Experimental Model-Based Control Design Using Multibody Codes

Raffaello D'Andrea
Mechanical and Aerospace Engineering
218 Upson Hall, Cornell University
Ithaca, NY 14853
rd28@cornell.edu
www.mae.cornell.edu/raff

Abstract

In this paper we discuss an on-going project at Cornell University aimed at introducing a significant graphical simulation component in the dynamics and control curriculum, and to expose the students to the interplay between simulations and experiments [1]. This is being achieved by incorporating control experiments from Quanser Consulting, MATLAB control software from the Mathworks, and the Working Model 2D and 3D multibody code software from Knowledge Revolution. The benefits of this approach are both economical and pedagogical; only a limited number of control experiments needs to be purchased and maintained, and it exposes to the students to computer simulation and the relationship between simulations and reality.

1 Introduction

Due to the great advances in computing power, simulation has become an integral part of most engineering disciplines. It is often easier and cheaper to simulate a physical system than to build a working prototype; even if this is not the case, the incremental costs of simulating an altered version of the system are small compared to the costs of modifying a physical prototype. Since the design process is iterative, it does not take many design cycles before the cost of prototyping overtakes the cost of simulation. There is thus a trend to move as much of the design cycle as possible to a simulated environment.

This is certainly true for the design of most high-performance control systems. For many control applications, the design cycle consists of obtaining high fidelity models of the system to be controlled, performing a control design based on reduced order models (for example), verifying the control design on the simulation model, altering the design until the required level of performance is achieved, and then testing out the design on the physical system. A typical example of this process is flight control design. In fact, for these applications there are often different levels of simulation employed, since obtaining simulation time on high fidelity simulators is relatively expensive.

It should be stressed, however, that simulation is only a tool in the design process, and that extensive experimental verification should always be the end result of the design process; the real world is simply too complex to be fully simulated. Thus, an important goal in engineering education is to expose the students to the obvious advantages of simulation, but at the same time to make strong connections between computer simulations and the real devices which are being simulated. The proliferation of powerful computers in education and at home, coupled with the availability of many affordable software packages for control design and dynamic simulation (or multibody simulation), makes it possible to introduce a significant graphical simulation component into the dynamics and control education. In particular, including a substantial graphical simulation component results in the following benefits:

[1] This project is being supported, in part, by the National Science Foundation, Grant No. DUE-9851406

1. It allows the educator to stress the importance of sensitivity and robustness when designing control systems; "what if" scenarios can readily be explored via simulations.

2. It allows the students to explore the tradeoff between high fidelity models and computational complexity; working with a high fidelity model leads to more credible results, but with the price tag of longer simulation run times. The students will thus learn that high fidelity models should only be used after extensive testing on lower fidelity models and/or after traditional control analysis.

3. It exposes the students to the fact that simulation models can never completely describe a physical system (at least tractable models), and that adjustments and refinements will always have to be made to the control design for peak performance.

4. It stresses the importance of incorporating experimental data into the modeling process, and in identifying which aspects of the model are important for control design.

In this paper we discuss an ongoing effort at Cornell University to introduce a significant graphical simulation component in the dynamics and control curriculum, and to expose the students to the interplay between simulations and experiments. This is being achieved by incorporating control experiments from Quanser Consulting [8], MATLAB control software from the Mathworks [6], and the Working Model 2D and 3D multibody code software from Knowledge Revolution [4]. A new course is being being developed around these experiments, entitled "Experimental Model Based Control Design Using Multibody Codes". Limited versions of the experimental setups have, and will be used, in other senior electives at Cornell.

To our knowledge, no other projects of this scope exist; there are several projects, however, which share a similar philosophy. In [2], a laboratory is described where computer simulations are compared to actual measurements as integral parts of an undergraduate mechanics laboratory. It is argued quite convincingly in this paper that simulations should not replace real experiments in the undergraduate curriculum, however fiscally convenient this may be; not surprisingly, students seem to learn more when confronted with real systems than with computer simulations. Similarly, in [10] an undergraduate process control laboratory is described which is very similar in scope; an emphasis is made on system identification, computer aided control design, simulation, and implementation. We are extending these concepts to the Mechanical and Aerospace engineering curriculum, and just as importantly, to expose the students to the use of dynamics simulation software in the control design process.

