A High School-Level Course in Feedback Control

A Matlab-Based Introduction Requiring Only Algebra and Trigonometry

JORGE CORTÉS and WILLIAM B. DUNBAR

The purpose of this article is to describe a control education and outreach effort being undertaken at the University of California, Santa Cruz (UCSC). The goal of our effort is to provide an introductory course on feedback control that is accessible to high school students and first-year undergraduate students. In the early stages of our effort, we found ample motivation in the literature for creating such a course. In 1998, an article from the "NSF/CSS Workshop on New Directions in Control Engineering Education" [1] made the following recommendations regarding needed reform in undergraduate control education:

- "to provide practical experience in control systems engineering to first-year college students to stimulate future interest and introduce fundamental notions like feedback and the systems approach to engineering"
- "to encourage the development of new courses and course materials that would significantly broaden the standard first introductory control systems course at the undergraduate level."

In 2003, a panel on future directions in control, dynamics, and systems provided a renewed vision of challenges and opportunities, along with recommendations to agencies and universities to ensure continued progress in areas of importance to the industrial and defense base [2]. One of the five primary recommendations is that the community and funding agencies invest in "new approaches to education and outreach for the dissemination of control concepts and tools to nontraditional audiences. As a first step toward implementing this recommendation, new courses and textbooks should be developed for both experts and nonexperts." The panel also recommended the integration of software tools such as Matlab into these courses.

Outreach motivation has also been provided from outside the control field. Ten years ago, feedback control was identified by the National Science Education Standards as being fundamental to understanding systems, and systems in turn was identified as a unifying concept for K–12 science education [3]. In 2005, the need for curricular material that motivates middle school and high school students to pursue advanced work in science and mathematics in the United States was underscored by the Committee on Prospering in the Global Economy of the 21st Century, a subcommittee of the National Academy of Sciences, National Academy of Engineering, and the Institute of Medicine [4].

This article presents an overview of an introductory course on feedback control amenable to traditional and nontraditional audiences. The course material is based on fundamental concepts in dynamical systems, modeling, stability analysis, robustness to uncertainty, feedback as it occurs naturally, and the design of feedback control laws to engineer desirable static and dynamic response. The course also includes an introduction to Matlab, provides Matlab exercises to reinforce concepts, and concludes with the design and application of a controller to achieve wall tracking with a kinematic robot experiment. The only prerequisite for the course is high school algebra, including basic trigonometry. During a four-week period in the summer of 2005, the course was taken by a group of 17 talented high school students, ranging from ninth grade to 11th grade. By the end of the course (∼30 hours of lecture time), each student had successfully implemented a wall-tracking controller on a robot called Robobrain. The students acquired a basic understanding of the principles of feedback control and its importance in applications. The students also displayed a strong motivation to learn more advanced mathematics and engineering subjects.

We focus on the course content as well as our first-hand experiences teaching the class. Our hope is that this information is useful to instructors interested in pursuing new approaches to outreach. All lecture material is freely available online for anyone to use, experiment with, and improve if desired [5]. The target audience for the course includes high school students, first-year undergraduates in engineering and science, and graduate and postdoctoral researchers seeking an introduction to dynamics and control. The material can be used to initiate collaboration between control theoreticians and biologists, for example, by helping the biologists learn the language and principles behind feedback control. By design, the course material provides *explicit motivation* for learning more advanced mathematics (including linear algebra, calculus, and differential equations) and physics. Since the only mathematical prerequisite is high school algebra and basic trigonometry, modeling is done entirely in discrete time and kept to low dimensions.

1066-033X/07/$25.00©2007IEEE

References [6]–[8] describe precollege outreach efforts that focus on the experimental design and control of robots, with the objectives of building robotic platforms and testing basic programming, engineering, and robotic concepts. The study [9] reports on a graphical-user-interface (GUI)-based approach using LabView and LEGO Mindstorm robots to introduce engineering design, embedded computer control, sensors, and software platforms to incoming college freshman. In contrast to [6]–[9], robotics is not the focus of our course. Instead, the power of abstraction is stressed in the lectures, and the robotic platforms used in the final lectures are employed as a venue for the modeling and control design concepts.

Our success with the high school course has led to a new one-quarter undergraduate course at the Baskin School of Engineering UCSC titled "Computer Engineering 8: Robot Automation: Intelligence through Feedback Control." While most control courses are designed for third- and fourth-year undergraduates [10], [11], our course is offered to first-year undergraduates. In a similar vein, a curriculum that focuses on systems and control for undergraduates at the Norwegian University of Science and Technology is described in [12]. Their first-year course, "Introduction to Computerized Control," covers more advanced topics than our crash course, beginning with continuous-time modeling using ordinary differential equations. A similar first-year course is described in [13], with calculus as the mathematical prerequisite.

At the other end of the prerequisite spectrum, the effort described in [14] introduces feedback control to freshmen at the University of New Haven using little mathematical modeling. Instead, a GUI-based approach is taken using LabView software and four applied control experiments that students can choose from. An advantage of this application- and GUI-based approach is that the variety of experiments is likely to entice more students to pursue control systems and engineering, while giving them broader insight into what control can and cannot do. In contrast, an advantage of including mathematical modeling from the beginning, as in our material and in [12] and [13], is that control design and analysis are linked to math subjects the students are taking or will soon be taking.

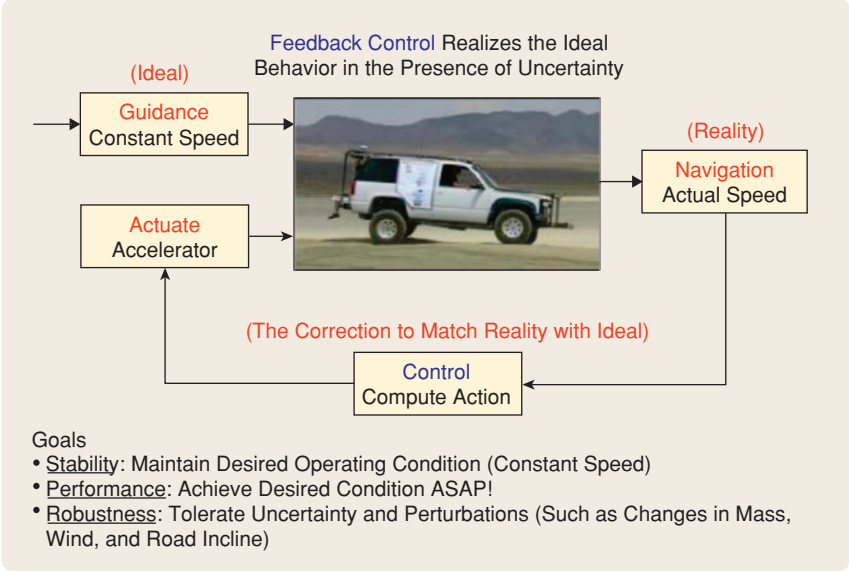

FIGURE 1 Introductory slide presenting feedback control in the context of guidance, navigation, and control of an autonomous off-road vehicle. This slide presents control as a mechanism for correcting the system response to match a desired response using the familiar objective of vehicle cruise control. This example helps introduce the concepts of stability, performance, and robustness. The photo is of Caltech's 2004 DARPA Grand Challenge autonomous vehicle entry, courtesy of Richard Murray.

