an effective Virtual Control Laboratory (VCL) are described from a generalized perspective that takes into account the objective of supporting flexible, active, and discovery learning. It is argued that a Web-publishable VCL accessible via standard browsers maximizes flexibility. A realization of the proposed architectural paradigm is presented in the form of a VCL specialized for the control of a DC motor, organized in a modular fashion in terms of an animation panel that provides a visual sensorial perception of the evolution of the state of the plant, an interaction panel that allows the student user to change key parameters of the plant and controller, and a navigation panel that consists of five tabbed windows. The windows can be selected one at the time, and they present to the student a users' guide, information about the plant, a description of the controller, as well as analysis and time-domain simulation results. The analysis window includes all the classical results of linear control theory, including the characterization of dynamic features such as the time-domain step response, the frequency-domain Bode plots, and the location of poles and zeros. In addition, the DC-Motor VCL is able to yield analysis and simulation results for open-loop as well as closed-loop configurations. A number of pedagogical scenarios where the VCL can be used to the benefit of the students' learning experience are discussed.*

## INTRODUCTION

Engineering educators are increasingly interested in developing tools that allow students to acquire hands-on laboratory experience without necessarily providing physical access to a building that houses specific experimental equipment [1, 2]. These tools often take the form of software environments that are commonly referred to as virtual or remote laboratories [1–6]. The software either allows a user to interact with an experimental setup located in another geographical location (i.e., a remote laboratory) or uses numerical simulation tools to emulate the behavior of an experimental system (a virtual laboratory). Hence, the environment is a virtual control laboratory when the plant is simulated via a mathematical model, and is more properly classified as a remote control laboratory when the interface allows the user to actually manipulate and monitor a physical plant located elsewhere. The appeal of virtual and remote laboratory tools is largely due to the increasing demand for active learning and flexible education, and for the appeal of implementing techniques of learning via discovery.

Active learning seeks to provide students with opportunities to better integrate and reinforce knowledge presented in the classroom, as well as to acquire the practical know-how that is such an essential component in engineering education. Problem-based learning and hands-on laboratory activities are good examples of active-learning vehicles. In traditional educational settings, the realization of an effective active-learning environment calls for intense levels of interaction with experimental resources, and requires the coordination of the efforts and schedules of multiple parties, including numerous students and pedagogical support staff. As a consequence, the effort may be challenged by significant obstacles stemming from logistical, organizational, and cost concerns. On the other hand, such constraints can be overcome by making available flexible learning resources that meet the student's individual needs and schedules. In particular, virtual and remote laboratories offer substantial flexible education benefits since they can be made accessible to the students at any time and from any location, features that cannot be easily matched by traditional learning environments [7]. Furthermore, virtual and remote laboratory resources can be combined with other flexible-education

* Accepted 26 July 2005.

tools, such as lectures offered via digital streaming-video, for example, to maximize the degree of flexibility of the learning environment provided to the learners. In addition, another substantial advantage of virtual and remote control laboratory tools is that they can be used to supplement traditional teaching methodologies. For example, they can be used during a traditional lecture to show the students how to apply concepts presented in class to a simulated or remote experimental system. This approach imbues a classical lecture with an active and flexible learning component, thus strengthening the pedagogical value of the lecture.

Another benefit of virtual and remote laboratories in education is that they promote discovery learning. In this approach students are given access to the laboratory with minimal instructions, and are allowed to explore the systems for themselves [8]. A virtual laboratory or a well-designed remote laboratory can offer students a chance to explore safely and easily the behavior of a system that may be physically inaccessible.

Given the significant appeal of the engineering-education benefits realized by virtual and remote laboratories, it is not surprising that several systems are already available on the Worldwide Web (cf. [3, 9–12], among others). An early vision for a virtual control laboratory architecture can be found in [13]. The available systems address the flexible hands-on learning paradigm at different functionality levels, and with different degrees of student interactivity. Some virtual laboratories, especially those developed using commercial simulation tools, can be easily customized to suit the users' needs, but tend to show limited potential for interactivity. For example, it is quite common that a simulation run has to be stopped for the purpose of changing controller settings or process parameters. Software plug-ins or expensive specialized third-party software are also often required to run the simulations. Another class of Web-based laboratories attains a substantial degree of interactivity through the use of proprietary software written by the laboratory developers in a formal programming language such as Java. As a consequence, these tools require knowledge of the specific language adopted, and attempts to customize the environment to address specific needs of the learner may involve substantial modifications of source code. Therefore, these laboratories are typically only developed for a specific and restricted set of simulated or physical systems. Another concern when using proprietary code is that instances of the same laboratory developed for different physical systems often have substantially different interfaces, requiring the user to re-familiarized themselves with the tools made available for each task. In summary, the typical Web-based laboratories that are currently available seek to strike a reasonable trade-off between user interactivity and usage versatility.

The objective of this paper is to define a general architecture for a Virtual Control Laboratory (VCL) and to demonstrate a realization of the proposed vision in terms of a concrete example. The VCL is a software environment specialized for control engineering education, capable of avoiding the trades-off present in existing Web-based laboratories, and offering a single framework that can easily be used to develop Web-based, interactive virtual and remote laboratories for deployment on a wide variety of systems and scenarios. The tool should be able to provide educational benefits in both flexible and traditional education, beyond those attained by existing software controls laboratories. It should also facilitate the developer's task of generating new laboratory experiments through a modular design.

This paper is organized as follows. The next section discusses the architectural requirements that must be met by a virtual control lab, followed by a section showing a realization of the proposed architecture specialized for the control of a DC motor. The section, 'Pedagogical scenarios' discusses pedagogical scenarios for deployment of the VCL and concluding remarks are given in the last section.

## ARCHITECTURAL REQUIREMENTS FOR A VIRTUAL CONTROL LABORATORY

### *General principles*

The relative shortcomings of the virtual control laboratories that have been proposed to date can be overcome by conceiving a VCL architecture that meets three general principles, namely, interactive Web-access capability, modularity of design, and an intuitive and user-friendly interface.

We argue that a VCL tool should be capable of deployment in the Worldwide-Web environment using the standard Web browsers. The alternative of using specialized client software renders the VCL less flexible, since the user cannot access the resources using only standard software that is available in most computers. Hence, Web access is the key ingredient in imbuing the VCL with flexible learning features, making it possible to accommodate each student's personal schedule. Therefore, the software development environment should support easy publication of the VCL on the Web. Furthermore, the Web VCL tool should be highly interactive, allowing the student to change selected parameters and immediately observe the effect of the changes. Such interactivity capabilities are crucial for the success of the VCL because the tool seeks to emulate experimentation activities that would traditionally be conducted in a physical laboratory where there is significant pedagogical value in exposing the student to a learning-by-doing environment.

Another crucial requirement for the VCL envisioned here is that the source code should be highly modular [14]. An instructor should be able to use the VCL to quickly and easily develop a simulation

and analysis environment for another system of interest. This modularity is realized by developing a framework where the interface used to set up the analysis and simulation for one physical system can be easily modified to set up the analysis and simulation for a different system.

A third requirement for the VCL is that is should have a user-friendly interface. In fact, the student should find it easy to learn how to use the Web-base laboratory if the tool is expected to have substantial pedagogical value [15]. An excessive number of instructions can interfere with the discovery–learning process. It is therefore of paramount importance that the interface be intuitive. It may be desirable to make the interface resemble that of other software and hardware interfaces that are already familiar to the user, in order to simplify and accelerate the learning process. Another principle regarding the intuitiveness of the interface is that the same information should be deliberately presented in several different formats, such as via graphical animation, through displays on slider bars, and also shown in a chart, for example. Consistent use of color is recommended to help the user quickly identify different representations of the same data.

