IV. Conclusion

I have presented a new FET ac model that clearly demonstrates the similarity between the ac behavior of the BJT and the FET. Use of this model can save classroom instruction time, and more important, increase the "comfort level" of students when confronted with two different types of transistor amplifiers.

Robotics Laboratory Exercises

Eric Krotkov

***Abstract*—We report new laboratory exercises in robotic manipulation, computer vision, artificial intelligence, and mechatronics, four areas that are central to any robotics curriculum. The laboratory exercises supply the student with hands-on experience that complements classroom lectures and software development. Through this experience, the student confronts the hard realities of robot systems and learns to deal with them. Such hands-on experience is essential for a sound robotics education, because many critical lessons about the real world can only be learned through personal experience.**

I. Introduction

Robotics has been characterized as the field where artificial intelligence meets the real world. This characterization, whether or not it is correct, successfully captures one of the key features of robots: they operate not in the abstract, but in the real, physical world, in factories and hospitals, in oceans and outer space.

Many robotics educators have recognized that "meeting the real world" cannot be taught entirely in the classroom. To teach the student about the physical interactions involved in robotic sensing, thinking, and acting, they have developed a variety of pedagogical tools. In the area of robotic sensing, Richard developed an interactive system that allows the student to gain practical experience with image processing and machine vision operators [3]. In robotic thinking, Vira and Tunstel implemented a symbolic computation package [5]. In robotic acting, Inigo and Morton developed a simulator of the dynamics of an industrial robot [1], and Vibet assembled elements to teach robot control [4]. In integration of robotic mechanisms with electronics, Pota defined an interdisciplinary project to build a flexible manipulator [2].

These efforts represent significant advances in robotics education. However, there appears to be great demand for further practical educational materials, especially those that address the physical interaction with the environment that lies near the heart of robotics.

The contribution of this paper is to report new laboratory exercises that provide the hands-on experience that is critical for robotics education. These laboratory exercises supply the student with hands-on experience that complements classroom lectures and software development. Through this experience, the student confronts the hard realities of robot systems—realities such as sensor noise, nonzero calibration residuals, low fidelity of first-order models, and real-time constraints—and learns to deal with them. Such hands-on experience is essential for a sound robotics education, because many critical lessons about the real world can only be learned through personal experience.

The goal of the paper is to provide an overview of the laboratory exercises and the required facilities. As an overview, the paper will not provide enough practical information to replicate the exercises. We will distribute detailed documentation to interested practitioners.

This paper proceeds as follows. In the next section, we describe the facilities of the Robotics Education Laboratory. In Section III, we present practical laboratory exercises in manipulation, computer vision, artificial intelligence, and mechatronics. We conclude by discussing future expansion of the curriculum and laboratory activities.

II. Facilities

The Robotics Education Laboratory (REL) occupies a 7×8 m space adjacent to classrooms and clusters of academic computing facilities. Founded in 1991, the REL contains equipment for use in existing undergraduate and graduate courses, independent study projects, and proposed new courses. In this section, we describe the laboratory facilities for robotic manipulation, machine perception, mechanical prototyping, and general infrastructure.

For computer vision, the laboratory contains a number of components for image acquisition and analysis. The basic setup includes a table-mounted 1×1.25 m optical breadboard (Fig. 1), and a "cage" built from slotted angle. A charge-coupled device (CCD) camera (with either a 16-mm lens or a zoom lens) can be clamped to the cage or attached to the optical breadboard. The video output can be viewed on a color monitor. Halogen spotlights can be clamped to the cage at arbitrary positions and orientations; variable transformers permit students to vary independently the intensity of the lamps.

For robotic manipulation, the laboratory contains three Microbot Teachmover manipulators (Fig. 1). These arms have five primary motion freedoms, plus an additional freedom for opening a parallel jaw gripper. The arm reach is 45 cm, and arm velocities range from 0 to 180 mm/sec. The device is driven by stepping motors, with power transmitted to the joints by cables, and controlled by a 6502 microprocessor. Each manipulator is attached to a workstation that communicates over a serial line to the manipulator, and provides a software development environment.

For mechanical prototyping, the laboratory contains a complete set of Fischer-Technik kits. These kits include a variety of structural building blocks, motors and gears, sensors, and electronics that students can assemble into functional prototypes of robotic devices and systems. With these kits, students can design and construct functioning models, gaining experience and insight in designing robotic mechanisms.