OVERVIEW OF LECTURE MATERIAL

The material is presented in nine lectures, ranging in length from one to three hours, depending on the complexity of the topics and the ability of the audience. The first lecture is a set of presentation slides. Each of the remaining lectures consists of two parts, specifically, a 15–20-min slide-based overview followed by 45–160 min of white-board lecture and Matlab exercises. Lectures are structured interactively, alternating between short explanations of important concepts given to the whole class and time allocated for the students to solve exercises individually or in pairs. The material assumes that each participant has in-class access to a computer with Matlab installed.

Lectures 1 and 2: Introduction to the Course and Matlab

Lecture 1 is a high-level introduction to feedback control, showing that feedback commonly occurs in nature and in engineering (see Figure 1). The introduction also gives an overview of the mathematics and analysis tools utilized in the course as well as a preview of the robotic platform Robobrain. Lecture 2 provides an introduction to Matlab, including variable assignment, vectors, plotting, m-file functions, and loop-functions for plotting data.

Lectures 3 and 4: Introduction to Discrete-Time Dynamics

Without a background in differential equations, the introduction to dynamics and modeling is facilitated by considering systems in discrete time. To avoid the need for linear algebra, the models considered must also be low dimensional. Therefore, one- and two-dimensional linear and nonlinear discrete-time models are considered in lectures 3 and 4. In the context of

these models, basic high school algebra is sufficient.

Dynamics are introduced with the scalar, discrete-time real-valued map

$$x_{k+1} = f(x_k), \tag{1}$$

where x_k denotes the state of the system at the kth time step. To understand causality, the *orbit* $\{x_0, x_1, x_2, \ldots, x_N\}$ is defined as the sequence of numbers that arise from a given initial point x_0 and by evaluating (1) N times. We then define what it means for f to be linear ($f(x) = ax$) and nonlinear ($f(x) = \cos x$, for example). The concept of *fixed point* (equilibrium) is then defined, and the quantitative and qualitative behavior of (1) near fixed points is explored in the case of linear f. In turn, we qualitatively define *stability* and *attractivity* of a fixed point of (1) by examining orbits for a set of initial conditions near the fixed point. Qualitatively, we define stability of a fixed point as "the property that orbits stay as close to the fixed point as you specify, as long as the orbit starts close enough to it." Likewise, attractivity of a stable fixed point is defined as "the property that orbits approach the fixed point, eventually reaching it." We also qualitatively define *unstable* fixed points and present simple examples to demonstrate each concept. We found that these definitions were sufficient to introduce the concepts of stability and attractivity. Moreover, the students were able to apply these definitions in the context of open-loop and closed-loop analysis of models studied in later lectures.

As a preview of the complexity that is possible with (1), the logistic map [15] is examined, where $f(x) = rx(1-x)$ and $r \in [0, 4]$. For a test of their Matlab skills, students are asked to generate a plot that shows the limiting behavior of orbits as the parameter r is varied. The instructions given in the notes of Lecture 4 are given in Figure 2.

The plot resulting from the algorithm described in Figure 2 illustrates chaos. The exercise thus promotes applied learning of writing for-loops and functions in Matlab, while generating enthusiasm for more programming opportunities and advanced dynamics. Most students like the idea that they are capable of creating chaos!

Lecture 5: Introduction to Modeling

Models of systems are presented as a tool for mathematically predicting how a system behaves under a variety of conditions. Based on the previous material, students can think of a model as a function f that generates an orbit closely matching the actual evolving behavior of a given system. At the outset, the concepts of uncertainty and robustness are impressed on the students. Although models are never perfect, the hope is that they are good enough for a close match to reality and, eventually, for control design and analysis. The sidebar "An Excerpt from Lecture 5" shows the language we use to introduce these fundamental concepts.

The lecture continues with a two-dimensional predator-prey model used to predict how hare and lynx populations influence one another over time. The example is the first leap into modeling in more than one dimension, extending the students' Matlab function-writing skills to generate orbit plots for the two populations. This example is followed by the three-dimensional kinematic model of the Robobrain robot. The model is given by

$$x_{k+1} = x_k + \Delta u_k \cos\theta_k, \tag{2}$$

$$y_{k+1} = y_k + \Delta u_k \sin\theta_k, \tag{3}$$

$$\theta_{k+1} = \theta_k + \Delta v_k, \tag{4}$$

where (x, y) denotes position in space, θ denotes the orientation of the robot, and Δ is the sample period. Definitions for *state, control inputs, disturbances, parameters*, and *outputs* are given with examples for both models. For example, v_k and u_k, which are the rotational and translational velocities of the robot, respectively, correspond to the two control inputs. Since the robot model has control inputs, the concept of fixed point is extended to depend on constant values of control values.

Lecture 6: Introduction to Feedback Control

Lecture 6 introduces model-based feedback control design and analysis. The advantages of using models and simulation for control design and

Name: Orbit diagram algorithm
Goal: Plot orbit diagram of logistic equation

The objective is to have a figure where many orbits of the logistic map are plotted, with one orbit for each value of the parameter r between 3.4 and 4. The x-axis of the figure corresponds to values of r, while the y-axis corresponds to values of the orbits.

1) Set $i = 0$.
2) Set $r = 3.4 + i$.
3) Iterate the logistic map for 200 cycles (after 200 cycles, the system should settle down to its eventual behavior—the settling portion of the response is called the *transient*) starting from $x_0 = .6$.
4) Once the transients have decayed, plot many points, say $x_{201}, \ldots, x_{400}$, versus the current value of r in the figure.
5) Set $i = i + 0.005$. If $i = 0.6$, exit the algorithm. Otherwise, return to step 2.

FIGURE 2 Orbit diagram algorithm. As part of the introduction to discrete-time dynamics, students are asked to generate a plot of the limiting behavior of the orbits of the logistic map. These instructions serve as their guide.

analysis, rather than trying different controllers on the real system, are emphasized. For example, we show the students the advertisement in [16] for Matlab and Simulink by The MathWorks. The ad identifies the savings and success of prediction-based (simulation-based) control design in the case of the Mars rovers, for which zero test flights to Mars are possible.