*Specific architectural requirements*

Specific architectural requirements that are of central concern in the design of a VCL include the following: (1) the ability to define and modify the plant and the controller, (2) easy transfer from open-loop to closed-loop operation by switching the controller from manual to automatic mode, (3) the inclusion of advanced analysis tools, (4) the ability to support control synthesis tasks, (5) the ability to produce open-loop and closed-loop simulation of time-domain responses, (6) the inclusion of versatile signal generators to produce standard input signals, (7) the inclusion of animation to project a visual perception of the evolution of the plant, (8) the ability to incorporate the dynamics of sensors and actuators, as needed, (9) embedded documentation and help-guides, and (10) reporting capabilities.

As a specialized tool for control engineering, the VCL must include two key components, namely a physical system of interest (which is hereafter denoted as the plant), and a controller. Examples of plants typically considered for pedagogical purposes in standard textbooks include a DC motor, an inverted pendulum, a helicopter, and a chemical reactor, among others. The plant model may contain significant nonlinear dynamics. On the other hand, unless the VCL is specifically designed to illustrate nonlinear control theory, from an educational perspective it is particularly important to include a linearized version of the plant model to enable the students to develop experience with the systematic and well-understood tools of linear control analysis and design. In addition, the VCL should allow the user to change the value of selected physical parameters of the plant, such as the inertia of a DC motor or the heat-transfer area in a chemical reactor, thus allowing for the investigation of the different dynamic-response regimes.

Examples of controllers include the ubiquitous proportional-integral-derivative (PID) control scheme, lead–lag control, on–off control, cascade control, and optimal control, among others [16] [17–23]. The user should be able to change key parameters of the controller, for example, to adjust the gain of a PID controller for the purpose of carrying out tuning experiments. When pedagogically suitable, the user should be able to select easily from a list of relevant control schemes. For example, the user may choose to select a single PID controller and proceed to evaluate its performance, and then select a dual-PID cascade scheme for evaluation and subsequent comparison with the single-PID choice. In the case of multiloop control schemes, the VCL should easily allow the user to modify the selection of appropriate input–output pairings. As in the case of the plant, the VCL should make available to the user concise and factual information about each control scheme available. Since multiple control schemes may be available, to avoid confusion it is important that specific care must be taken to present the documentation to the user in a contextual fashion. In other words, only PID-related information is presented to the user when a PID controller is selected. Analogously, only lead–lag-control documentation is presented when such a type of controller is selected.

Another required feature of a VCL is that it should enable the user to set the controller in manual mode, allowing the user to test the response of the plant to diagnostic input signals, such as a step forcing or a sinusoidal excitation. This situation is known as open-loop control because the output of the process is effectively disconnected from the controller. Alternatively, the user may select a specific control structure for deployment, in which case the plant is manipulated in automatic mode by the controller, defining a closed-loop control configuration. A key requirement is that the VCL should clearly and unambiguously show the user that the system is in either open-loop or closed-loop mode. This can be achieved by displaying a message in all the relevant locations where it is important to distinguish between the two modes. A supplemental solution could involve the use of a different color for the controller, or for the interface background, to reflect the alternative modes. From a pedagogical viewpoint, a still more creative solution would be to present the user with a specific graphical rendition of the relationship between the manual controller and the plant under open-loop operation, clearly showing that the output of the process is no longer available to the controller. The VCL would then switch to a different graphical representation during closed-loop operation, perhaps showing a signal-flow diagram that clearly

connects the outputs of the plant to the automatic controller and the outputs of the controller to the input ports of the plant.

A most important concept taught in control engineering is the process of analysis of both the open-loop and the closed-loop systems. First, under open-loop control the analysis studies consist of revealing all the features of the plant that are of dynamic relevance. For linear systems the analysis task often involves the determination of the plant poles and zeros, the characterization of the frequency-response of the plant through magnitude and phase diagrams in the form of Bode or Nichols plots, as well as a characterization of the time-domain response of the plant to step or sinusoidal input signals implemented via the manual-mode controller. The VCL should update the poles, zeros, Bode plots, and all other analytical characterizations immediately after the user introduces any changes to the values of the plant parameters. The open-loop analysis tools included in the VCL should allow the user to classify the plant as a stable or unstable entity, to assess its level of damping, determine the value of its dominant time constants and its bandwidth, etc. Second, under closed-loop control the analysis task also consists of producing a map of poles and zeros, generating Bode or Nichols plots, and tracing time-domain response curves; however, in this case the results describe the entire loop created between the plant and the automatic controller rather than to only the plant. For example, a classical closed-loop analysis task is the determination of whether the controller succeeds in defining a loop that is stable, an issue that can be assessed by inspecting the location of the closed-loop poles on the complex plane. To realize its pedagogical objectives, it is required that the VCL be able to update the map of the closed-loop poles immediately after the user has made changes to the parameters of the controller or after a new controller structure is selected. Obviously, all other analytical results should also be updated after such user-defined changes occur in the controller and in the plant parameters.

Another key concept taught to control engineering students is the process of control synthesis, which typically consists of specifying an appropriate control structure (say a PID or a lead–lag scheme) as well as values of the parameters included in the structure (such as controller gains and tuning time-constants). Such specification must meet specific closed-loop requirements, such as the location of closed-loop poles, the degree of overshoot, the settling time of the time-domain response, etc. A requirement of the VCL architecture is that it must allow the student user to carry out synthesis work with the benefit of the ability to quickly observe the effect of any changes made to the controller and to quantify the resulting closed-loop performance. Since the assessment of the controller performance in closed-loop mode is in fact an analysis task, the VCL should be able to deliver a desirable control synthesis environment whenever its analysis component meets the architectural specifications discussed in the preceding paragraph.

It is of critical importance that the VCL enables the student to carry out open-loop as well as closed-loop simulations of time-domain responses. Numerical simulation studies under open-loop mode permit the characterization of the plant dynamics through the observation of plots that document the response produced by standard input signals, such as step and sinusoidal excitations. In addition, the closed-loop simulation capabilities allow the student to observe the time-domain effect of analytical features (such as the effect of the closed-loop poles on the plant response), as well as to assess the time-domain performance of alternative control design choices. The VCL should therefore include software resources that solve differential equations using robust numerical techniques, and have the ability to cope with situations where the equations are stiff.

The VCL must include a number of signal generators that can inject appropriate types of input at the user's discretion. These elements should be able to produce standard signal patterns, including constant, step, sinusoidal, ramp, and pulse signals of different amplitudes and frequencies. Making available a large palette of options increases the versatility of the VCL, given that certain problems require the use of less common signals, such as saw-tooth patterns, and other specialized input waveforms. In particular, the signal generators are used: (1) to define set-point signals that are fed to the controller under closed-loop control operation, (2) to produce a user-specified manual signal emanating from the controller under open-loop operation, (3) to inject a process disturbance signal as needed to test a controller's ability to perform as a regulator, and finally (4) to simulate the injection of noise signals at selected points in the loop, in which case it is necessary to produce appropriate white and colored noise signals of a random nature.

A most desirable feature in a VCL is the ability to present the user with the evolution of the plant variables via two-dimensional or three-dimensional graphic animation. For example, the motion of a robotic arm may be shown in a graphic form that gives the user the impression that he/she is observing the movement as it would occur in three dimensions, much as if the motion were presented to the observer through a video-camera image. As another example, the motion of the indication needle in a pressure gauge can be shown to reflect changes in gas pressure inside a chemical reactor.

