2015 American Control Conference
Palmer House Hilton
July 1-3, 2015. Chicago, IL, USA

Enhancing Linear System Theory Curriculum with an Inverted Pendulum Robot

Brian Howard, *Member ASME and IEEE*, and Linda Bushnell, *Senior Member, IEEE*

***Abstract*— The demands on both delivery methodology and content of control curriculum continue to evolve as more applications in our world incorporate control theory and the need for continuing education increases. Not only has the content evolved, but the application of social-behavioral science research has resulted in cooperative and active learning practices by faculty. In response to these shifts in education, an open-source inverted pendulum robot was used in a linear systems theory (LST) class taught as part of a professional master's program (PMP). The robot had to have a low cost, and enough capability to enable the students to explore and test the ideas presented in lecture and to engage in collaborative learning experiences. This paper discusses the robot, describes the key control theory experiments, and reviews the lessons learned from this experience.**

I. Introduction

Control theory and its applications continue to grow and expand into new areas of our lives. From driverless cars to electric grid management to financial modeling we see new applications almost daily. These advances require that education grows and adapts as well. The content of control theory curriculum also has to include these new topics and applications to keep up with technology.

Since the goal of control theory is to control something in the real world, advances in computing and manufacturing enhance and change how the theory is applied. This sensitivity to changes in technology means that engineers benefit greatly from continuing education [1]. In addition, many students must continue working while participating in a continuing education program so there is a strong desire to take advantage of distance learning opportunities and course structures such as MOOCS.

Social-behavioral science research continues to show that students benefit from active and collaborative learning experiences [2,4]. Working together on a controls experiment or lab allows such experiences for students. Extension of the material to a lab is natural and provides additional insight into how control theory is applied in physical systems [3].

Brian Howard is with General Electric, Minden, NV 89423 USA (phone: 425-281-3271; e-mail: brian.howard@ihmail.com).

Linda Bushnell is with the Department of Electrical Engineering at the University of Washington, lb2@uw.edu.

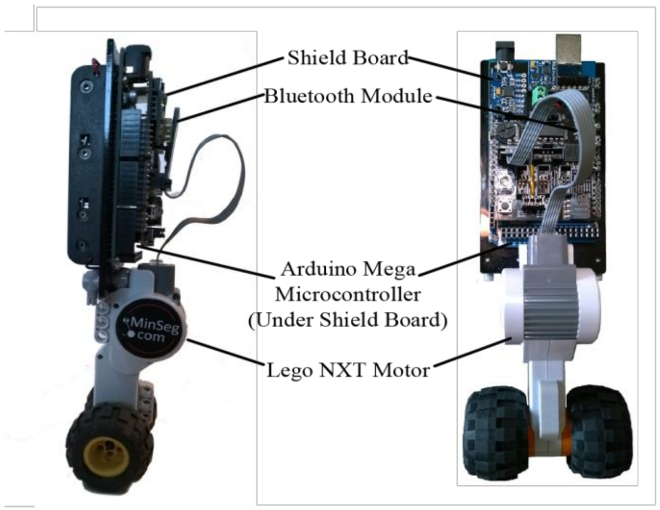


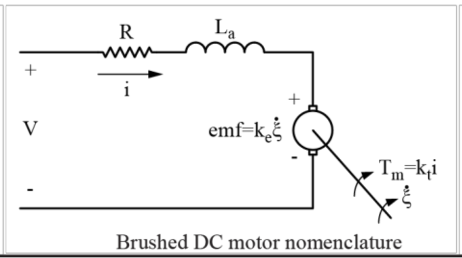
Figure 1 – MinSeg™ Robot.

In response to these shifts, there was a desire to add an experimental component to an existing linear systems class. This class was taught as part of a professional master's program (PMP), a program designed to accommodate working students. The class meets only once a week, so any control lab or experiment platform must be accessible by the students outside of class room hours. Ideally, the control lab platform would be portable and open source with a wide breadth of documentation available on the internet so that students could access resources and help at any time.

The MinSeg™ robot met this criteria. This inverted pendulum robot is based on the Arduino Mega microcontroller. The microcontroller can be programmed with the Arduino compiler or Mathworks® Simulink® program using the Rensselaer Arduino Support Package [7]. In this case the Simulink program was used because the block diagrams and models related to the materials presented in the lecture component.

The robot includes a shield board with serial port interfaces, a MEMS gyro and accelerometer, LEDs, and H-bridge drivers for the motor. The motors, axles, and wheels are all Lego® parts allowing for a number of different configurations beyond the inverted pendulum.