Based on the mathematics we have presented so far, feedback control is introduced in this lecture as a means of *shaping the dynamics*. In particular, control is shown to allow one to reassign the fixed point(s) of a model as desired, while ensuring that these desired fixed points are stable and attractive. The model in (1) is reformulated to include the control input u_k as

$$x_{k+1} = f(x_k, u_k). \tag{5}$$

Next, the *unforced* or *open-loop* dynamic model of the cruise-controlled car is reexamined, by setting $u_{\text{eng,R}} = 0$ for all k in (S1). Students are asked to calculate and analyze the fixed-point speed v_{eq} in the absence of any road incline, yielding convergence to $v_{\text{eq}} = 0$ (due to the friction term) from any initial speed. For nonscalar examples, simulations (orbit calculations), rather than analysis, are the sole means for determining stability since students do not know matrix algebra.

Continuing with the cruise-control example discussed in lectures 4 and 5, students are asked to recalculate the fixed point with the control in (S1) defined as $u_{\text{eng},k} = K(v_{\text{des}} - v_k)$, where v_{des} is the desired speed of the car when the cruise controller is working. Assuming a constant, possibly nonzero incline disturbance, the resulting equilibrium velocity v_{eq} is given by

$$v_{\text{eq}} = \frac{K}{b+K} v_{\text{des}} + \frac{1}{b+K} u_{\text{hill}}.$$

This example allows the students to *engineer* a fixed point by choosing the control. The students find that, the larger the value of K, the closer the steady-state speed approaches its desired value and the less influence the road incline has on the steady-state speed. By calculating orbits, students can also examine the stability and attractivity of the resulting fixed point v_{eq}.

Lecture 7: Feedback Control of an Inverted Pendulum

This lecture provides a case study in control design and simulation-based analysis. A normalized discrete-time model of a pendulum with angle θ_k and torque control u_k is

$$\frac{1}{\Delta^2}(\theta_{k+1} - 2\theta_k + \theta_{k-1}) = -\sin\theta_k + u_k,$$

where Δ is the sample period. The objective is to design u_k to make the pendulum upright position $\theta_{\text{eq}} = \pi$ a stable and attractive fixed point. The model is converted into the form of (5) since all analysis to this point relies on difference equations in first-order form. Students are often mystified when first introduced to a change of variables, and this example provides a good opportunity to demonstrate the technique. Specifically, in (5), the state x_k and function f become two dimensional, defined as $x_k = (z_k, y_k) = (\theta_k, \theta_{k-1})$ and

$$f(x_k, u_k) = \begin{bmatrix} 2z_k - y_k - \Delta^2 \sin z_k + u_k \\ z_k \end{bmatrix}.$$

An Excerpt from Lecture 5

Models are never perfect at predicting the response of real systems. At best, a model produces a good approximation of the actual behavior of a real system. One reason for the imperfection of models is *uncertainty*, which can arise from various sources. For example, we usually do not know with total exactness the values of the parameters of the system, such as the mass or the friction coefficient. Another reason is that many dynamic processes are just too difficult to model exactly, such as the global carbon cycle in the atmosphere of Earth. Although approximations are always required in modeling, a principle that has the power to save us is called *robustness*. Robustness is the ability of a system to be *insensitive to measurement, parameter, and environmental variations or uncertainties*. Let us consider examples of such uncertainties, in the case of cruise control.

EXAMPLE 1 (SOURCES OF UNCERTAINTY IN CRUISE CONTROL)

Recall the cruise control model (presented in Lecture 4)

$$v_{k+1} = v_k + \frac{\Delta}{m}[-bv_k + u_{\text{eng},k} + u_{\text{hill},k}]. \tag{S1}$$

The variable v_k is car speed, $u_{\text{hill},k}$ is road incline, $u_{\text{eng},k}$ is accelerator control input, all at the kth time step. Δ is the sample period, m is the mass, and b is a friction coefficient. Measurement uncertainty occurs when measurements of v_k are not exact. For example, your speed sensor is actually giving you $v_k + \sigma_k$, where σ_k is a variable that changes randomly, contaminating the speed measurement. Parameter uncertainty arises if we do not know the mass m exactly. This is the case as fuel is being burned causing a decrease in mass, and we are not taking this into account since it is assumed that m is constant for all time. Lastly, an example of environmental uncertainty is $u_{\text{hill},k}$. At best, we might be able to say how large this term is over a certain time period, but we do not know in general the exact value of the road incline, now or in the future. □

Robustness is one of the most useful properties of control. Think again about the cruise controller of your parents' car. The controller automatically adapts the accelerator setting so that the system is insensitive to climbing uphill or going downhill, and it does so *with no exact knowledge of the true incline of the road traveled!* In this way, the cruise controller makes the actual behavior of the car robust to changes in the road incline conditions.

Students are then asked to find the constant torque u_{eq} such that the fixed-point state is the upright position, that is, $x_{eq} = (\pi, \pi)$. Next, the control is defined as $u_k = u_{eq} + \tilde{u}_k$, and we explain that the first term u_{eq} reassigns the equilibrium as desired, while the second term $\tilde{u}_k$ makes the equilibrium stable and attractive. The desired fixed point can be stabilized by choosing the parameters K_1 and K_2 in the feedback control

$$\tilde{u}_k = K_1(\theta_{eq} - z_k) + K_2(\theta_{eq} - y_k).$$

Using the Matlab function `pendulum.m`, students examine the response of the pendulum to various control parameter choices and thus can identify parameters that result in stability and attractivity. In the latter portion of Lecture 7, an integrator (summation term for discrete-time control) is also explored in the cruise-control model (S1) to remove the steady-state error in the equilibrium speed v_{eq} shown above. Although the students are now capable of writing a function such as `pendulum.m`, this preprogrammed routine allows us to maintain the desired pace in the course.

Lecture 8: Analysis and Simulation of the Robobrain Model

Lecture 8 begins by identifying the equilibrium states for the robot model (2)–(4), which are all constant $(x_{eq}, y_{eq}, \theta_{eq})$, with associated control input $(u_{eq}, v_{eq}) = (0, 0)$. Students are asked to decide whether the equilibrium states are stable and attractive, using the qualitative definitions for stability and attractivity covered in Lecture 3. Once they decide that each equilibrium state is stable but not attractive, we begin to explore feedback controllers that make a chosen equilibrium state both stable and attractive.