Although the plant and the controller are the essential elements of the control loop, the VCL should have enough flexibility to include other components that are sometimes the focus of significant pedagogical interest, such the case of

sensors, power converters, and actuators. The VCL architecture should also include a window that reports error messages to the user, and provides advice on how take corrective action. For example, the user can be alerted with a message that a negative integral time-constant is not allowed in a PID controller, and advised to specify an acceptable non-negative value. Finally, even though the analysis tools of the VCL provide the reader with plots and figures that characterize the dynamics of the open- or closed-loop system, it is of significant value to the user to allow for the capability to export the data to a file that can later be opened by another specialized utility, such as a plotting program or a text editor. The export feature should be conceived with a view to producing an archival file that contains an integral report of the experiment, thus documenting the values of the plant and controller parameters, the structure of the controller, and the analysis results.

The VCL should include documentation that presents the user with a fundamental description of the plant and of the controller, including the dynamic model used for analysis, along with a clear description of the most important physical parameters. Even when the interface is highly intuitive, explicit instructions are necessary to ensure ease of use. This is especially true as the system being simulated or the controller type used may not be familiar to beginner learners who stand to gain the most educational value from the VCL tool. In order to accomplish this, help documents should be embedded directly in the virtual laboratory interface, making them conspicuous and easy to access. The documentation should be complete, and sufficiently concise and factual, and should be presented from a perspective that is useful for control engineering design and analysis. Inclusion in the VCL of all relevant plant and controller documentation as well as a users' guide contributes to making the environment self-contained, and hence more effective from the pedagogical point of view.

For archival purposes, the VCL should include reporting capabilities, including a print resource that is easy to use, such as a button that responds to a single computer-mouse click, hence enabling the user to produce records of the results in the form of a paper copy or as a graphical computer file. In addition, it is highly desirable to include the capability for exporting the results of a simulation run to a file that in turn can be used as a source for

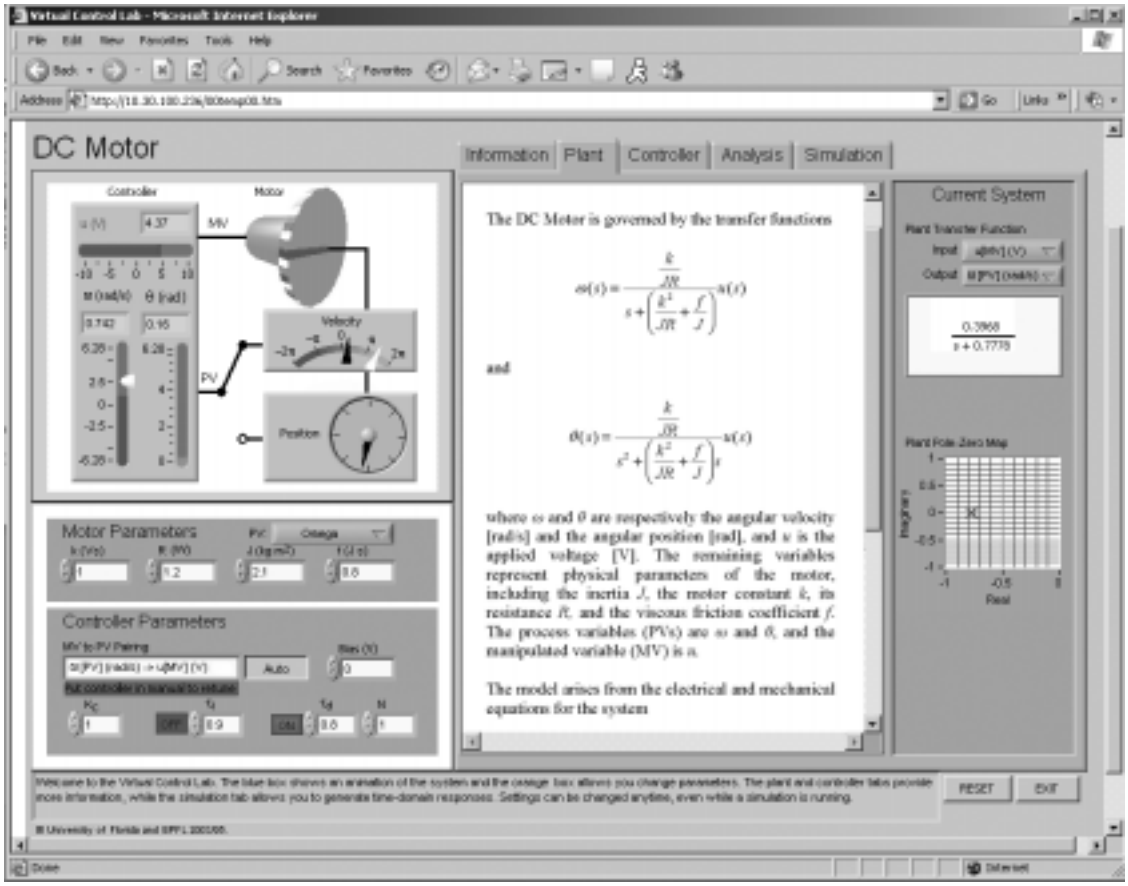


Fig. 1. Realization of a DC Motor Virtual Control Lab accessed by students using a standard Web browser. The interface presents three distinct panels: (1) an Animation Panel (located on the upper left area) showing details of the DC motor under closed-loop control, (2) an Interaction Panel (lower left) that allows the user to define the parameters of the plant and of the controller, and (3) a Navigation Panel (right-half area) consisting of a series of five tabbed windows that contain different types of information as indicated by the tab labels

other specialized software tools, or for creating report tables using word-processing software.

## REALIZATION OF A VIRTUAL CONTROL LABORATORY

Taking into consideration the architectural principles and requirements given in the previous section, the authors have developed a VCL framework that can be accessed by students using a Web browser. Figure 1 shows a view of the proposed VCL inside a standard Web-browser window, as it is presented to a student. In this case the plant studied is a DC motor that is shown as an animated graphics on the upper-left of the VCL interface window. On the lower-left rectangle of the window the user can define different physical parameters for the DC motor, and can also select a controller structure as well as specify all values of the controller parameters. Finally, the right-half of the screen shows five tabbed windows that contain various types of information, including plant and controller documentation, as well as the results of analysis and simulation studies.

The DC-Motor VCL attempts to address many of the key architectural criteria outlined in the previous section, while seeking to strike a favorable a compromise between the perceived advantages and disadvantages of the Web-based laboratories discussed in the first section. The following sub-sections describe specific details of the proposed DC-Motor VCL, and discuss the extent to which different features of the VCL succeed in meeting the design standards.

*Software platform for development*

The example shown in Fig. 1 is a realization of the Virtual Control Lab concept created using National Instruments' LabVIEW software [24]. This software offers many powerful resources for developing and maintaining a VCL, including ease of creating standard interface controls, and a built-in Web publishing capability. In addition, the software provides a suite of recently released resources that are useful for control-engineering education, such as the NI LabVIEW Simulation Module and the NI LabVIEW Control Design Toolkit, which automate many of the analysis and simulation tasks carried out by the VCL. These advantages of LabVIEW, combined with the option of programming using a graphical language, make developing, maintaining, and modifying the VCL a relatively simple and straightforward task. Development and implementation alternatives for concepts analogous to the VLC prototype proposed here are also possible using the Java programming language, as described in [25]; however, in that approach all advanced analysis tasks must rely on calls to external software packages.