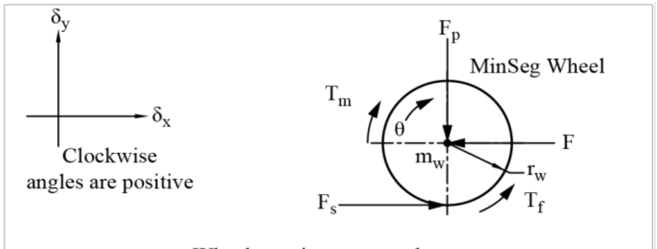
This paper is organized as follows. Section II of this paper describes the structure of the linear systems theory (LST) lecture and describes how the lab components were added and integrated with the lecture topics. Sections III through V discuss the key experiments performed on the robot. These began with "outer loop" characterization of the robot sensors and systems (motor parameterization and gyro calibration) and then moved on to application of control



Brushed DC motor nomenclature

V:	Motor voltage
T_m:	DC motor torque
R:	Motor resistance
k_t:	Motor torque constant
k_b:	Motor back-EMF constant
L_a:	Armature inductance
i:	Armature current
I_m:	Motor rotor inertia
ξ:	Motor rotation relative to the stator (pendulum frame)
b_m:	Motor rotor viscous friction

Figure 2 – Lego motor model and nomenclature



Wheel reactions nomenclature

F_p:	Force from pendulum body on wheel
T_m:	Motor torque
T_f:	Friction torque proportional to motor angular velocity
F :	Horizontal force at motor shaft
F_s:	Reaction force between wheel and surface
m_w:	Mass of wheel
r_w:	Radius of wheel
θ:	Wheel angle, relative to inertial frame
$I_{cm,w}$:	Moment of inertia at center of mass of wheel

Figure 3 – Wheel reactions and nomenclature

theory (modeling, linearization, stability analysis and pole-zero placement). Section V discusses what worked and what did not work, suggests improvements, and concludes the paper.

II. Linear System Theory Structure

The topics covered in the lecture component of the LST class began with an introduction to the concepts of linear systems and how they can be expressed with block diagrams. After showing how a physical system relates to a model, the fundamental concepts of causal and linear time invariant systems are developed to define zero-state and algebraically equivalent systems. Next, the lecture component developed the techniques to convert from state space to transfer function systems and vice versa. Using the fundamental theorem of differential equations with these concepts, the materials then showed how to find the state transition matrix. This results in general solutions for LTV, LTI, and transfer functions. With the general solution known, Lyapunov and bounded input, bounded output stability can be determined. Once a system is determined to need control the topic leads naturally to controllability and observability followed by state feedback and observers. The lecture component included homework assignments and a midterm. Both the content and structure were very similar to how the course had been taught in the past.

In addition to the lecture for the above topics, the laboratory component was added. The laboratory component had two areas of focus. The first provided the students a chance to apply the lecture topics using Matlab and Simulink. The second half covered the labs which introduced the students to various systems on the robot including the motors, encoders, gyro, and accelerometer [7].

III. Model Development

The first step in designing a controller for a physical plant is to model the system, so the first lab developed this model. The modelling approach begins with the electrical subsystem (motor) and then connects it to the mechanical subsystem [5]. The Lego motor is a brushed DC motor, shown schematically in Figure 2.

The motor model needs to relate applied voltage to delivered torque. This model can begin by relating the voltage to the current as [6]:

$$V = L_a i + Ri + k_b\dot{\xi} + 2v_{\text{brush}}$$

The current can then be related to the torque as:

$$I_m\ddot{\xi} = ik_t - T_m - b_m\dot{\xi}$$

Assuming the electric system response is much faster than the mechanical response, the inductance can be ignored. In addition, the brush voltage and viscous friction are all assumed to be negligible as well. Combining the two equations and noting that the angle of the armature rotation is the difference between the robot body angle, α, and the wheel rotation angle, θ, the torque can be written as:

$$T_m = \frac{k_tV - k_bk_t(-\dot{\alpha} + \dot{\theta})}{R} \qquad (1)$$

The motors drive the wheels, and the wheels connect the robot to the ground. Figure 3 shows the wheel nomenclature. The equations of motion for the wheel have rotational and translation components that have to be written separately [7]. Summing the torques about the wheel axis to derive rotational equations of motions gives:

$$I_{\text{cm,w}}\ddot{\theta} = -F_sr_w - T_f + T_m$$

This equation can be combined with the expression for motor torque to eliminate a variable:

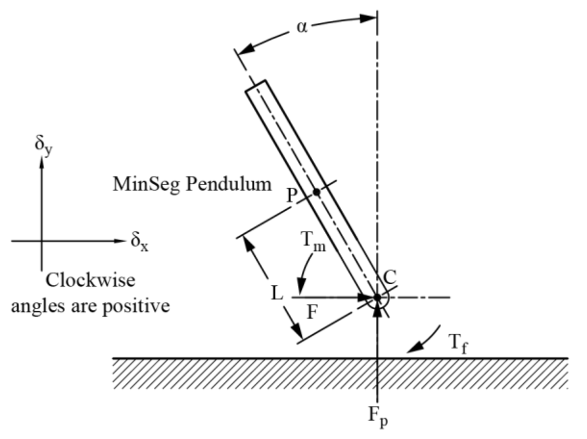
$$F_s = -\frac{T_f}{r_w} + \frac{k_t}{Rr_w}V + \frac{k_bk_t}{Rr_w}\dot{\alpha} - \frac{k_bk_t}{Rr_w}\dot{\theta} - \frac{i_{\text{cm,w}}}{r_w}\ddot{\theta} \qquad (2)$$

The translation equations of motion for the wheel are written as:

$$m_w\ddot{x} = -F + F_s \qquad (3)$$

Finally, the equations of motion for the robot body, shown in Figure 4, are derived for the horizontal direction, rotation about robot center of mass, and motion perpendicular to the pendulum:

$$m_p\ddot{x} = F - m_pL\ddot{\alpha}\cos(\alpha) \qquad (4)$$



Inverted pendulum nomenclature

P:	Pendulum reference point
α:	Pitch angle between MinSeg pendulum and vertical axis
L:	Distance from wheel center to P
m_p:	Pendulum mass
I_p:	Moment of inertia at P
x:	Travelling distance of wheel
C:	Center of wheel
F_p:	Force from pendulum body on each wheel
F:	Horizontal force at motor shaft
T_f:	Friction torque proportional to motor angular velocity
T_m:	Motor torque

Figure 4 - Robot body and nomenclature.

$$I_p\ddot{\alpha} = -L\cos(\alpha)F + L\sin(\alpha)F_p - T_m + T_f \tag{5}$$

$$m_p\ddot{x}\cos(\alpha) = F\cos(\alpha) - m_pL\ddot{\alpha} - F_p\sin(\alpha) + m_pg\sin(\alpha) \tag{6}$$

Assuming the wheels do not slip, $x = \theta r_w$, and equating the forces at the motor shaft, equations 1, 5 and 6 can be combined to eliminate the horizontal force at the motor shaft:

$$(I_p + L^2m_p)\ddot{\alpha} + L\cos(\alpha)m_p\ddot{x} = -\frac{k_t}{R}V + gL\sin(\alpha)m_p + \frac{k_bk_t}{Rr_w}\dot{x} - \frac{k_bk_t}{R}\dot{\alpha}$$

Linking the wheels and body, equations 3 and 4, together and combining with the expression for motor torque in equation 2 gives the second equation for the system:

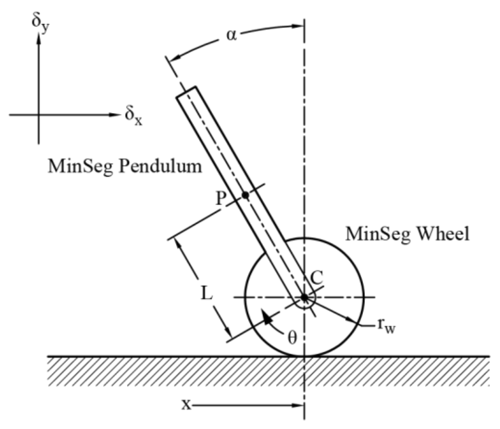
$$L\cos(\alpha)m_pr_w\ddot{\alpha} + \left(\frac{I_{\text{cm,w}}}{r_w} + m_pr_w + m_wr_w\right)\ddot{x} = \frac{k_t}{R}V - \frac{k_bk_t}{Rr_w}\dot{x} + \frac{k_bk_t}{R}\dot{\alpha}$$