In Section 2, we discuss the interface between Working Model and MATLAB. In Section 3, we outline how this setup was used in a senior elective, an introductory course in feedback control. In section 4, we discuss a new course being developed whose central theme is the interplay between simulation and experiments.

2 Working Model and MATLAB

Working Model 2D and Working Model 3D are graphical simulation software packages which can be used to simulate a large class of systems governed by Newtonian mechanics, including collisions. Working Model allows one to define models which include masses, springs, dampers, gears, joints, and many other simple mechanical devices, through a graphical interface or by object oriented programming languages such as C++ and Visual Basic. Various physical parameters may be specified, such as friction and collision coefficients. Working Model also allows one to place actuators and sensors anywhere in the model; these actuators may be force or torque actuators, position, velocity, or acceleration actuators (both linear and angular), while the sensors can measure these quantities, in addition to other physical variables. One may also specify general force fields, alter the gravitational constant, include air resistance in the simulation, and change many other "world" parameters. As the names suggest, Working Model 2D is restricted to two dimensional motion, while Working Model 3D can simulate full three dimensional Newtonian mechanics.

One of the key features of Working Model 2D is that it can be interfaced to MATLAB via DDE (Dynamic Data Exchange). MATLAB is a popular software package used throughout academia for control education, and also used extensively in industry. At a user-specified rate, Working Model suspends its simulation, exports specified variables to MATLAB (the "sensor" measurements), runs a user specified set of MATLAB scripts, and imports user specified variables into its simulation (the "actuator" values). This allows a seamless

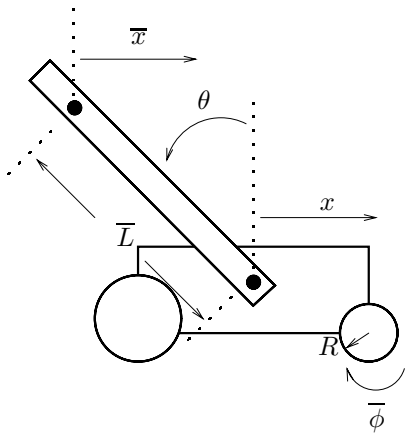

Figure 1: Inverted Pendulum

implementation of "real-time" control; a user need only construct the MATLAB m-files (or mex files) which implement the real time control strategy. Since time is one of the variables which Working Model can export to MATLAB, the control strategies need not be time invariant. The variables can be stored in the MATLAB workspace for debugging and analysis purposes.

The interface to Working Model 3D and MATLAB does not currently exist. We are developing a new interface based on OLE (Object Linking and Embedding) which offers a lot more power and flexibility. One of the goals of this project is a completely seamless interface between Working Model and MATLAB. Currently, the MATLAB interface is restricted to m-files and mex files; we would like to extend this functionality to encompass Simulink and the Real Time Workshop. Working Model would then look to MATLAB just like an actual experiment. One could then implement hardware-in-the-loop experiments with both real experiments and virtual experiments, concurrently.

3 The Inverted Pendulum

In this section we describe one of the assignment used in a senior elective, Feedback Control Systems, which was used as a test-bed for what will be developed in the new course described in Section 4. The class make-up was approximately twenty Electrical Engineering Students, and twenty Mechanical Engineering students. What follows is information essentially extracted from the description of the project.

3.1 Overview

The task is to design a control system for an inverted pendulum. Since this is a first course in feedback control systems, the system parameters are given to the students. In the new course being developed and discussed in Section 4, the system parameters can be identified by physically measuring masses, lengths, etc., by applying test signals and deducing system parameters, and via formal system identification techniques [5]. In addition, due to time constraints and the amount of material which had to be covered in this introductory course, the nonlinear equations of motion are provided to the students. The specific tasks are:

1. construct an appropriate analytical model of the system;
2. design a control system using the techniques developed in the lectures;
3. simulate the control design in Working Model and MATLAB;
4. test the design on the experiment.

The students are expected to iterate on the above steps. For example, a linearized model of the system should be tested against the Working Model simulation to ensure that the model is correct. There will be many iterations between steps two and three; most of the control design will occur at this stage. There may also be limited iteration between two, three, and four; this is further explored below.