Ultimately, the control objective for the Robobrain robot is autonomous wall tracking. Specifically, starting from an arbitrary initial configuration, the robot must be programmed to find a wall and then track the wall at a desired separation distance with constant velocity. Before tackling this objective, the students are guided through a series of exercises involving the robot model, in which the concepts of open-loop versus closed-loop control, uncertainty, and robustness to disturbances are reinforced through simulations. Students create two Matlab programs, `robobrain.m` and `robobrainDist.m`. The first program plots the orbit of the robot from a generic initial configuration, given a specific open-loop control signal that produces a sinusoidal state orbit. Students realize that the open-loop controls are specified ahead of time, without looking at the actual evolution of the Robobrain robot during its motion. The program `robobrainDist.m` is meant to show the risks associated with this open-loop strategy. Students are asked to simulate the model with the same control signal and initial conditions as before but with a slight disturbance in the equations. The resulting orbit is no longer sinusoidal due to the effect of the disturbance. In our experience, this exercise convinces students of the need for feedback for control to be effective for tracking a desired orbit.

Lecture 9: Feedback Control of the Robobrain Robot

Lecture 9 consists of three parts. First, students are reintroduced to the discrete-time model (2)–(4) and the control inputs u_k, v_k, that correspond, respectively, to the forward and angular velocity of Robobrain. In the second

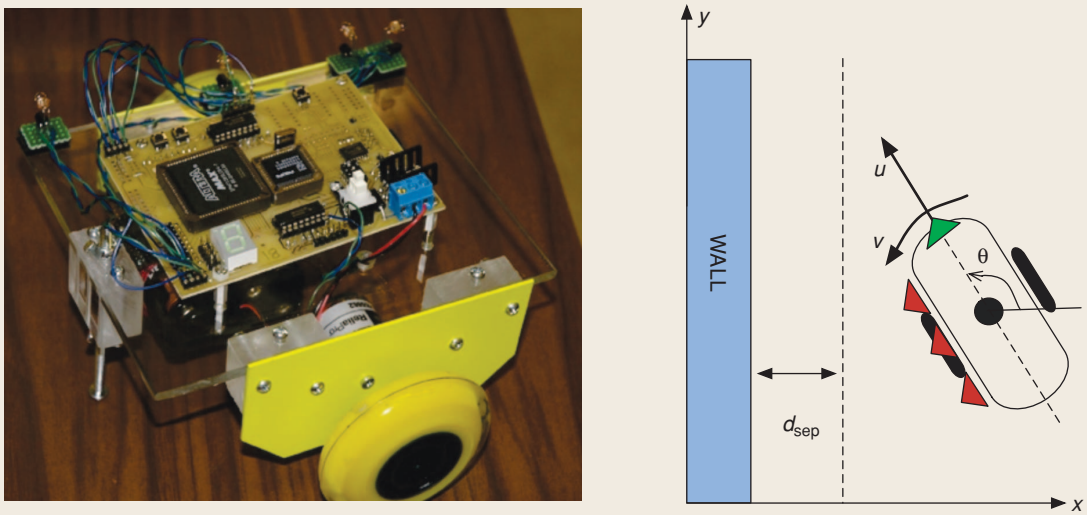

FIGURE 3 (a) The Robobrain robot developed at UCSC and (b) a schematic drawing of the robot. The three (red) triangle shapes on the left side of the robot represent infrared sensors used to determine distance and heading relative to the wall, while the single (green) triangle shape on the front represents an infrared sensor used to determine the distance to the wall in the forward direction. The objective is to have the robot track the wall at a constant separation distance d_{sep}.

part, students are guided through the task of defining a strategy for the robot to find the wall and then make it turn to track the wall at a specified separation distance d_{sep}. The measured distances to the wall from each infrared sensor (see Figure 3) are defined by $d_{l,f}$, the distance from the left-front sensor; $d_{l,m}$, the distance from the left-middle sensor; and $d_{l,b}$, the distance from the left-back sensor. Students are asked to figure out how to convert the three scalar distance measurements $d_{l,f}$, $d_{l,m}$, and $d_{l,b}$ into the state values x and θ of the robot and write a Matlab program to compute them, assuming a long, straight wall.

The logic used to find the wall is simple. The robot first moves forward at a constant speed with $v = 0$ and $u = v_{\text{nom}}$, where v_{nom} is 50% of the maximum velocity of each wheel. The robot continues to move until the front sensor reads $d_f \approx 2d_{\text{sep}}$, where d_{sep} is the desired separation distance. The robot then turns clockwise slowly in place with $(u, v) = (0, -0.05v_{\text{nom}})$. The robot keeps turning until the three left distances $d_{l,b}$, $d_{l,m}$, and $d_{l,f}$ are within 0.2 cm of each other. After this procedure, the robot uses the three left sensors to keep track of position x and heading θ relative to the wall.

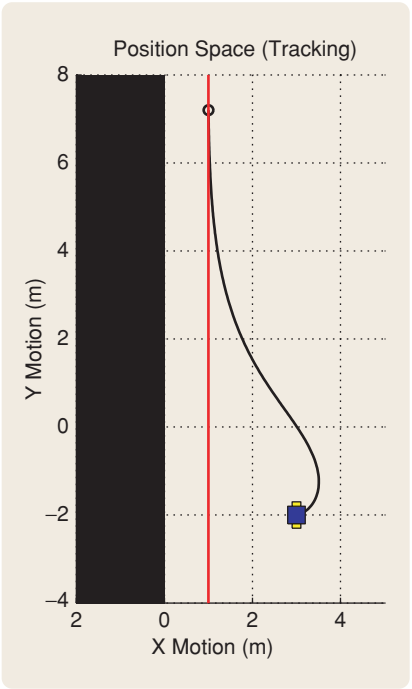

FIGURE 4 Simulation of feedback control for keeping a desired separation distance from the wall at a constant speed. The initial condition is $x_0 = 3$ m, $y_0 = -2$ m, and $\theta_0 = 0$ rad, the nominal wall-tracking velocity is $v_{\text{nom}} = 1$ m/s, and the desired separation is $d_{\text{sep}} = 1$ m. The control gains are $k_p = 1/3$ and $k_d = 1$. The sample period is $\Delta = 0.05$ s.

The third part of the lecture consists of designing and analyzing the controller. Students are given the discrete-time proportional-derivative wall-tracking controller

$$u_k = v_{\text{nom}}, \tag{6}$$

$$v_k = k_p(x_k - d_{\text{sep}}) + k_d \frac{x_k - x_{k-1}}{\Delta}. \tag{7}$$

Students are asked to modify `robobrain.m`, created in Lecture 8, to include the feedback controller (6)–(7). The new function asks for the initial configuration x_0, y_0, θ_0 of the robot, the sample time Δ, the number of iterations N, the desired separation distance d_{sep}, and the control gains k_p, k_d, and then outputs the orbits x, y, θ and the elapsed time. Figure 4 illustrates a simulation of this controller.