This results in two equations with non-linear terms. The students then linearize the system about the desired equilibrium point (pendulum angle, α, equal to 0) using small angle assumptions ($sin(x) \approx x$ and $cos(x) \approx 1$):

$$(I_p + L^2m_p)\ddot{\alpha} + Lm_p\ddot{x} = -\frac{k_t}{R}V + gLm_p\alpha + \frac{k_bk_t}{Rr_w}\dot{x} - \frac{k_bk_t}{R}\dot{\alpha}$$

$$Lm_p\ddot{\alpha} + \left(\frac{I_{\text{cm,w}}}{r_w^2} + m_p + m_w\right)\ddot{x} = \frac{k_t}{Rr_w}V - \frac{k_bk_t}{Rr_w^2}\dot{x} + \frac{k_bk_t}{Rr_w}\dot{\alpha}$$

The next step is to cast the linearized equations into state space form. The lecture had introduced the students to representation of a system in state space form:



Inverted pendulum nomenclature

P:	Pendulum reference point
α:	Angle between MinSeg pendulum and true vertical axis
L:	Distance from wheel center to P
m_p:	Pendulum mass
I_p:	Moment of inertia at P
x:	Travelling distance of wheel
C:	Center of wheel
m_w:	Mass of wheel
r_w:	Radius of wheel
$I_{cm,w}$:	Moment of inertia at center of mass of wheel
θ:	Wheel angle, relative to inertial frame

Motor nomenclature

T_m:	DC motor torque
R:	Motor resistance
k_t:	Motor torque constant
k_b:	Motor back-EMF constant

Figure 5 - Robot model and nomenclature.

$$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu}$$

$$\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{Du}$$

Where, for the physical robot, the state variables, $\boldsymbol{x}$, is defined as $x \coloneqq [\alpha\ \dot{\alpha}\ x\ \dot{x}]^T$, the input vector u as $u \coloneqq Voltage$, the output vector y as $y = x \coloneqq [\alpha\ \dot{\alpha}\ x\ \dot{x}]^T$, and $D = 0$.

Using the linearized equations from the model, the state space representation for this model can be written as:

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{gq_1q_4}{q_1^2 + q_2q_4} & \frac{q_3(q_1 - q_4r_w)}{(q_1^2 + q_2q_4)r_w} & 0 & \frac{q_3(-q_1 + q_4r_w)}{(q_1^2 - q_2q_4)r_w^2} \\ 0 & 0 & 0 & 1 \\ \frac{gq_1^2}{q_1^2 + q_2q_4} & -\frac{q_3(q_2 + q_1r_w)}{(q_1^2 - q_2q_4)r_w} & 0 & \frac{q_3(q_2 + q_1r_w)}{(q_1^2 + q_2q_4)r_w^2} \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 \\ \frac{k_t(q_1 - q_4r_w)}{R(q_1^2 - q_2q_4)r_w} \\ 0 \\ \frac{k_t(-q_2 + q_1r_w)}{R(q_1^2 - q_2q_4)r_w} \end{bmatrix} \qquad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where:

$$q_1 = Lm_p \qquad q_2 = I_p + L^2m_p$$

$$q_3 = \frac{k_bk_t}{R} \qquad q_4 = -m_p - m_w - \frac{i_{\text{cmw}}}{r_w^2}$$

The students measured the parameters for their individual robots. In many ways this task was about as challenging as the analytic work of deriving the model. Looking at the model, it could be seen that the mass of the robot, m_p, and the pendulum moment of inertia, I_P, occur in several terms. The mass can be measured easily, but the inertia is more difficult. This drove discussion about how the accuracy of the measurements would affect the results of the stability analysis and controller design. This insight would likely not have occurred without the lab component of the course.

The measured parameters differed slightly from robot to robot. The table below shows representative values.

Table 1 - Robot parameters.

Parameter	Measurement
g	9.81 meter / (second second)
L	11.2 centimeter
m_p	381 gram
I_p	0.00616 kg meter meter
m_w	36 gram (both wheels and axle)
$I_{cm,w}$	$7.46x10^{-6}$ kg meter meter
r_w	2.1 centimeter
R	4.4 ohms
k_b	0.495 volts second / radian
k_t	0.470 newton meter / amp

With the parameters measured, these values can be substituted into the state space model to find the state space matrices for the robot:

$$\boldsymbol{A} = \begin{bmatrix} 0.00 & 1.00 & 0.00 & 0.00 \\ 62.1 & -44.6 & 0.00 & 2120 \\ 0.00 & 0.00 & 0.00 & 1.00 \\ -6.11 & 10.2 & 0.00 & -485 \end{bmatrix} \qquad \boldsymbol{B} = \begin{bmatrix} 0.00 \\ -90.0 \\ 0.00 \\ 20.6 \end{bmatrix}$$