3.2 System Details

The inverted pendulum essentially consists of three devices, as depicted in Figure 1. The first is a cart which moves along a track. The second device is an aluminum rod which is pinned to one side of the cart with a hinge, and is thus free to rotate in a plane perpendicular to the gravity vector and the cart track. The third is a motor which turns the driving wheel through a gear box; the driving wheel, in turn, moves the cart up and down the track. The input to the system is the voltage applied to the motor. The output of the system is the horizontal displacement of a point on the rod a distance $\overline{L}$ from where the rod is attached to the moving cart. Note that the experimental setup is actually equipped with two sensors, the cart position and the angular position of the pendulum. Since the students are only exposed to single-input, single-output control design techniques, these two measurements are combined into one, the point on the rod. This actually makes the control design non-trivial, since two of the resulting poles are unstable, one of the zeros is non-minimum phase, and the magnitude and bandwidth limitations described below severely limit where the poles and zeros of the controller can be placed.

The design task is to stabilize the rod in the upright position. This is achieved by providing the appropriate voltage to the motor based on the observed horizontal displacement. The design specifications are the following:

1. Keep the rod in the upright position for *at least* 10 seconds; during this time period, the cart may move up and down the track, and the rod may wobble, but as long as the rod stays above the horizontal for at least 10 seconds, the design specifications will have been met.

2. The maximum gain of the control system (from position in meters to voltage in volts) should be less than 85 dB, and the maximum gain at the sampling rate of 1000 Hz should be less than 40 dB; not meeting these specs will results in motor chatter.

The system variables are the following:

Symbol	Name
V	Voltage applied to motor
F	Force applied to the cart by the motor
τ	Torque delivered by the motor
$\overline{\tau}$	Torque delivered by the driving wheel
ϕ	Motor rotation
$\overline{\phi}$	Driving wheel rotation
x	Horizontal position of the cart
$\overline{\text{x}}$	Measured horizontal displacement of point $\overline{L}$ on the rod
θ	Angular position of the rod

The cart and rod parameters are the following:

Name	Symbol	Value	Units
Cart mass	M_c	914	g
Rod mass	m	210	g
Rod C.O.M. position	L	30	cm
Position of horizontal displacement sensor on rod	$\overline{L}$	40	cm
Radius of driving wheel	R	6.35	mm
Cart Coulomb friction constant	c	0.59	N

The motor parameters are the following:

Name	Symbol	Value	Units
Armature Resistance	R_a	2.6	Ω
Torque constant	K_a	0.0063	N·m/A
Back-emf constant	K_m	9.3	mV· s/rad
Gear ratio	k	3.71	
Rotor inertia	I_{mot}	3.87×10^{-7}	kg · m^2

The equations of motion of the cart and rod are the following:

$$\begin{bmatrix} M+m & -mL\cos\theta \\ -mL\cos\theta & I+mL^2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} mL\dot{\theta}^2\sin\theta \\ mL\dot{\theta}\dot{x}\sin\theta - mgL\sin\theta \end{bmatrix} = \begin{bmatrix} F - c\,\mathrm{sgn}(\dot{x}) \\ 0 \end{bmatrix}, \tag{1}$$

where $M = M_c + I_{mot}k^2/R^2$ is the effective mass of the cart, I is the mass moment of inertia of the rod (about the center of mass), and sgn(·) is the sign function; sgn(y) is 0 if y is 0, 1 if y is positive, and -1 if y is negative.

The equation relating the voltage applied to the motor, the rotational rate of the motor, and the torque supplied by the motor is

$$\tau = K_a \frac{V - K_m\dot{\phi}}{R_a} \tag{2}$$

The maximum voltage that can be applied to the motor is 5 V; both Working Model and the real-time control software will limit the voltage applied to the motor to 5V. This limitation clearly needs to be incorporated in the control design.

The drive wheel rotation is related to the motor rotation by the relation $\phi = k\overline{\phi}$, the torques by $\overline{\tau} = k\tau$.