To conclude the lecture, students are asked to tune the values of the controller parameters k_p and k_d. Although knowledge of linear algebra would allow the students to perform this tuning analytically, this background is not assumed. Instead, the students are encouraged to select some values manually and, using the newly defined program to simulate the system, decide whether the resulting wall-tracking controller makes $x = d_{\text{sep}}$, $\theta = 0$ a stable, attractive fixed point of (2) and (4) for x and θ. Although students keep trying parameter values until the desired behavior is obtained, we have observed that students become disappointed at not being able to ensure that the control yields the desired wall-tracking behavior. Guessing parameters and simulating the model is, in other words, lame. When told that more advanced math, specifically calculus and college-level linear algebra, would enable them to determine the control parameters analytically to get the desired result, the students are enthusiastic to learn these subjects.

Until now, no experimental equipment has been present in the classroom. To put an exclamation point on the power and joy of feedback control requires experimentation, and the students are anxious to work with real robots. In the remainder of this section, we briefly describe the experimental platform Robobrain. Although other kinematic robot platforms such as LEGO Mindstorms are consistent with the modeling and control objective presented so far, a different placement of the position sensors on the robot would require a reparameterization of the position measurements (x_k, θ_k) as a function of the sensor data. We note that successful uses of LEGO in control courses are described in [17] and [18].

Once the control parameter tuning task is completed, the students receive a Robobrain robot ready to be programmed. The boards on each Robobrain robot are created by UCSC undergraduate students as part of a computer engineering senior design project. The remaining components of the robot are purchased and assembled by student aides. The components of each robot cost about US$200. Additional details regarding Robobrain's hardware and software are available online at [19].

Students implement the controller on the Robobrain robot in a hands-on fashion. Each student is instructed to transcribe their parameter choices into the robot program, which is run in the C programming language. Aides assist the students with this task since most students have little or no prior exposure to C programming. The course finale consists of requiring students, working in teams of two or three, to program their robot to track a curved indoor path. The teams compete to see which team robot can track the curved path for the longest distance. Figure 5 shows typical results. Although the robot is tracking a curved wall, the controller is designed

with a straight wall in mind. The performance of the robots to actually track a constant distance from the curved wall demonstrates the robustness of the controller. The students also observe that at places on the wall where the curve is "tighter," the controller does not perform as well, often bumping into the wall or losing distance measurements altogether when the wall is no longer in the range of the infrared sensors. The reduced tracking performance at places where the wall turns more sharply demonstrates the limitations of the controller and sensors as well as the inherent coupling of sensor limitations and controller stability, robustness, and performance. Naturally, it is at this point in the course that the students clearly see feedback control as a tangible and effective engineering tool.

IMPLEMENTATION OF THE COURSE MATERIAL: COSMOS

In the summer of 2005, a four-week course was taken by a group of talented high school students in grades 9–11. The course was part of the California State Summer School for Mathematics and Science (COSMOS) program held at the UCSC campus. COSMOS is a selective four-week summer residential program for young scholars with demonstrated interest and achievement in math and science [20]. The program is located on four University of California campuses: Davis, Irvine, San Diego, and Santa Cruz. The student population of COSMOS 2005 at Santa Cruz was composed of 76 female and 75 male students, with 37% of the total from underrepresented groups. Each COSMOS student enrolls in one of seven topical clusters for the duration of the program. Each cluster is comprised of two courses designed and instructed by UCSC faculty, lecturers, researchers, or graduate students. Cluster 2 was comprised of the courses "Nanotechnology" and "Robot Automation: Intelligence through Feedback Control," the latter being our course in feedback control. Our course typically met for 90–120 min each day for 16 days, in which the nine lectures were covered. Minimal assignments outside of the classroom were required.

The overall experience was positive, and the lecture format worked well. Students responded enthusiastically to the introduction to Matlab and our lecture model consisting of short theoretical explanations combined with simple computer exercises. The general introduction on the first day of class prepared them to understand the main ideas of feedback control. We often referred to this lecture during subsequent lectures, placing the specific content of the class into the big picture. We believe that students appreciated and benefited from the introduction as well as the brief high-level introductions at the beginning of each lecture. In the last week of class, each student team received a Robobrain robot ready to be programmed and tested. Working in teams, the students implemented the wall-tracking controller developed in Lecture 9 and thus had the chance to put into practice the lessons learned during the course.

On the last day of class, the COSMOS organization mixed different clusters so that students from each cluster could give a presentation to the students from another cluster, sharing what they had learned in their courses. Our cluster was paired with the Stars, Sight, and Science cluster. The students from our course formed three groups and prepared a Powerpoint presentation, an excerpt of which is shown in Figure 6. The first team explained what feedback control is, drawing examples from real-life systems and robotics. The second team discussed modeling, analysis, and control of physical systems from a controls perspective. This team placed special emphasis on the ability of feedback to deal with uncertainty, measurement errors, and making engineering systems autonomous. The third team focused on their experience in developing the controller for the Robobrain model with the mathematical tools learned and then on implementing the controller on the actual robot. Each student presented an equal part of the presentation. The presentation confirmed that the students had received an increased awareness of the importance and ubiquity of feedback control.

In future offerings, we plan to integrate the Robobrain robots into the lectures sooner. We anticipate that presenting the students with

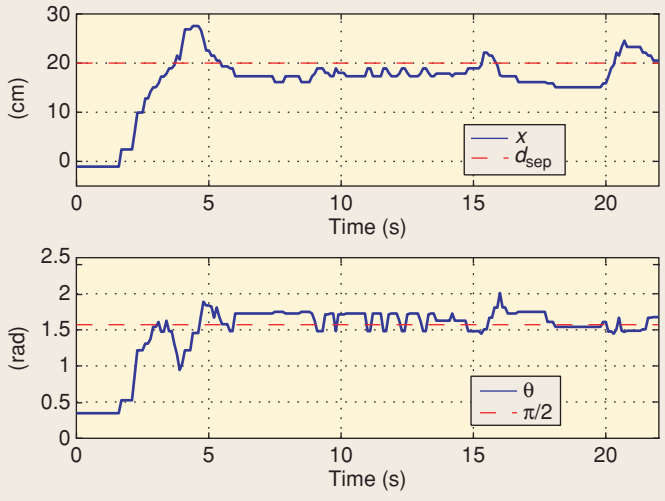

FIGURE 5 Experimental results of feedback control for keeping the Robobrain robot at a desired separation ($x = d_{sep}$) from the wall at a constant speed ($u_k = v_{nom}$) and constant heading ($\theta = \pi/2$). The feedback control law (6)–(7) is turned on at time $t = 2$ s. The students select the control parameters by means of simulation-based trial and error.

hardware can motivate them to master the theoretical content of the course. Students are eager to understand the math that can allow them to make the robot operate autonomously.