*Interface design based on three specialized panels*

To provide the user with an organized and intuitive access to the learning resource, an interface consisting of the following three distinct areas is proposed: (1) an Animation Panel, (2) an Interaction Panel, and (3) a Navigation Panel. Figure 1 shows an example of the interface, featuring the three-panel structure. The Animation Panel (located on the upper left region of the interface window) and the Interaction Panel (located on the lower left region) are permanently displayed, allowing the observation of key features of the system and the controller on a continuous basis, emulating a perspective that the user would have when running the equivalent experiment in a physical laboratory. The Navigation Panel,

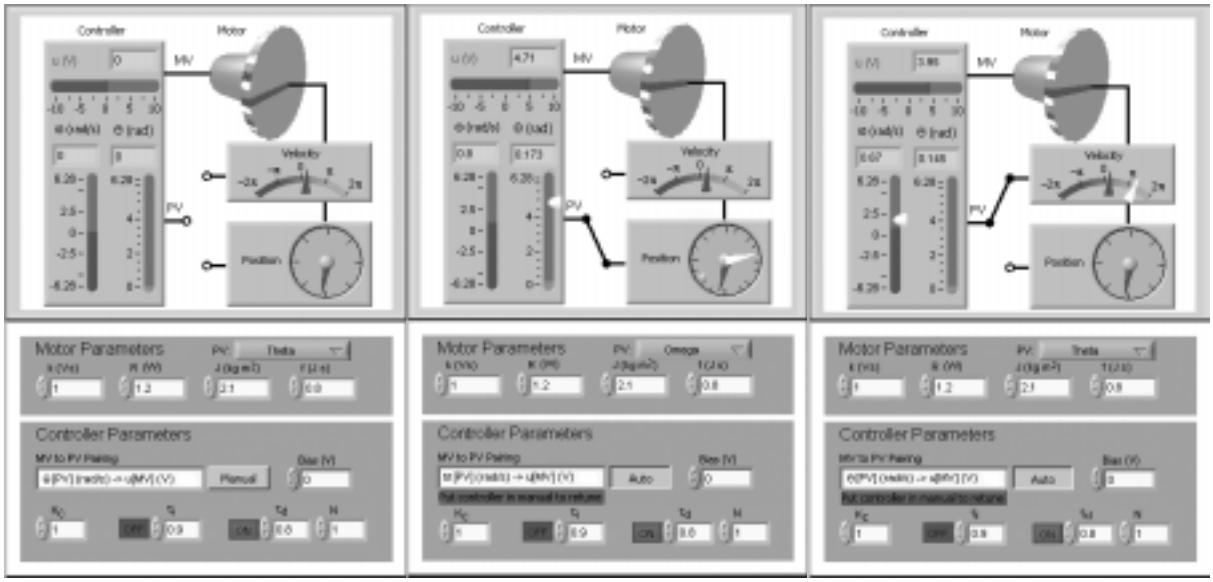


Fig. 2. Animation Panel and Interaction Panel showing (a) the controller set in Manual mode, (b) the controller set in Auto mode using the angular position as the process variable used as feedback, (c) the controller set in Auto mode using the angular velocity as the process variable used as feedback

defined by the larger rectangular area located on the right of the interface, is characterized by the presence of five tabs that activate windows that display different types of information.

Special care has been taken when creating the content of the three panels to ensure an intuitive interface. Whenever possible, standard dialog buttons, input boxes, and tab panels are used, and they are chosen to visually resemble those found in standard home-computer software. Thus, users have to learn only the specifics of the laboratory experiment itself, avoiding the confusion that may potentially be introduced when adopting esoteric interface controls. The intuitive nature of the VCL interface is also enhanced by assigning to each input and output signal a distinct color and, whenever possible, all interface elements related to a given variable are characterized with that variable's specific color.

*Animation and interaction panels*

The Animation and Interaction Panels are closely integrated with each other, and it is therefore convenient to discuss them jointly. The purpose of the Animation Panel is to give the user a visual representation of the plant, along with indicators that display two-dimensional or three-dimensional motion. The purpose of the Interaction Panel is to allow the user to define plant parameters, and to adjust control parameters as needed. Choices made by the user in the Interaction Panel (such as a change in the controller state from Manual to Auto) trigger a change in what is displayed in the Animation Panel. Hence, the two panels are highly coupled. Figure 2 shows an implementation of the Animation and Interaction Panels of the DC-Motor VCL.

The Interaction Panel in Fig. 2(a) shows a situation where the controller is set in Manual mode (see the area labeled 'Controller Parameters'). Hence, the system is in an open-loop configuration. In this case the Interaction Panel also indicates that the motor inertia $J$ has the value of 1 kg-m$^2$ (see the area labeled 'Motor Parameters'). The Animation Panel of Fig. 2(a) shows a controller linked with an electrical wire to the input of a three-dimensional rendition of the DC motor. In turn, the motor has an electrical connection to two sensors, namely velocity and position indicators. The DC motor is represented using a geometrical perspective consisting of a stationary cylindrical body (the stator) connected to a rotating disk (the rotor). The disk rotates at different speeds, and the motion is made visible to the observer with the aid of bars painted on the edge of the disk and a rotating ray drawn on the disk face. An electrical wire connects the motor to the angular velocity indicator represented by a speedometer, and also to a round dial that displays the angular position. These two output indicators reflect the fact that in this DC motor control problem either the angular position $\theta$ or the angular velocity $\omega$ may serve as the process variable (PV), namely as the measured variable that is eventually to be maintained at a set point by the controller. Since the controller is set to Manual mode in the Interaction Panel, in this case the system is configured in open-loop mode and consequently the Animation Panel shows that the input wire representing the process variable (PV) is disconnected from the indicators. When in Manual mode, the student user can change at will the output of the controller indicated in the panel as the manipulated variable (MV), using either the bias box in the Interaction Panel or using a signal generator located in the Navigation Panel that is discussed later in this section.

Figure 2(b) depicts a situation where the user has used the Interaction Panel to switch the controller to Auto mode. In this case the Interaction Panel shows that the student has specified the angular position $\theta$ as the chosen process variable (PV), as indicated in the box labeled 'MV to PV pairing'. The system is now in a closed-loop configuration where the angular position is the process variable used for feedback control. The structure of the loop is made obvious to the user in the Animation Panel, where the controller input wire labeled PV is connected to the angular position indicator. The wiring scheme defines a closed loop, hence serving as a visual aid to enhance the student's awareness of the fact that the controller is in the Auto mode of operation. Note the appearance of a white needle on the angular position indicator, representing the controller set point. Likewise, a white triangular marker is visible on the side of the vertical slide bar that represents the angular position, also indicating the value of the set point of the controller. Either the white needle or the white triangular marker can be moved to a different position using a computer mouse, thus introducing a set-point change. Set point changes can also be specified using a signal generator located in the Simulation window of the Navigation panel, as discussed later. The Interaction Panel shows the values of the proportional-integral-derivative (PID) controller, including a gain $K_c = 1$, an integral time constant $\tau_i = 0.8$, and a derivative time constant $\tau_d = 1$. On a location immediately to the left of the boxes that specify the values of the integral and derivative time constants, the user finds square buttons drawn with a relief perspective, which can be depressed via a computer mouse click to activate the integral and derivative terms. The figure shows that the integral-action term is turned OFF (which has the effect of ignoring the numerically-specified value in the $\tau_i$ box), and it also shows that the derivative-action term is turned ON. In keeping with the objective of maximizing interactivity, the parameters shown in the Interaction Panel may be changed at any time, even while a simulation is running.