For general purposes, the laboratory supports several electronic instruments. A small computer (PC with five expansion slots) with an analog/digital I/O board allows students to sample 16 input signals at 100 kHz with 12-bit resolution. Bench instruments include digital multi-meters, a triple power supply, and others. The chief support equipment in the current inventory includes a band saw, a drill press, and an assortment of hand tools.

III. Laboratory Exercises

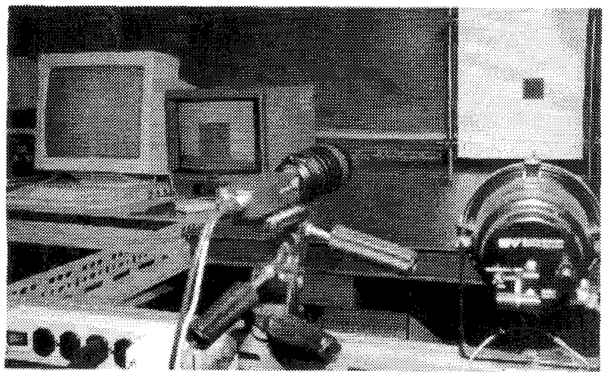
A. Manipulation

Robotic Manipulation is a junior/senior-level course covering the foundations of robotic manipulation and the fundamentals of robot motion. Most of the material is theoretical in nature, including the

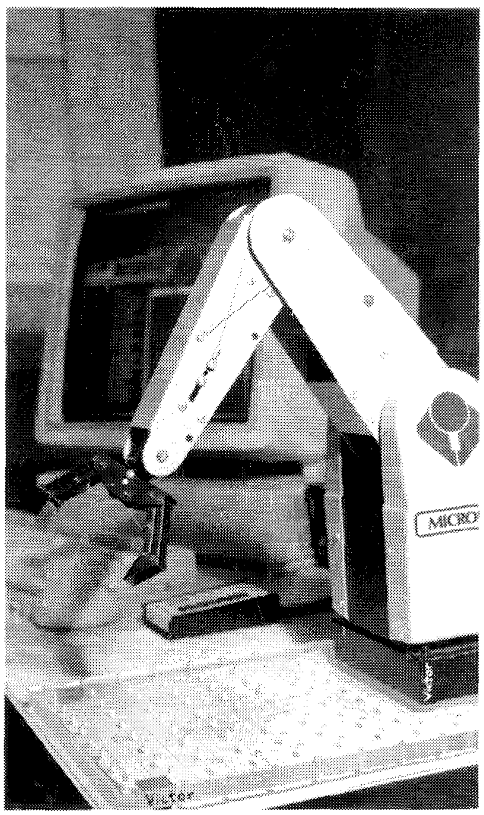
Manuscript received January 19, 1994; revised June 3, 1994. This work was supported in part by a grant from the AT&T Foundation.

The author is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Publisher Item Identifier S 0018-9359(96)02087-0.



(a)



(b)

Fig. 1. (a) Setup for image acquisition and (b) five-dof manipulator.

low level of kinematics and dynamics of articulated mechanisms, the middle level of trajectory generation and control, and the high level of planning and sensing.

No theory is adequate for predicting how a robot will behave in even a simple environment, however, so the course has three laboratory assignments that use the robot arms. These assignments require the student to demonstrate working robot programs for building various toy structures, and serve to confront the student with the nondeterministic vagaries of friction, noise, and uncertainty. These effects cannot be modeled at the theoretical level of this class, but must still be accounted for in the student projects. This knowledge is important, because coping with these practical issues is a prerequisite to making a robot do anything useful.

Exercise 1—Pick-and-Place: The student programs the manipulator to construct an arch from three blocks: two short blocks and one long block. The positions and orientations of the blocks are given, except that the robot has to figure out which is the long block, presumably using the grip switch, whose state is checked by a sensor-based conditional plan.

Exercise 2—Part Acquisition in the Presence of Uncertainty: The student programs the manipulator to find and pick up a known object with an initially unknown position and orientation.

Exercise 3—Path Planning: The student programs the manipulator to plan a path through a field of obstacles and move a gripped object to a user-specified goal location. There, the robot disengages the object. Constraints are imposed that result in a planar path planning problem, in which the part is a rectangle moving around rectangular obstacles.

Tournament: The course culminates with an optional robot tournament. One year, the objective was for pairs of neighboring robots to build a toy tower while simultaneously trying to frustrate the competing robot. Another year, the objective was to throw a projectile as far as possible.