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{D} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

IV. Stability Analysis

The lecture and homework assignments provided the students with a theoretical understanding of stability. In the lab, this is applied to the robot. The characteristic polynomial of a square matrix $\boldsymbol{A}$ can be determined as:

$$\Delta(\lambda) = \det(\lambda \boldsymbol{I} - \boldsymbol{A})$$

And in this case the characteristic polynomial can be written as:

$$\Delta(\lambda) = 0 - 17200\lambda - 62.0\lambda^2 + 530\lambda^3 + \lambda^4$$

This equation has four roots, or eigenvalues, that provide an indication of the dynamics of the system:

$$\lambda = 0 || \lambda = -530 || \lambda = -5.66 || \lambda = 5.72$$

Most of the eigenvalues cluster around the origin; however, there is one far in the left hand plane at -530. This pole is associated with the settling time of the motor and since it is so much faster than the other poles it has little effect on the system dynamics and is not shown in the root locus plots.

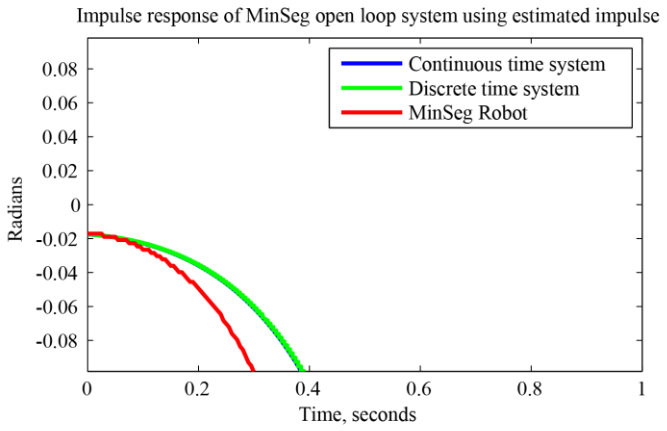


Figure 6 - Robot open loop response for the pendulum angle, α. Discrete time sampling frequency of 200 hertz.

Asymptotic stability can be found by examining the eigenvalues of the zero-input response:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t)$$

This equation is asymptotically stable if, and only if, all eigenvalues of $\boldsymbol{A}$ have negative real parts [8]. From the previous step it can be seen that some of the roots have positive real parts so the system is *not* asymptotically stable.

This equation is marginally stable if, and only if, all eigenvalues of $\boldsymbol{A}$ have zero or negative real parts [8] and those with zero real parts are simple roots of minimal polynomial of $\boldsymbol{A}$**.** From the previous step it can be seen that some of the roots have positive real parts so the system is *not* marginally stable.

As an inverted pendulum, the robot falls without control so the results of the stability analysis are much as expected. The results can be expanded further by plotting the predicted open loop response and the actual results of the robot as it falls. Figure 6 shows these results. Due to the linearization, there should be close agreement between the model and the actual robot for small angles. If the results do not agree, the students can go back and check their measured parameters, gyro calibration, and model to correct any mistakes.

With the modeling work done and the parameters confirmed to be reasonably accurate, the next step is adding feedback control.

V. Pole Placement

The lectures discuss the concept of controllability, that is, can control be added. For this problem, the controllability matrix, $\boldsymbol{\mathcal{C}}$, can be found as [8]:

$$\boldsymbol{\mathcal{C}} = [\boldsymbol{B} \quad \boldsymbol{AB} \quad \boldsymbol{A}^2\boldsymbol{B} \quad \boldsymbol{A}^3\boldsymbol{B}]$$

For this robot the controllability matrix is:

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} 0.00 & -90.0 & 47700 & -2.53 \times 10^7 \\ -90.0 & 47700 & -2.53 \times 10^7 & 1.34 \times 10^{10} \\ 0.00 & 20.6 & -10900 & 5.77 \times 10^6 \\ 20.6 & -10900 & -5.77 \times 10^6 & -3.06 \times 10^9 \end{bmatrix}$$