Figure 2(c) shows an alternative automatic-control configuration specified by the user via the Interaction Panel. In this case the selected process

variable is the angular velocity $\omega$, as clearly shown in the Animation Panel where the speedometer is wired to the controller input port. Note that in this case the white needle now appears on the speedometer, identifying the value of the set point that the user has defined for the angular velocity. Likewise, a white triangular set-point marker appears on the vertical slider bar displays the value of the angular velocity. Hence, the Animation Panel is updated in an immediate fashion to reflect the configuration defined in the Interaction panel.

*Navigation panel and its tabbed windows*

The Navigation Panel allows the user to access a variety of pedagogical resources through a series of five tabbed windows bearing the following titles: (1) Information, (2) Plant, (3) Controller, (4) Analysis, and (5) Simulation.

First, when the user selects the 'Information' tab, a window opens up in the Navigation Panel displaying general information on how to utilize the VCL environment. This window serves as a users' guide that is embedded in the VCL, which gives the student access to information without leaving the interface. The text displayed is an embedded portable-document file (PDF) that can be scrolled down to reveal the entire contents of the document. The Information window succinctly describes the key features of the Animation, Interaction, and Navigation panels. In addition, the window defines for the student a number of control-engineering experiments that can be carried out with the VCL, clearly stating the objective of each experiment and suggesting a sequence of steps that could be followed to accomplish the stated goals. For example, one experiment has the objective of characterizing the open-loop step response of the DC motor. The text suggests to the users that they proceed by putting the controller in Manual mode, exciting the plant with a step change in the manipulated variable, and then identifying key features of the response of the process variable observed in the Simulation window of the Navigation Panel. Note that the choice of embedding a standard PDF document format in the window gives the user familiar access to the help documents directly from the VCL, and allows a developer to create the needed documents using standard tools that are in widespread use.

Second, selection of the 'Plant' tab brings up a window that displays an embedded PDF file containing information on the dynamics of the DC motor. The Navigation Panel of Fig. 1 shows that the Plant window presents the user with two transfer functions that describe the relationship between the applied driving potential $u$ (the manipulated variable) and each of the two variables that can be of interest, namely the angular position $\theta$ and angular velocity $\omega$ (the process variables). Using the scroll bar located on the side of the window, the user can display the remainder of the text, which describes how the transfer-functions are derived from the mechanical and electrical differential equations that define the dynamics of the motor. It also shows how the modeling equations can be written in a standard state–space form.

Third, the 'Controller' tab activates a window that displays an embedded PDF file with information about the different controllers that are available to the user. In the case of the DC-Motor VCL, the Controller window reviews the key principles

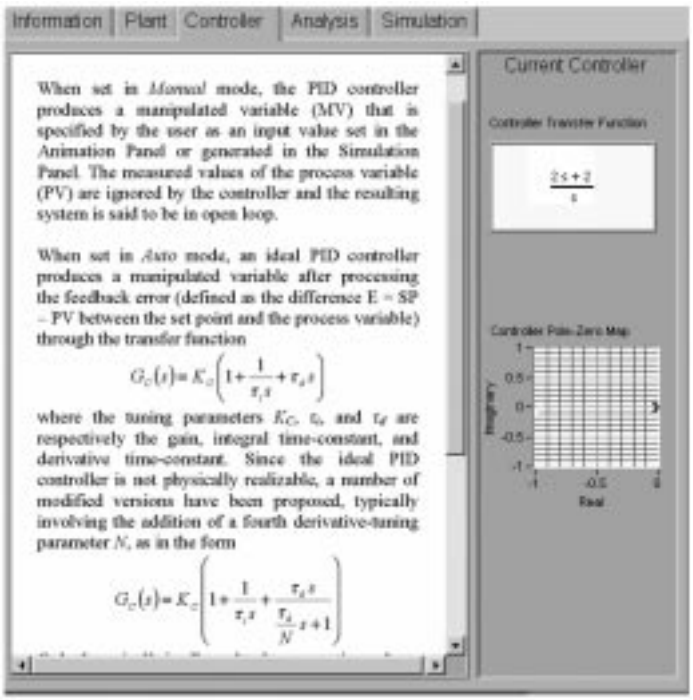


Fig. 3. Navigation Panel showing the Controller window selection. An embedded PDF file describes the features of the controller, while the frame on the right-column shows the controller transfer function as well as the location of its poles and zeros

of PID control theory, as shown in Fig. 3. The document describes the differences between the Manual and Auto modes of the controller, introduces the classical transfer function of an ideal PID controller, describes a practical realizable version of the PID law, and discusses the effect of the different tuning parameters of the controller. The window also shows the numerical coefficients of the numerator and denominator polynomials that define the transfer function of the controller (see the frame entitled 'Current Controller'). These coefficients are immediately updated when the controller parameters are modified by the user in the Interaction Panel. In addition, the Controller window shows a map of the location of the poles and zeros of the controller transfer function.

Fourth, the 'Analysis' tab displays a window that contains the classical analysis tools supported by control-engineering theory. As shown in Fig. 4, this window contains (1) a transfer function showing numerical values for all the coefficients of the numerator and denominator polynomials, (2) a plot of the time-domain response to a unit-step input excitation, (3) a map showing the location of poles and zeros on the complex plane, and (4) a Bode plot that provides frequency response information. When the controller is in Manual mode, the information displayed in the Analysis window refers only to the plant. In that case the system is in open-loop mode, and the DC motor is devoid of a feedback connection to the controller. Hence, the transfer function, step response curve, pole-zero map, and the Bode plot refer to the dynamics of the DC motor. On the other hand, when the controller is set to Auto mode, the information displayed in the Analysis window refers to the closed-loop transfer function relating the set point of the controller to the process variable (either $\theta$ or $\omega$, as defined by the user in the Interaction Panel). Hence, in this case the transfer function, step-response curve, pole-zero map, and the Bode plot describe the dynamics of the closed-loop established between the DC motor and the PID controller. Hence, the information shown in the Analysis window is synchronized with the user's choice of Manual or Auto control modes in the Interaction Panel. In that sense, the Analysis window is a contextual window, showing different analysis results depending on whether the system is in open-loop or closed-loop mode. As shown in Fig. 4, the Analysis window clearly displays a message indicating whether the system is in open-loop or closed-loop mode, hence ensuring that the user correctly interprets the contextual analytical results presented. The maximum and minimum values for the horizontal and vertical axes of any graph in the window can be easily adjusted by simply clicking on the corresponding tick mark and typing in a new value. This feature gives the