This experience provides the basis for an introductory undergraduate course in the engineering curriculum at UCSC. In the fall of 2006, the course "Computer Engineering 8—Robot Automation: Intelligence through Feedback Control" was offered, introducing first-year undergraduate students to Matlab, programming, dynamics, feedback control, and robotics. We believe that this course can serve as an excellent recruiting and retention tool. Additionally, the experience can motivate students to pursue more advanced mathematics, programming, and hardware courses in the computer engineering, electrical engineering, applied mathematics, and statistics curricula.

Feedback from Students

At the end of the course, we conducted an online survey concerning student satisfaction to assess the overall impact of the course material and the suitability of the lecture notes, Matlab exercises, and experimental implementation format. According to the students' evaluations and the parents' comments, the experience was as challenging and rewarding for the students as it was for us. Parents expressed the feeling that the COSMOS experience will shape their children's future careers.

We asked the students to rank the course with respect to four different themes (quantitative), and by answering some qualitative questions. Of the 17 enrolled students, we received 14 evaluations. Table 1 summarizes the results of the quantitative rankings.

Below are some student responses to the qualitative questions:

» **What did you like most about the course?**

- "It was a challenge so I (was) never bored. I also enjoyed discovering all of the practical applications of the material we were learning. It was also an experience I never could have had in high school."
- "I liked most how first we learned the idea, then the math, then the application of it all. I really liked how it all came together eventually, it was really clear at the end."
- "I liked working on the programming of the Robobrain and I guess that includes the work leading up to the Robobrain. The last day of class was amazing, but that was only because we had to work up to it. If the Robobrain was just handed to us than I don't think it would have been as good an experience."

» **How much and in what ways did the handouts help your learn the material in this course?**

- "The handouts were key in the course—they explained and helped to reinforce the reasoning and purpose of all that was taught in the class. The handouts gave meanings

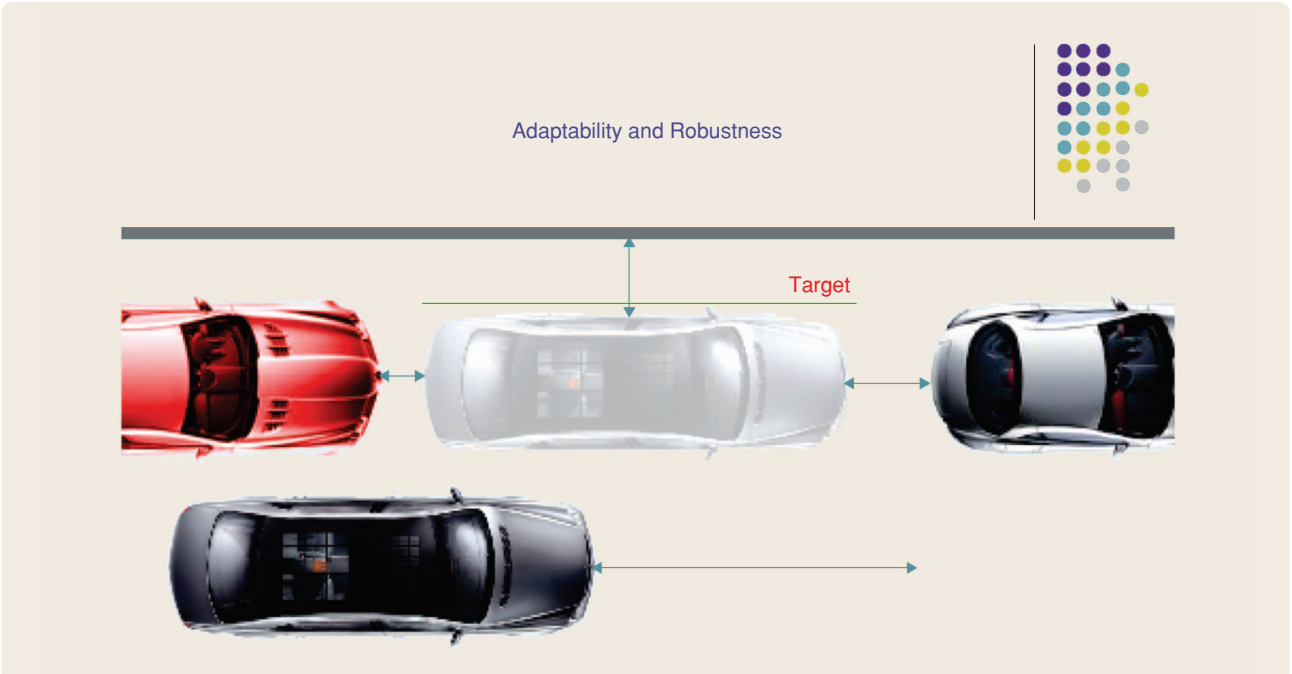

FIGURE 6 Excerpt from the final presentation prepared by COSMOS students on the lessons learned during the course. The slide comes immediately after having motivated uncertainties that might occur in real life such as unknown "ground conditions," "sensor accuracy," and "quality of parts." The students' comments on this slide are "Feedback means that you can take your system (the car), drop it into different environments and conditions, such as wider or narrower spaces or different starting distances from the curb, and feedback control will continue to correct for differences to make your system follow your desired model as closely as possible."

TABLE 1 Quantification of student satisfaction with the course. For each topic, students rated the course by choosing one of five possible rankings.

	Poor	Fair	Satisfactory	Very Good	Excellent
Clarity and understandability	–	–	–	10	4
Preparation and organization	–	–	1	7	6
Course as learning experience	–	–	–	5	9
Will recommend course to friends	–	–	1	1	12

and definitions to the terms, assigned tasks to help the student have a better understanding of the material, Matlab, and making robots intelligent."

- "The handouts were very helpful. I liked how they were written and how they introduced the ideas, explained, and then had us do a task to learn them. Referring back to the handouts was helpful as well."

» **Please tell us the concepts covered in the course that you understood the most and the least.**

- "Most of the concepts I had a good grasp on, although there were a few I was a little hazy about. Equilibrium, steady state, fixed points, stability, I understood really well."
- "I understood all of the algebra and altering the equations very well. I also understood why we change the equations like we do. I got the graphing very well too, both writing the programs to produce the graphs and reading the graphs to tell what they meant."
- "The concepts covered in the course that I understood the most included the mathematical concepts and the entire necessity of feedback control for robotics. The concepts I understood least probably included much of the later math for Robobrain, which to me was a little rushed. I'm sure if we went over the math a few more days, I would be able to understand."