3.3 Working Model and MATLAB Interface

At each simulation time step, the following variables are written by Working Model to the MATLAB variable workspace:

t time

xbar the horizontal position of point $\overline{L}$ on the rod

Working Model expects the following variable from MATLAB at every simulation step:

voltage the voltage applied to the motor

3.4 Simulation Information

In addition to the standard Working Model interface buttons (RUN, STOP, RESET, etc., which are described in the course Web Site), an ACCURACY button is provided to control the resolution of the simulation. When this button is activated, the Animation Step which is currently set to 1000 frames/second may be changed. This is, in fact, the sampling rate of the real control system. This sampling time may be decreased to speed up the simulation when testing preliminary designs.

The controller is designed in continuous time; the controller is converted to discrete time in the Working Model/MATLAB interface, and is automatically converted to discrete time by the Real Time Workshop toolbox in MATLAB, which is being used for real-time control.

3.5 Implementation

Once a controller which stabilizes the linearized system is designed (using classical control design techniques such as root locus, bode plots, etc.), and which can stabilize the inverted pendulum in Working Model, the students are given two chances to implement the controller on the real system (on separate days).

3.6 Assesment

Most of the design iteration was performed in MATLAB (stabilizing the system and meeting the magnitude and bandwidth requirements). On average, the students iterated five times between the Working Model simulation and their control design; this iteration was necessary to refine their system to work in the presence of the system nonlinearities, such as stiction and the geometric nonlinearity of the pendulum. All of the fifteen groups in the class (the group size varied between two and three), were able to stabilize the real experiment *on their first attempt*. Both the simulation and the experimental setup exhibited a limit-cycle, caused by the stiction non-linearity. The major difference between the simulation and the experiment was a slight motor chatter, which was not captured by the simulation. A more refined model with gears is being developed which will capture that behavior as well. The bandwidth limitations imposed on the controller essentially made these chatter dynamics benign. The reaction of the students was definitely positive; most of them could not believe that their controller, which had never been tried on the real system, actually stabilized the pendulum.

4 Experimental Model-Based Control Design using Multibody Codes

The culmination of our efforts is a new course, with title given above. The course is an introduction to the use of graphical dynamics simulation packages for experimental, model-based control design. Topics covered include: state space methods for control design, sensitivity analysis and robust control, system identification, computer control, and multibody codes. Laboratory projects consist of identifying the dynamics of four mechanical-aerospace systems, constructing models of the systems using graphical dynamics simulation packages, control system design using analytical techniques and simulation, and real-time control implementation. Lecture material includes a thorough treatment of the systems being controlled. The following design cycle is being used for each of the four experiments:

1. Learn the theoretical concepts;
2. Identify the system;
3. Simulate the system; go back to 2 if necessary;
4. Design the control system;
5. Simulate the design; go back to 4 if necessary;
6. Implement the control design; go back to 2 (and perhaps 1) if necessary.

One of the systems has already been described, the inverted pendulum. This experiment will be fitted with a second link, and be made into a doubly inverted pendulum; this will result in a three output, one input system. This will be an excellent opportunity to expose the students to multivariable control techniques. Provisions exist to fit the inverted pendulum with a flexible link. A flexible link can be readily implemented in Working Model by treating it as a chain of pinned joints, as illustrated in Figure 2. This will introduce a new level of complexity in the simulation and the control design; a very detailed model will not only take a long time to simulate, but will make control design intractable. The students will thus learn the benefits of reduced order models.

4.1 Planar Manipulator

One of the major strengths of Working Model is that it handles collisions, a very non-trivial task. We discuss below an experimental setup being constructed which makes extensive use of this capability, and will form the second experiment in the course. The simulation component of this project has already been used in a senior elective, Robotics and Control.