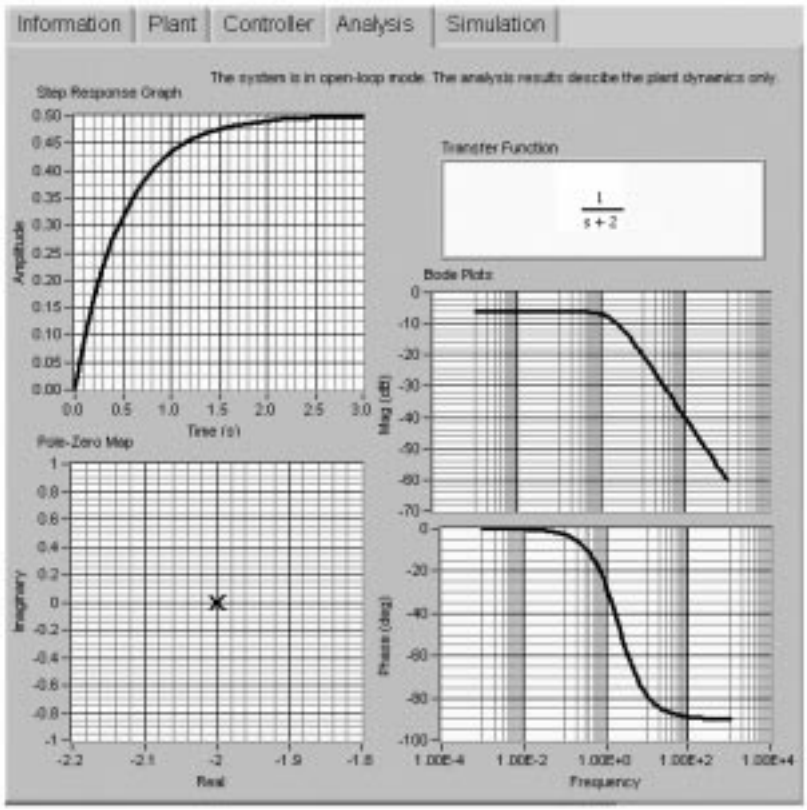


Fig. 4. Navigation Panel showing the Analysis window selection. The results include the unit-step response, Bode plot, pole-zero map, and the transfer function of the plant given that the system is in open-loop mode, as indicated by the message across the top of the window. Equivalent results are presented by the Analysis window when the system is switched to closed-loop mode

user great flexibility in adjusting the scale of the graphs, as needed to reveal features of interest.

Finally, the 'Simulation' tab of the DC-Motor VCL brings to the front of the Navigation Panel a window with a graph showing the results of a numerical simulation that describes the response of the angular position $\theta$ and angular velocity $\omega$ to a user-selected input. This is also a contextual window. More specifically, when the controller is in Manual mode, the graph describes the open-loop responses of the plant to a user-specified driving-voltage input $u$, in the absence of feedback control. In this case the user can change the driving voltage signal by selecting an excitation from the Signal Generator appearing in the Simulation window (see bottom half of Fig. 5). The signal options available in the DC-Motor VCL include a constant or sinusoidal signal, as well as a saw-tooth and a pulsed waveform. An alternative scenario is when the controller is switched to Auto mode and the system is therefore configured in closed-loop mode. In this case the user can change the set point of the controller by selecting an excitation waveform from the signal generator. In contrast to the previous case, the values of the driving voltage are now dictated by the PID control law, reflecting the fact that the controller feedback path is active during closed-loop operation. The numerical response of the plant or of the closed loop to the selected excitation is computed using a fourth-order Runge–Kutta integration algorithm. All the signals of relevance are displayed in the graph using lines of different line types and colors. The maximum value of the time axis can be easily changed by selecting the corresponding tick mark and typing a new value. In addition, the slider bar located below the graph allows the user to display previous traces that are no longer visible in current plot scope (*cf*. Fig. 5). The window also has a button labeled 'Run' that starts the numerical simulation task when it is depressed. The button label then turns to 'Stop', and serves to terminate the calculations whenever it is pressed. A button entitled 'Clear Chart' allows the user to erase all the traces. Finally, a button entitled 'Export' allows the user to create an ASCII file that contains all the results of the simulation calculations, as well as a summary of the relevant parameters that describe the plant and the controller. As in the case of the Analysis window, a message is clearly displayed, indicating whether the system is in open-loop or closed-loop mode, hence ensuring that the user can correctly interpret the simulation results presented. The simulation window is conceived to allow the user to make changes in the Interaction Panel while a numerical simulation is taking place. All changes are immediately accepted by the simulation resources, without requiring the user to stop a run in progress.

The authors have found that the use of a tabbed-window format for the Navigation Panel provides

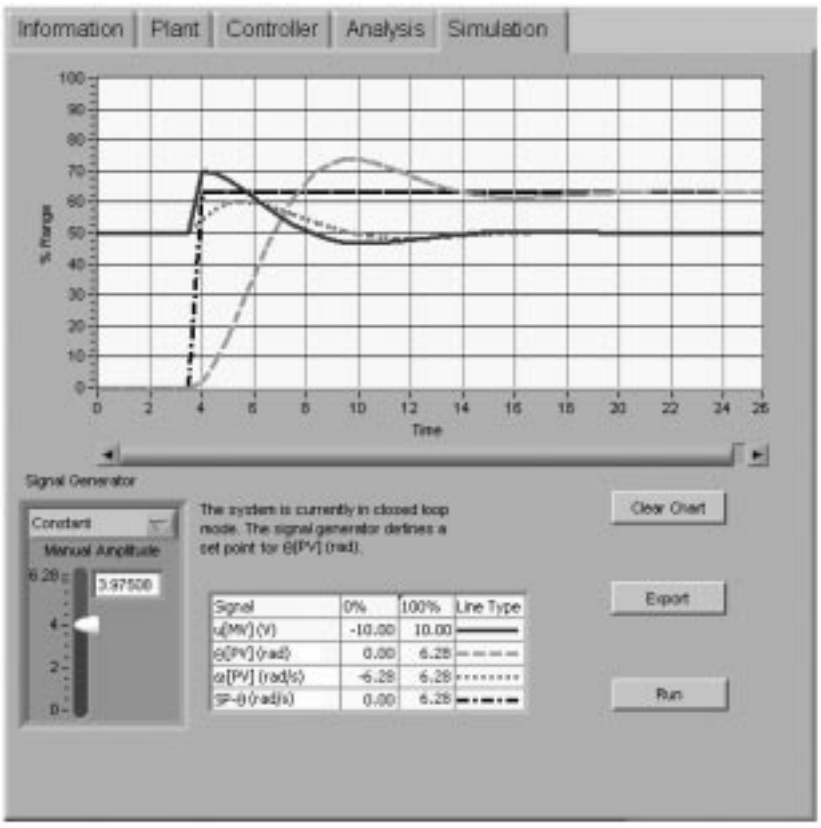


Fig. 5. Navigation Panel showing the Analysis tab selection. The graph displays the traces of all the relevant variables, and the signal generator allows the introduction of set-point changes

an intuitive method for organizing different aspects of the VCL. On the other hand, the use of an excessive number of windows appears to be counterproductive, given that the user can be confused or overwhelmed by a large number of options. In our opinion, restricting the number of tabs to only five appears to be a compromise that successfully resolves the competing need for a simple, elegant interface, and the need for displaying large amounts of relevant analysis and simulation results as well as documentary information.

*Publication and deployment on the Worldwide Web*

LabVIEW has built-in Web publishing capabilities that in principle make the software developments Web-accessible via a simple command. This creates one hypertext instance that can be accessed by one student at a time using a Web browser. On the other hand, in a teaching environment it is necessary to permit multiple users to have full access to the VCL at the same time. This can easily be accomplished by creating a common gateway interface (CGI) front end that, upon request, produces replicas of the VCL and runs a copy for each connected user. To minimize the use of storage resources, the CGI must also remove any VCL copies that are no longer in use. An effective CGI can be created using LabVIEW, which is the alternative selected by the authors, or could be written using traditional languages such as Perl [26]. An extensive discussion of technical alternatives is presented in [27].

In order to make available on a Web browser window all the interactive features of DC-Motor VCL created by the authors, it is necessary to install the LabVIEW Run-Time Engine in the remote computer. This engine is available as a free download that in turn installs an appropriate browser plug-in that is of critical importance to ensure an interactive interface. Fortunately, when a student attempts to use the VCL for the first time a dialog message appears if the Run-Time Engine plug-in is not installed, and the user is prompted to download and install the required executable file. On the other hand, the need for a plug-in solution, common to many interactive virtual or remote laboratories, reduces the flexibility of the learning environment because the students are constrained to using computers on which they have the appropriate permission to install a browser plug-in.