Ten Months Later . . .

In an effort to gauge the impact of our course on these exceptionally bright students, we sent an e-mail asking them to reflect on their experience. Here are three responses:

» A female student headed to UC San Diego said "Right from the start I could tell the robotics course was going to be challenging. In fact, in the beginning I was scared it was going to be too challenging for me to handle. But as the classes went on, and I began to put in more effort (like reading the class notes the night before until I understood them), I realized I could handle it. This was the first lesson I learned from your course, just how to succeed in something that is challenging... Another big thing your course did was introduce me to Matlab. At many of the schools I was applying to, I heard Matlab mentioned. I know the introduction I received in your course will help me greatly when I have to learn it in greater depth... Also, just the whole concept of feedback control has popped up everywhere, and I know it will continue popping up since I am going into the field of bioengineering. Feedback control was applicable to some of the 'smart medicines' I discussed in my COSMOS final project."

» A male student headed to Harvard said "I definitely used the robotics skills I learned (programming in C, working with a development kit + microcontroller platform, basic electronics) in my Intel STS project and of course used them in my personal engineering projects for fun. I don't think the course really influenced what I want to study in college, since I've always been headed toward the engineering track; it only further directed me toward it. The overall impact: a greater familiarity and knowledge with robotics and how to engineer them."

» A male student headed to UC Berkeley said "Matlab will help in college, but I haven't been able to use it yet... (the course has) raised my interest and helped me decide to do either nanotechnology or robotics as a career choice."

CONCLUSIONS

We have summarized the contents of an introductory course on feedback control developed at UCSC. The course material, available at [5], is based on fundamental concepts in dynamical systems, modeling, stability, robustness, and the design of feedback control laws. We have reported our positive experience in using the material in a summer course for motivated high school students during the summer of 2005. We believe that researchers from systems and control theory will find the material useful for a variety of activities, ranging from education and outreach to high school and undergraduate students, to interdisciplinary collaborations with scientists from other disciplines.

An interesting study performed in [21] examines how novices learn feedback control concepts, in particular how they gain a perspective on control that includes abstraction, so that the

concepts are not tied to a specific problem description. The study involves the use of a GUI-based software that allows the students to connect and simulate a set of standard functional objects (for example, sensor, actuator, setpoint unit) that exist in many control applications. As part of an introductory engineering course, the software was made available to a group of high school students for about 45 min per day for five days. A key finding of the study is that "the idea of a signal is a core concept in an expert's conception of feedback systems and is intricately tied to the definition of the functional components that make for a feedback control system." The study further reported that, by the end of the course, students still had trouble with the concept of signals.

In our course, we used Matlab and described vectors, which most students had seen in a math or physics course, as a sequence of measurements taken at snapshots of time. Most students needed a few days and several exercises to connect the list of numbers (vector) in Matlab with the evolution of a model of a real physical system. Once they grasped this idea, we were able to describe the use of models for prediction and control design. We never formally introduced the word signal (suggestions for introducing this concept in primary and secondary education can be found in [22]) and did not attempt to generalize the vector description to more general scenarios, for example, continuous time, since this step seemed unnecessary. As a future direction, our material, based largely on algebra and Matlab exercises, as well as a GUI-based approach to control such as the software described in [23], can further help students comprehend the concepts and gain intuition for control design.

Finally, a completely new approach to engineering education is being undertaken at the Olin College of Engineering, in Needham, Massachusetts; see [24]. The school opened to incoming freshman in 2001, and the emphasis of the curriculum is on entrepreneurship, interdisciplinary learning, and teamwork. Feedback control is considered an "important engineering concept" in the Olin curriculum [24, p. 32]. The article also shows pictures of students and instructors working on autonomous vehicles, robots, and nonlinear and chaos theory graphs. This overlap between the topics covered in our short course and the topics offered at Olin [24] is reassuring, since Olin College is already demonstrating an impact on the need for engineering education reform.

ACKNOWLEDGMENTS

This material is supported in part by NSF CAREER Award ECS-0546871. We thank Richard Murray for inspiring our effort and a good portion of the material in the slide introductions for each lecture. We sincerely thank Bill Thompson, the high school instructor assigned to our cluster, for his help and enthusiasm. We thank our student aides Noah Wilson and Daniel Garalde, who assembled the Robobrain robots. We would also like to thank all of the students who took the course and provided invaluable feedback. Their enthusiastic response to the course far exceeded our best expectations. Teaching feedback control to them was truly a joyful experience.

REFERENCES

[1] P. Antsaklis, T. Basar, R. DeCarlo, N.H. McClamroch, M. Spong, and S. Yurkovich, Eds., "NSF/CSS Workshop new directions control engineering education," National Science Foundation and IEEE Control Systems Society, Tech. rep., 1998 [Online]. Available: http://robot0.ge.uiuc.edu /~spong/workshop

[2] R.M. Murray, K.J. Astrom, S.P. Boyd, R.W. Brockett, and G. Stein, "Future directions in control in an information-rich world," *IEEE Contr. Syst. Mag.*, vol. 23, no. 2, pp. 20–33, 2003.

[3] *National Science Education Standards*. Washington, DC: National Research Council, 1996.

[4] "Rising above the gathering storm: Energizing and employing America for a brighter economic future," Committee on Prospering in the Global Economy of the 21st Century: An Agenda for American Science and Technology, National Academy of Sciences, National Academy of Engineering and Institute of Medicine, Washington, DC: National Academy Press, Tech. Rep., 2005 [Online]. Available: http://www.nap.edu/catalog/11463.html

[5] "A crash course on feedback control," [Online]. Available: http:// www. soe. ucsc. edu/~jcortes/controlcrashcourse/

[6] L. Greenwald and J. Kopena, "Mobile robot labs," *IEEE Robot. Automat. Mag.*, vol. 10, no. 2, pp. 25–32, 2003.

[7] I.R. Nourbakhsh, E. Hamner, K. Crowley, and K. Wilkinson, "Formal measures of learning in a secondary school mobile robotics course," in *Proc. IEEE Int. Conf. Robotics Automation*, vol. 2, 2004, pp. 1831–1836.

[8] H. Mukai and N. McGregor, "Robot control instruction for eighth graders," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 20–23, 2004.