B. Computer Vision

Computer Vision is a junior/senior-level course covering two- and three-dimensional computer vision. The emphasis is on physical, mathematical, and information-processing aspects of vision, rather than biological, psychological, or cognitive aspects. The course covers the fundamental techniques and background of machine vision, from basic digital image processing to symbolic image understanding. Topics include image formation, early processing of images, region analysis, line drawing understanding, "shape-from-X" methods, three-dimensional analysis, vision architecture, and applications.

The student is expected to assimilate the theoretical background and to apply it in programming projects such as segmentation, object recognition, and photometric stereo. Although the course gives a solid foundation, the programming projects address only artificial problems using canned data. Without having to take data themselves, the student never has to think about problems like illumination, shadows, focus, depth of field, resolution, aliasing, calibration, or sensor placement. There is much to learn from working directly with a sensor rather than using a small set of sample images.

Exercise 1—Image Center: The student uses a zoom lens and CCD camera to acquire images of a rectangular object with different focal length settings. The student then analyzes these images to identify graphically the image center.

Exercise 2—Perspective Projection: The input is a set of images of the same square lying at different distances from the camera. First, the student writes and executes a program to compute the area of the square in an image. Next, the student derives the relationship between projected area and the distance from the lens center to the target. Then, the student compares the observed and predicted areas.

Exercise 3—Illumination Effects and Temporal Noise: The student acquires a sequence of images of the same scene taken under different lighting conditions. Then they write programs to histogram and compute statistics of the image intensity distributions.

C. Artificial Intelligence

Artificial Intelligence is a junior/senior-level course covering problem spaces, search, game playing, predicate logic, knowledge representation, reasoning, natural language, learning, robotics, and more. Much of the material is abstract and theoretical in nature. The goal of the laboratory for this course is to ground the material, and elaborate it in a realistic setting. The setting happens to be robotic, although it

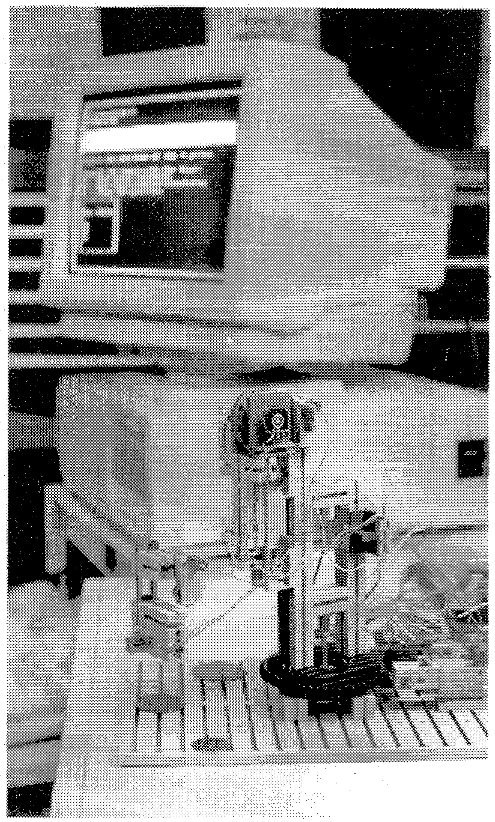


Fig. 2. Simple manipulator solving Tower of Hanoi problem.

could take many other forms. Thus, this laboratory differs somewhat from the manipulation and vision laboratories, which necessarily use robotic devices.

Exercise 1—Tower of Hanoi: The Tower of Hanoi problem can be stated as follows. There are three poles labeled A, B, and C. On pole A, n discs of radius $1, 2, \cdots, n$ are piled up so that the disc of radius i is the ith from the top. The problem is to move all n discs to pole C so that again the disc of radius i is the ith from the top. Only one disc may be moved at a time, from any pole to any other one, subject to the restriction that no disc may ever rest above a smaller disc on the same pole.

One of the first assignments in the course (and many courses like it) is to write a program to solve this problem. To expose students to manipulation, this exercise requires students to adapt their program to move real discs. The apparatus is a simple manipulator built from the Fischer-Technik kits, which uses an electromagnet to handle metal disks (Fig. 2).

Exercise 2—Learning Inverse Kinematics: The inverse kinematic problem can be stated as follows: Given the desired position and orientation of the end-effector, calculate all possible sets of joint angles that attain the desired position and orientation. This is a nontrivial problem, especially in higher-dimensional spaces. To appreciate the issues involved, the student uses an apparatus that includes a workstation connected to a manipulator and to a camera. Attached to the manipulator's end effector is a light-emitting diode (LED) that is easily detected in the images captured by the camera.