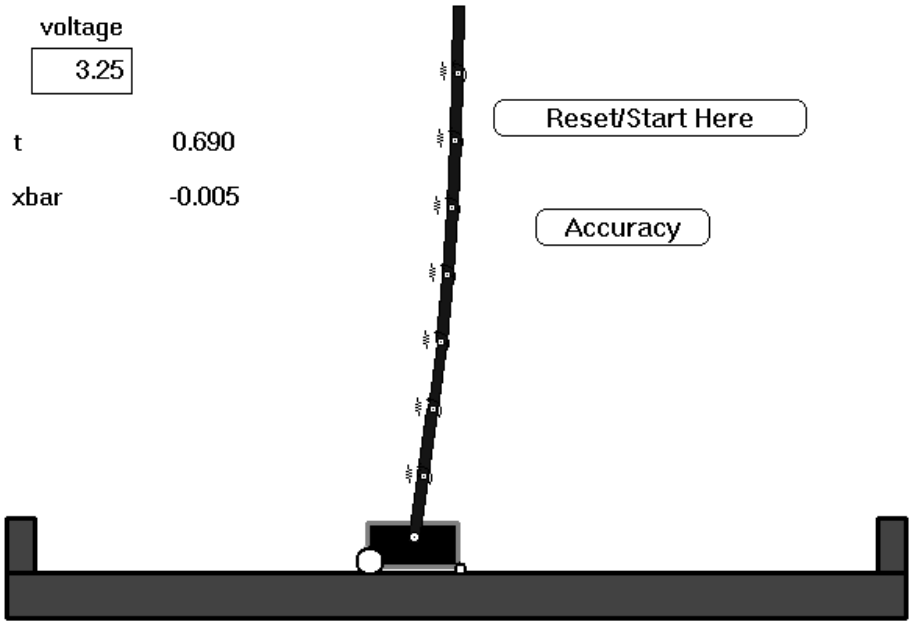

Figure 2: Working Model simulation of the inverted pendulum with flexible link

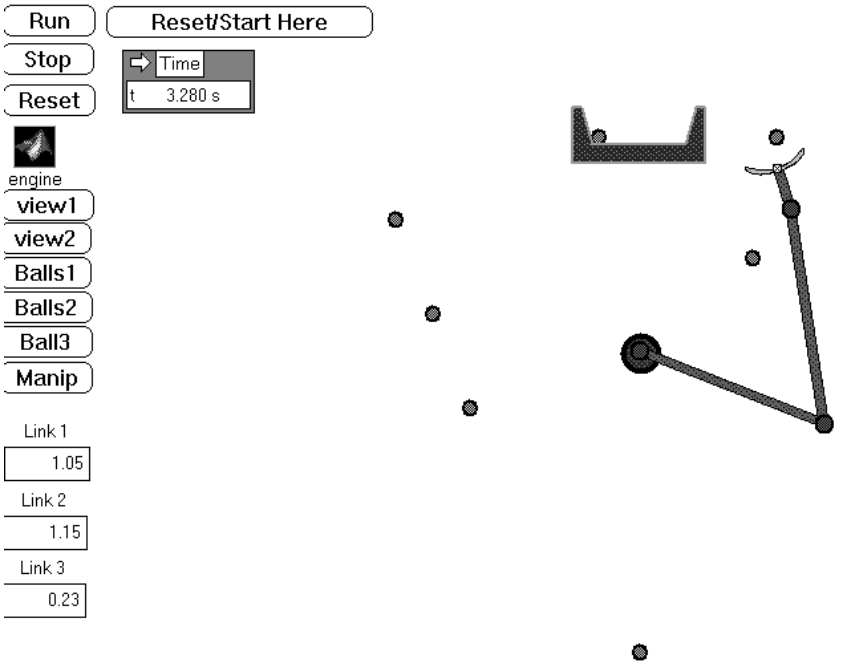

Figure 3: Simulation of planar manipulator. The balls are subject to the force of gravity only after initial contact.

Referring to Figure 3, the task consists of placing seven balls into a bin in as short a time as possible. The manipulator sits on a table which can be inclined to any desired level, allowing the control of the effective gravitational constant the system is subject to. The system inputs are the three motor torques; the system outputs are the position and velocity of each of the links *and* of the motors (not necessarily the same due to gearing, backlash and flexibility effects), and the position of the end-effector.

We discuss below how we envision using this system:

1. Learn the theoretical concepts. This includes modeling the dynamics and kinematics of the manipulator (see [3], for example), introduce the concepts of trajectory generation, and teaching the students some of the popular methods used for manipulator control (such as PID control, dynamic inversion, etc.; see [7, 1, 3]).

2. Identify the system. The students will run various tests on the manipulator in order to identify the various system parameters, such as the the link lengths, the mass moments of inertia, and the motor losses.

3. Simulate the system. The students will then simulate the system using Working Model. Various physical parameters may be specified, such as friction and collision coefficients. One may also specify general force fields, alter the gravitational constant, include air resistance in the simulation, and change many other "world" parameters. If the simulated behavior does not adequately predict the observed behavior, further identification is required.