[9] A. Nagchaudhuri, G. Singh, M. Kaur, and S. George, "LEGO robotics products boost student creativity in precollege programs at UMES," in *Proc. 32nd ASEE/IEEE Frontiers Education Conf.*, vol. 3, 2002, pp. S4D-1–S4D-6.

[10] R.M. Murray, S. Waydo, L. Cremean, and H. Mabuchi, "A new approach to teaching feedback," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 38–42, 2004.

[11] K.J. Astrom and R.M. Murray. "Feedback Systems: An Introduction for Scientists and Engineers." Preprint, 2005. [Online]. Available: http://www.cds.caltech.edu/~murray/amwiki

[12] J.T. Gravdahl and O. Egeland, "New undergraduate courses in control," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 31–34, 2004.

[13] T.E. Djaferis, "Automatic control in first-year engineering study," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 35–37, 2004.

[14] S. Daniels, D. Harding, and M. Collura, "Introducing feedback control to first-year engineering students using LabVIEW," in *Proc. American Society Engineering Education, Annu. Conf. Exposition*, Portland, OR, 2005, Session 2161 [Online]. Available http://www.asee.org

[15] S.H. Strogatz. *Nonlinear Dynamics and Chaos*. Boulder, CO: Westview, 1994.

[16] Back cover page, *IEEE Spectr.*, Aug. 2005 [Online]. Available: http://mathworks.com/mbd

[17] P.J. Gawthrop and E. McGookin, "A LEGO-based control experiment," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 43–56, 2004.

[18] J.M. Rieber, H. Wehlan, and F. Allgöwer, "The ROBORACE contest," *IEEE Contr. Syst. Mag.*, vol. 24, no. 5, pp. 57–60, 2004.

[19] N. Wilson, "Robobrain User Manual," [Online]. Available: http://www.soe.ucsc.edu/~noahness/papers/RoboBrain-Manual_rev2.pdf

[20] The California State Summer School for Mathematics and Science (COSMOS) [Online]. Available: http://www.ucop.edu/cosmos and http://epc.ucsc.edu/cosmos/

[21] J. Ma, "A case study of student reasoning about feedback control in a computer-based learning environment," in *Proc. 29th ASEE/IEEE Frontiers Education Conf.*, San Juan, Puerto Rico, 1999, pp. 12d4-7–12d4-12.

[22] D.S. Bernstein, "Introducing signals, systems, and control in grades K through 12," *IEEE Contr. Syst. Mag.*, vol. 23, no. 2, pp. 10–12, 2003.

[23] M. Casini, D. Prattichizzo, and A. Vicino, "A student control competition through a remote robotics lab," *IEEE Contr. Syst. Mag.*, vol. 25, no. 1, pp. 56–59, 2005.

[24] E. Guizzo, "The Olin experiment," *IEEE Spectr.*, vol. 43, no. 5, pp. 30–36, 2006.

AUTHOR INFORMATION

Jorge Cortés received the Licenciatura degree in mathematics from the Universidad de Zaragoza, Spain, in 1997 and his Ph.D. degree in engineering mathematics from the Universidad Carlos III de Madrid, Spain, in 2001. Since 2004, he has been an assistant professor in the Department of Applied Mathematics and Statistics at UC Santa Cruz. He previously held postdoctoral positions at the Systems, Signals, and Control Department of the University of Twente, and at the Coordinated Science Laboratory of the University of Illinois at Urbana-Champaign. His research interests focus on mathematical control theory, distributed motion coordination for groups of autonomous agents, and geometric mechanics and geometric integration. He is the author of *Geometric, Control and Numerical Aspects of Nonholonomic Systems* (Springer Verlag, 2002).

William B. Dunbar (unbar@soe.ucsc.edu) earned a B.S. degree in engineering science and mechanics from Virginia Tech in 1997, an M.S. degree in applied mechanics and engineering science from UC San Diego in 1999, and a Ph.D. in control and dynamical systems from Caltech in 2004. He is currently an assistant professor in the Department of Computer Engineering at UC Santa Cruz. His research areas include distributed model predictive control, air traffic control, and applications of control to problems in biophysics and sequencing. He can be contacted at Department of Computer Engineering, University of California at Santa Cruz, 1156 High Street, Mail Stop: SOE3, Santa Cruz, CA 95064 USA.

What Is Your Favorite Book on Classical Control?

Responses to an Informal Survey

DANIEL E. DAVISON, JIE CHEN, SCOTT R. PLOEN, and DENNIS S. BERNSTEIN

As part of this special section on classical control, we asked control experts in academia and industry about their favorite textbook on classical control. The survey was informal and not meant for statistical purposes, but it did result in some interesting replies. Due to the large number of responses, not all are included here, and many responses have been edited for length.

If you did not contribute to this survey, we invite your responses for a follow-up article. Interesting and memorable stories about your favorite classical control textbook, course, or instructor are most welcome.

CURRENT TEXTBOOKS FIRST PUBLISHED MORE THAN 35 YEARS AGO

Not surprisingly, many responses refer to textbooks that have been around for decades but are still in print. Some favor the textbook written originally by Dorf and now co-authored with Bishop:

» I have used the textbooks by the following authors over the past 20 years: Dorf and Bishop, Kuo, Ogata, Franklin, Powell, and Emami-Naeini, D'Azzo and Houpis, Philips and Harbor, and Nise. Each has strengths and weaknesses. However, among all of them, I prefer the first one. My main reasons are as follows: The book covers the control topics in a logical fashion; each chapter explains the subject in a simple way; the derivations of the formulas are complete and convince the students; tables, figures, and illustrations are excellent and useful for better understanding of the subjects; the numerous examples and problems represent all fields; the book shows the students the applicability of control systems to many facets of real life. The book makes the students aware that control systems are multidisciplinary. The modeling of many diverse systems taken from different applications are either derived or referenced for further investigation. This approach makes it easy for the students to realize that real systems can be modeled by differential equations, transfer functions, or a state variable description. The students are convinced that models are useful for analysis and design phases. There is a fair balance between design based on transfer function and state-space methods. The integration of Matlab and Simulink in the book over the past few years has enhanced the quality of this book. The division of exercises, problems, design problems, and Matlab problems make the students comfortable learning step-by-step theoretical concepts from simple situations to more complex cases. *Bahram Shafai, Northeastern University, Boston, Massachusetts*

» I feel slightly dated seeing that my favorite text falls in the "more than 35 years ago category"! In any event, I learned classical control in the fall of 1977 from Dorf's book in a graduate-level course at the University of Michigan in the Computer, Information, and Control Engineering Program. The course was taught by Bill Powers before he left to head up research at Ford. I was fascinated by the beauty and simplicity

1066-033X/07/$25.00©2007IEEE