First, the student derives the relationship between the Cartesian coordinates of a point on the manipulator and the image coordinates of the projection of that point. This involves first deriving the inverse kinematics for two links of the manipulator, and then deriving the relationship between image coordinates and manipulator joint angles. The required mathematical effort is significant.

Second, the student designs a neural network back-propagation learning procedure that learns the inverse kinematic mapping from Cartesian to image coordinates. They acquire training data including the image location of the LED target and the joint angles of the arm. Given the pixel coordinates, the learned procedure generates the joint angles that cause the LED to appear at the given pixel coordinates.

Finally, the student is asked to articulate the key differences between the analytical approach and the neural network approach, considering such issues as program complexity, robustness to image noise, sensitivity to manipulator model inaccuracies, and execution time.

D. Mechatronics

Mechatronics is a graduate course covering the synergistic integration of mechanism, electronics, and computer control to achieve a functional system. The course covers the spectrum of topics required for developing an actual mechatronic device: mechanisms, structures, fasteners, actuators, sensors, actuator drivers, sensor interfaces, power sources, wiring, microprocessors, peripherals, real-time programming, control, and control architectures. Because of the emphasis upon integration, the course centers on laboratory projects in which student teams configure, design, and implement mechatronic devices.

The most recent project challenged students to build a micro-robot to operate in an automated "warehouse." The robot was required to enter a table-sized structure, locate and pick up two colored blocks, transport each to the appropriately colored shelf, and then exit through the doorway.

The teams met this challenge by building wheeled robots with manipulators for block handling. Each team received a box full of components, including a single-board computer, actuators, sensors, and other electrical and mechanical pieces. Then the students constructed mechanical and electrical assemblies using tools such as band saws, drills, soldering irons, and multimeters. Finally, the students designed and implemented control programs in the language C.

IV. Discussion

In this paper we have presented practical laboratory exercises in robotic manipulation, computer vision, artificial intelligence, and mechatronics. These laboratory exercises provide the hands-on experience that is essential for robotics education.

Currently, we are developing laboratory activities for a graduate course in Robot Control. These include observing robot arm behavior near kinematic singularities, and evaluating robot arm link inertias for low-, medium-, and high-velocity motions. In the near future we expect to formulate practical laboratory exercises for courses in computer-aided manufacturing, and in manufacturing processes.

In the more distant future, our goal for the Robotics Education Laboratory is to integrate the facility with the entire robotics curriculum, increasing the number and type of laboratory exercises routinely conducted by students of robotics.

Acknowledgment

The author would like to thank H. Borysewicz for the laboratory work and the many teaching assistants who contributed to the laboratory exercises. Y. Xu and M. Mason provided the materials on the manipulation course, and P. Muir provided the materials on the mechatronics course.

REFERENCES

[1] R. Inigo and J. Morton, "Simulation of the dynamics of an industrial robot," *IEEE Trans. Educ.*, vol. 34, no. 1, pp. 89–99, Feb. 1991.

[2] H. Pota, "A prototype flexible robot arm—An interdisciplinary undergraduate project," *IEEE Trans. Educ.*, vol. 35, no. 1, pp. 83–90, Feb. 1992.

[3] W. Richard, "An educational image processing/machine vision system," *IEEE Trans. Educ.*, vol. 34, no. 1, pp. 129–133, Feb. 1991.

[4] C. Vibet, "Elements to teach robot control design," *IEEE Trans. Educ.*, vol. 36, no. 1, pp. 45–51, Feb. 1993.

[5] N. Vira and E. Tunstel, "Use of symbolic computation in robotics education," *IEEE Trans. Educ.*, vol. 35, no. 1, pp. 18–31, Feb. 1992.