The Web-based DC-Motor VCL is designed to support a high level of interactivity with the student user. The user can change controller and plant parameters, select alternative process variables for control purposes, establish an open-loop or closed-loop mode of operation, change the ranges of all the horizontal and vertical axis on the analysis plots, launch simulation studies, and change set points. In this fashion, the proposed tool satisfies a key structural requirement, namely, the attainment of a highly interactive interface via a Web-browser window.

*Modularity other software design details*

LabVIEW routines and their interfaces are stored in files referred to as virtual instruments (VIs). The actual functions that define the plant and controller are stored in sub-virtual-instruments (subVIs) in LabVIEW. The DC-Motor VCL is conceived in a highly modular fashion designed to enable an instructor to reuse the software to design another VCL. The instructor would proceed to first change the icons in the Animation Panel to reflect the physics of the new plant. Second, a state–space realization of the transfer function describing the relationship between the input variable and the manipulated variable must be coded into the hidden LabVIEW wiring diagram associated with the Simulation Panel VI. Third, the input boxes used to define plant parameters in the Interaction Panel VI are wired to their corresponding entries in the transfer function, so that any parameter changes are immediately reflected in the simulation. Fourth, updated PDF files are embedded in the Information and Plant windows of the Navigation Panel VI, respectively describing the new VCL experiment and its plant. A subVI is used to define information common to multiple VIs, such as variable names, for example, so that the framework can automatically update all the interface elements in the Navigation Panel to reflect the current system without any effort on the part of the developer. The developer would change the minimum and maximum values of the percentage scale used in the graph of the Animation Panel. Other smaller tasks may be involved in the process, such as adding or eliminating new plant-parameter boxes in the Interaction Panel, and rearranging the buttons, boxes, and other icons to produce an esthetically pleasing interface. Finally, the resulting VCL can be published for Web access. In this fashion, the resulting modified VCL and its original counterpart will have a common interface, each one customized in terms of animation and labels that reflect the physics of their respective plants.

The development takes advantage of the NI LabVIEW Simulation Module to calculate closed-loop time-domain responses. It is straightforward to configure the simulation to allow the user to implement changes to the plant or to the controller as a simulation is in progress, hence avoiding the need to restart a run whenever control or process changes are made. This behavior is obtained by conducting the simulation over a series of successive finite periods of durations larger than the integration step size used by the simulator. Data decimation used to report the results at pre-specified intervals, and an update of the plant and controller parameters is done at the end of each finite period so that user-defined changes can be quickly recognized by the

simulator. We have found that a Runge–Kutta algorithm of order four with a fixed step size [28] is an adequate solver for the differential equations that describe the dynamics of the DC motor.

## PEDAGOGICAL SCENARIOS

The VCL can be used in a number of pedagogical scenarios conceived to take advantage of the benefits of active learning, flexible learning, and learning via discovery.

A first scenario of interest is the use of the VCL during a traditional lecture to illustrate a particular control-engineering concept. For example, the instructor could use the VCL closed-loop capabilities to show the students how the offset characteristic of proportional-only controllers can be reduced by choosing larger absolute values for the controller gain. The instructor could then show how at the point of ultimate-gain the benefits of reduced offset begin to erode due to the onset of an undesirable persistently oscillatory response. The students would then observe animated evidence of how even larger values of the controller gain lead to instability. Other examples of concepts that can be illustrated are the effects of time constants on the plant time-domain response and on the frequency-domain bandwidth, the relationship between pole locations and stability, and the effect of open-loop zeros on the time-domain response of the plant, among many others.

The instructor can choose to present the VCL demonstration either before or after the control engineering concept in question is treated in the lecture. Presenting the VCL demonstration before the lecture discussion gives the students a strong motivational incentive to develop a curiosity for the problem, and helps them to identify the key technical problems that need to be addressed by the theory. The ensuing lecture discussion of the topic, presented on the blackboard or via other traditional delivery methods, such as using viewgraph support, is likely to be more successful in keeping the students focused on the problem because they have already acquired relevant experience on the problem through the VCL. Alternatively, the instructor may choose to conduct a VCL demonstration after the topic has been presented on the blackboard. In this case the VCL provides an opportunity for reinforcing the knowledge gained, providing visual experience in the form of two-dimensional or three-dimensional animation and simulated signal graphs that help the students make cognitive links between associated concepts that often appear more abstract in nature when presented on the blackboard.

A second scenario of interest is the use of the VCL in the context of informal cooperative learning exercises in the classroom [29]. Using well-established techniques the instructor directs the students to organize themselves in small working groups, and defines the goals of a pre-planned exercise that needs to be solved cooperatively by all the students in the group [29] with the assistance of the Web-accessible VCL. Examples of exercise activities include tuning controllers, identifying the order of a plant from an open-loop step response, finding the ultimate gain of a proportional-only controller, etc. These activities fall under the categories of discovery and active learning.

A third scenario is the use of the VCL to support formal cooperative exercises [29], also known as group projects. In this case the students are given a long-term assignment that involves several intermediate steps, culminating with a comprehensive final report. In this case the VCL would be used to provide support for the completion of the intermediate steps, such as modeling the plant, characterizing its frequency-domain properties, comparing and contrasting the frequency-domain and time-domain performance of alternative control schemes, and validating the stability of proposed control design in terms of closed-loop pole maps. These activities also fall under the category of discovery and active learning.

A fourth scenario is the use of the VCL as a support tool for homework assignments. The flexible nature of the VCL and the comprehensive nature of its analysis and simulation resources can effectively support the instructor's plan for engaging the students in active learning while completing homework assignments. All aspects of modeling, analysis, design, and simulation can be addressed with the aid of the VCL as a supplemental learning resource. In this case, the VCL is used to support active learning.

Finally, from a pedagogical viewpoint the instructor may find it advantageous to expose students to new concepts in a sequential fashion. The modular structure of the VCL framework allows the instructor to enable only selected parts of the software, thus preventing the students from experimenting prematurely with advanced features. For example, students could be presented with a VCL version where the controller is locked in Manual mode, and where the Simulation window is enabled. The student are then asked to excite the plant with diagnostic input signals, such as steps and sine waveforms, and use the resulting response to produce and validate a plant model. Once this task is completed, the instructor can release a version of the VCL that is identical, except that the updated version allows the controller to be switched into Auto mode. The new assigned exercise could consist of tuning the controller by selecting appropriate parameters and assessing the resulting closed-loop response using the Simulation window.

## CONCLUSIONS

The key elements required in an architectural description of a VCL have been identified as a

conceptual guide for developing tools that successfully meet the needs of flexible and active learning. An example VCL specialized for the control of a DC motor meets most of the architectural requirements, realizing a high level of user interactivity while remaining a versatile modular platform that can be modified to accommodate other plants.

Although the DC-Motor VCL represents an effective and elegant realization of the proposed VCL paradigm, it does not necessarily satisfy all the desired architectural features. One conceivable shortcoming is the need to install an appropriate browser plug-in, which reduces the tool's ability to provide service to all Web-enabled client computers. This, however, is a common limitation of interactive Web-based systems, and the plug-in alternative selected in the DC-Motor VCL is a good compromise within the context of the currently available technology.

In contrast to the case of other virtual laboratories created using specialized programming languages such as Java, the DC-Motor VCL presents the student with a wide array of analysis and simulation options without making external calls to other supporting pieces of software. In that sense, the VCL is a self-contained tool because it does not require that either the student's machine or the Web-server computer maintain external control-engineering support software. All the numerical simulation computations, analysis calculations, interface management, and Web-interactivity support tasks are carried out by a single software tool, namely LabVIEW.