4. Design the control system. This will be achieved using analytical techniques, in conjunction with MATLAB and MATLAB toolboxes. Part of the control design consists of trajectory generation. The students have freedom to choose joint based schemes or Cartesian based schemes [3], to pick up the balls and place them in the bin or to catapult them into the bin, etc.

5. Simulate the design. The students should first determine whether the designed control system achieves the desired objectives (all the balls are placed into the bin). They should also explore the sensitivity of their design to various simulation parameters, such as the simulation time step, the various friction coefficients and coefficients of restitution (important when making contact with the balls, or if trying to catapult the balls into the bin). The students will quickly realize that the more accurate the simulation is (as determined by the number of physical phenomena being modeled), the longer it takes to run, and will thus have to make a decision of when the simulation run time is not justified by the increased fidelity of the simulation. If the simulation does not yield acceptable results, the control system will have to be redesigned.

6. Implement the control design. If the achieved performance is not satisfactory (i.e., not all the balls are placed into the bin), the first step is to attempt to tweak the control parameters to improve the performance; the students will quickly realize the benefits of a simple and hierarchical control structure. If this does not work, the discrepancies between the simulated behavior and the observed behavior will have to be determined, requiring further identification, and perhaps acquiring new theoretical tools for modeling and control (modeling and control of systems with backlash, for example).

The diagram in Figure 3 consists of a frame of a simulation used in the Robotics and Control course. The students were only responsible for steps 4 and 5 in the design cycle. Even though the students enjoyed this and another Working Model/MATLAB project and learned a great deal from them, many students commented on one aspect of the course which they thought could have used improvement: the absence of a *real* robot manipulator. The students still felt a need (rightly so) to test out their designs on the real physical system. The reader is referred to D'Andrea's WWW site for select AVI simulations of the above project [9].

4.2 Mass Damper System and Wing Flutter

The other two experiments being developed for the course are a two-floor active mass damper system (this experiment is being purchased from Quanser Consulting), and an elastic wing that can undergo bending-

torsion flutter (this experiment is being developed at Cornell and is currently being used (without control) in a junior dynamics course at Cornell).

References

[1] H. Asada and J. J. Slotine. *Robot Analysis and Control*. Wiley, 1986.

[2] R. A. Behr. Computer simulations vs. real experiments in a portable structural mechanics laboratory. *Computer Applications in Engineering Education*, 4(1):9–18, 1996.

[3] J. J. Craig. *Introduction to Robotics*. Addison-Wesley Publishing Company, 1989.

[4] Knowledge Revolution. *Working Model*. http://www.krev.com/.

[5] L. Ljung. *System Identification*. Prentice Hall, 1987.

[6] The Mathworks Inc. *Matlab*.

[7] R. Paul. *Robot Manipulators*. MIT Press, 1981.

[8] Quanser Consulting Inc., http://www.quanser.com/.

[9] R. D'Andrea, Robotics and Control: http://www.mae.cornell.edu/raff/classes/mae417Y97/robotics.html.

[10] D. E. Rivera, K. S. Jun, V. E. Sater, and M. K. Shetty. Teaching process dynamics and control using an industrial scale real-time computing environment. *Computer Applications in Engineering Education*, 4(3):191–207, 1996.

Raffaello D'Andrea

Raffaello D'Andrea received the B.A.Sc. degree in engineering science from the University of Toronto in 1991, and the M.S. and Ph.D. degrees in electrical engineering from the California Institute of Technology in 1992 and 1996. Since then, he has been with the Department of Mechanical and Aerospace Engineering at Cornell University, where he is an Assistant Professor. He is also a member of the Applied Mathematics and Electrical Engineering fields at Cornell University. His research interests include the development of computational tools for the robust control of complex interconnected systems, and applying these techniques to mechanical and aerospace systems. His teaching interests include Systems Engineering and Robot Soccer. Dr. D'Andrea is an NSERC 1967 Fellow, and is the recipient of the American Control Council O. Hugo Schuck Best Paper award (paper co-authored with Fernando Paganini and John C. Doyle in 1994), and the IEEE Conference on Decision and Control Best Student Paper award (1996).