MIMO Systems—Transfer Function to State-Space

Hemanshu Roy Pota

***Abstract*—In this short note we present a very simple method, useful for classroom teaching, to get a minimal state-space representation (with the exception of systems where there is a pole-zero cancellation) from a transfer function matrix. The method is direct and does not involve the intermediate step of obtaining nonminimal realization—which further requires system reduction routines—and hence is suitable as a compact self-contained topic which has been hitherto neglected in undergraduate linear control theory curriculum.**

I. INTRODUCTION

The undergraduate linear control theory curriculum can be divided into two parts. The introductory part of the curriculum deals with classical frequency domain material and the advanced part is devoted to state-space theory. Single-input single-output (SISO) transfer functions are central to the teaching of the classical frequency domain theory. Considerable part of the curriculum is devoted to deriving state-space canonical forms from the SISO transfer functions [2], [4], [5]. The idea of minimal state order and the associated notions of controllability and observability are discussed as a prelude to the state-space theory. With the state-space theory, the fact that the system under study is SISO or multi-input multi-output (MIMO) is less relevant. Hence, the knowledge of the state-space theory enables the students to attack MIMO control problems, provided a state-space representation is available; this is the case in some practical situations, while in other situations transfer function matrices arise naturally.

Many books on linear systems theory [4], [5] discuss the derivation of a state-space representation from a transfer function matrix. The discussion normally starts with obtaining either block observer or block controller state-space (nonminimal) realization, which is followed by algorithms to obtain minimal realization (both controllable and observable) from these nonminimal realizations. These algorithms [5] are suitable only for a digital computer implementation and are more of an exercise in linear algebra than in controller design. The consequence is that this material has to be left out of classroom teaching (even in standard text books such as [1] and [2], this material is omitted) and the students get little confidence is attacking problems where transfer function matrices arise naturally from the physics of the problem [6]. Most of the practical control systems are multivariable systems, and the students should be taught the connection between a transfer function matrix and its state-space realization in a simple way. Students, when taught this connection using methods given in [4] and [5], lose the physical feel behind the definition of system state and presume that controllability and observability are more important issues than the system representation itself.

In this short note, a simple procedure is given that can be used to get a *minimal* state-space realization (with the exception of systems where there is a pole-zero cancellation) directly from a transfer function matrix by hand calculation, without going through the intermediate nonminimal realization stage. The theory behind this realization is not new [3] and the main motivation for writing this short note arises from the fact that the author could not find a single book where such a simple method is given. This work aims at providing material in a suitable form to enable the inclusion of the realization of state-space from a transfer function matrix in an undergraduate curriculum, giving students a tool to have a first go at multivariable control design for problems such as those that arise in [6].

Manuscript received December 31, 1992; revised June 9, 1993.

The author is with the Department of Electrical Engineering, University College, University of New South Wales, Australian Defence Force Academy, Canberra ACT 2600, Australia.

Publisher Item Identifier S 0018-9359(96)02136-X.

II. TRANSFER FUNCTION TO STATE-SPACE

Let a general multivariable system (l inputs and p outputs) transfer function matrix be given as follows:

$$\begin{bmatrix} Y_1(s) \\ \vdots \\ Y_p(s) \end{bmatrix} = \begin{bmatrix} \frac{n_{11}(s)}{d_{11}(s)} & \cdots & \frac{n_{1l}(s)}{d_{1l}(s)} \\ \vdots & \cdots & \vdots \\ \frac{n_{p1}(s)}{d_{p1}(s)} & \cdots & \frac{n_{pl}(s)}{d_{pl}(s)} \end{bmatrix} \begin{bmatrix} U_1(s) \\ \vdots \\ U_l(s) \end{bmatrix}. \tag{1}$$

Let us consider two examples before giving a general procedure to derive a state-space representation of the above MIMO system (1).

A. Examples

The first example is of a multi-input single-output system and the second example is of a two-input two-output system. In the general method, to be presented later in this paper, every multivariable system (l inputs and p outputs) is first reduced to p multi-input single-output systems. Each of the inputs to these p multi-input single-output systems is an output of a single-input single-output system. The procedure to get a state-space representation for a multi-input single-output system used in the first example can be used to get the final state-space representation. The second example is chosen so that this two-step procedure is made clear to the reader.

Example 1: The transfer function matrix for this example system is

$$Y(s) = \begin{bmatrix} \dfrac{1}{(s+a)(s+b)} & \dfrac{(s+c)}{(s+a)(s^2+ds+e)} \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}. \tag{2}$$

The first step in writing state-space equations for a MISO system is to pull out all the common factors in the denominator polynomials. Notice that the factor $(s+a)$ is common to both the columns. Pulling this term out, the above transfer function matrix can be written as

$$Y(s) = \frac{1}{(s+a)} \begin{bmatrix} \dfrac{1}{(s+b)} & \dfrac{(s+c)}{(s^2+ds+e)} \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}. \tag{3}$$