Finally, even though developed on a proprietary software platform, the proposed DC-Motor VCL offers a very high level of user-interactivity while preserving the ability to allow the instructor to customize the base-case design to suit other physical plants. The modular structure of the VCL allows a developer to create new simulations that take advantage of the features and interface of the framework proposed. It is possible to model a wide range of plants without sacrificing any of the interactive features of the virtual control lab. The reusability of the DC-Motor VCL described here is fairly straightforward when the new plant of interest has only one manipulated variable and either one or two process variables. An extension to the case of multiple-input/multiple-output (MIMO) plants using an analogous procedure is easily envisioned provided that the instructor has available a base-case MIMO VCL that complies with the architectural constraints described in the second section.

## REFERENCES


1. T. F. Junge and C. Schmid, Web-based remote experimentation using a laboratory-scale optical tracker, *Proceedings of the American Control Conference*, Chicago Illinois, June 2000.
2. D. Gillet, C. Salzmann, R. Longchamp and D. Bonvin, Telepresence: an opportunity to develop practical experimentation in automatic control education, *European Controls Conference,* Brussels Belgium*,* July 1997.
3. D. Gillet, F. Geoffroy, K. Zeramdini, A. V. Nguyen, Y. Rekik and Y. Piguet, The cockpit: An effective metaphor for Web-based experimentation in engineering education, *International Journal of Engineering Education,* **19**(3), (August 2002), pp. 389–397.
4. D. Gillet and G. Fakas, eMersion: A new paradigm for Web-Based training in engineering education, *International Conference on Engineering Education*, Oslo, Norway, 2001, 84B-10.
5. A. Bhandari and M. H. Shor, Access to an instructional control laboratory experiment through the World Wide Web, *Proceedings of the American Control Conference*, Philadelphia, June 1998.
6. Ch. Schmid, T. I. Eikass, B. Foss and D. Gillet, A remote laboratory experimentation network, *1st IFAC Conference on Telematics Applications in Automation and Robotics*, Weingarten, Germany, July 24–26, 2001.
7. H. Latchman, C. Salzmann, S. Thotapilly and H. Bouzekri, Hybrid asynchronous and synchronous learning networks in distance education, *International Conference on Engineering Education*, Rio de Janeiro, Brazil, 1998.
8. B. Armstrong and R. Perez, controls laboratory program with an accent on discovery learning, *IEEE Control Systems Magazine*, February 2001, pp. 14–20.
9. Java control module, Georgia Institute of Technology, http://dot.che.gatech.edu/information/research/issicl/che4400/javamodule.html.
10. Control tutorials for Matlab, University of Michigan. http://www.engin.umich.edu/group/ctm/.
11. J. W. Overstreet and A. Tzes, Internet-based client/server virtual instruments designs for real time remote-access control engineering laboratory, *Proceedings of the American Control Conference*, San Diego, 1999.
12. C. Schmid, The virtual control lab VCLab for education on the Web, *Proceedings of the American Control Conference*, Philadelphia, June 1998.
13. O. D. Crisalle and H. A. Latchman, Virtual control laboratory for multidisciplinary engineering education, NSF Award No. DUE-9352523, proposal funded by the National Science foundation under the Instrumentation and Laboratory Improvement Program, (1993).
14. D. Gillet, C. Salzmann, H. Latchman, O. Crisalle, Recent advances in remote experimentation *Proceedings of the American Control Conference*, Chicago Illinois, June 2000.
15. X. Vilalta, Denis Gillet, Christophe Salzmann, Contribution to the definition of best practices for the implementation of remote experimentation solutions, *IFAC Workshop on Internet Based Control Education IBCE'01*, Madrid, Spain, December 12–14, 2001.
16. B. Kuo, *Automatic Control Systems*, 7th edn, Prentice Hall, (1995).

17. W. Luyben, *Process Modeling Simulation and Control,* 2nd edn, McGraw-Hill, (1990).
18. K. Ogata, *Modern Control Engineering,* 4th edn, Prentice Hall, (2002).
19. B. Ogunnaike and W. H. Ray, *Process Dynamics, Modeling, and Control*, Oxford University Press, (1994).
20. G. Franklin, J. D. Powell and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, 4th edn, Prentice Hall, (2001).
21. D. Seborg, T. Edgar and D. Mellichamp, *Process Dynamics and Control*, 2nd edn, Wiley, (2004).
22. N. Nise, *Control Systems Engineering,* 4th edn, Wiley, (2004).
23. R. Stefani, B. Shahian, C Savant and G. Hostetter, *Design of Feedback Control Systems,* 4th edn, Oxford University Press, (2002).
24. National Instruments Corp., 11500 N. Mopac Expressway, Austin, TX 78759-3504, USA, www.ni.com.
25. J. Sánchez, S. Dormido, R. Pastor and F. Esquembre, Interactive learning of control concepts using easy Java simulations, *IFAC Workshop on Internet Based Control Education IBCE'04*, Grenoble, France, (2004).
26. S. Guelich, S. Gundavaram and G. Birznieks, *CGI Programming with Perl*, 2nd edn, O'Reilly, (2000).
27. C. C. Ko, B. M. Chen, and J. Chen, *Creating Web-Based Laboratories*, Springer-Verlag (2004).
28. S. C. Chapra and R. P. Canale, *Numerical Methods for Engineers*, McGraw-Hill, New York, (1990).
29. P. C. Wankat and F. S. Oreovicz, *Teaching Engineering*, McGraw-Hill, (1993).

**Christopher S. Peek** is a Research Assistant at the Chemical Engineering Department of the University of Florida, Gainesville, FL, USA, where he is pursuing the Ph.D. Degree specializing in control engineering. He received his B.S. degree from the University of Virginia in 2003 and joined the graduate program in August of 2004. His research interests include the optimization of control systems, and the design of advanced predictive control structures.

**Samuel Dépraz** received his Diplôme d'Ingénieur (M.S. degree) in 2004 from the Mechanical Engineering Department of the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. He carried out the studies required for his Diploma Thesis at EPFL and at the University of Florida. His interests lie in the areas of automatic control the design of industrial production systems. He is an avid enthusiast of aerospace science and engineering, and holds a private pilot license.

**Oscar D. Crisalle** is Professor of Chemical Engineering at the University of Florida, Gainesville, FL, USA. He received his B.S. degree from the University of California, Berkeley (1982), his M.S. degree from Northwestern University (1986), and his Ph.D. degree from the University of California, Santa Barbara (1990). All his degrees are in chemical engineering. His research interests focus on robust control analysis and design, synthesis of predictive controllers, and nonlinear control analysis, with applications to industrial processing operations such as microelectronics and pulp-and-paper manufacture. Dr. Crisalle has a strong interest in engineering education, and his pedagogical efforts have been recognized through several awards, including the University of Florida Teacher of the Year Award (2001), the Outstanding Teacher Award (1996) granted by AIChE Student Chapter of the University of Florida, and the University of Florida Teaching Improvement Award (1994).

**Denis Gillet** is MER (Associate Professor) at the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. He received his Ph.D. degree in control systems in 1995 from EPFL. His research interests include optimal and hierarchical control systems, distributed eLearning systems, sustainable interaction and real-time Internet services. Dr. Gillet received the 2001 Recognition Award for Innovations and Accomplishments In Distance and Flexible Learning Methodologies for Engineering Education granted by the International Network for Engineering Education and Research (iNEER).