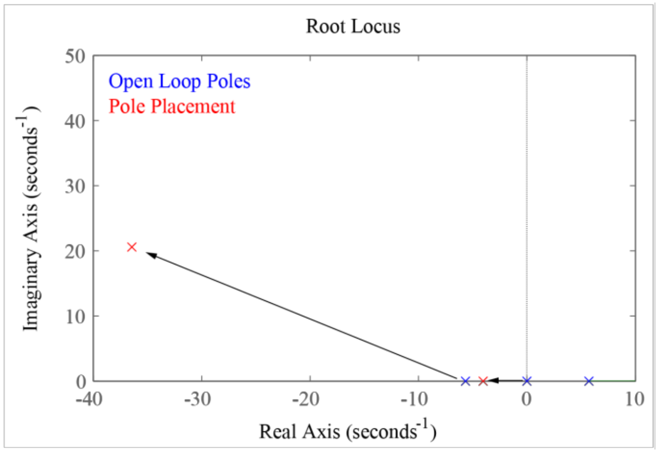


Figure 7 – Root locus plot showing the open loop poles and the location of the poles as placed for 10X response.

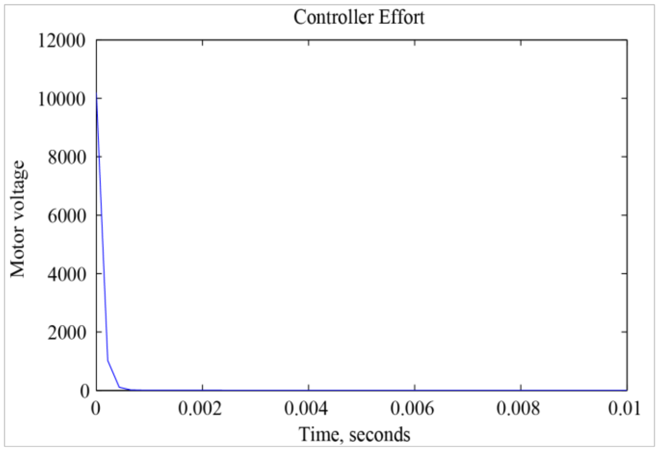


Figure 8 – Controller effort for the gains of 10X system impulse response.

The n-dimensional pair $(\boldsymbol{A}, \boldsymbol{B})$ is controllable if the controllability matrix, $\boldsymbol{C}$, has rank of n [8, p. 180]. This matrix has a rank 4. Since this is full row rank, the system is controllable.

With controllability confirmed, the final step is to add feedback and use pole placement to stabilize the robot. The lectures have explained that pole placement can be used to shift a pole from the RHP to the LHP and the students have had a chance to work on the concept in homework problems. Accomplishing this on the physical robot should result in the robot balancing, upright. The students undertake this exercise as the final project in the course.

In theory, it is a boundable problem to add feedback and shift the poles, but in practice there is a considerable amount of art involved. Unlike the theoretical model, the robot has finite power and finite response times. The pole placement must not only achieve stable operation, but do so within the operating bounds of the physical system. In order to achieve stability, the control system must respond faster than the physical system. Trial and error began by designing a controller that was ten times (10X) faster than the physical system. The poles were placed at:

$$\lambda = -10650 || \lambda = -36 + 20i || \lambda = -36 - 20i || \lambda = -4$$

Figure 7 shows the location of the open loop system poles in blue and the location of the placed poles in red. Poles are always conjugate pairs so only the positive roots are shown in the root locus plots. As was mentioned earlier, the fast pole (-10650) is not shown in the root locus plots. This resulted in the following gains [8, pp 318-320]:

$$\boldsymbol{K}_{10X} = [-31700 \quad -6760 \quad -99300 \quad -29100]$$

For brevity, units are omitted when presenting the gain vectors. The first gain has units of $volts/degree$, the second has units of $(volts - seconds)/degree$, the third has units of $volts/meter$, and the final gain has units of $(volts - seconds)/meter$.

Figure 8 shows a plot of the controller effort in response to an impulse event. While these gains work in theory, they would require that hundreds of volts be applied to the motors. Only by having a physical implementation of the controller to work with do students come to an understanding of how the theory must be adapted to the real world.

Realizing that smaller gains were needed, the desired controller response was reduced to one times (1X), resulting in the following gains:

$$\boldsymbol{K}_{1X} = [-93.1 \quad -18.1 \quad -9.93 \quad -52.6]$$

Experimenting on the actual robot showed that with these gains, it would balance for a few seconds. Trial and error continued, but it was not easy to find the right combination of gains. It was suggested that a linear quadratic regulator (LQR) might give better results with less tuning effort [10].

The textbook referenced in [13] has a good introduction to the LQR. The next few paragraphs present an overview drawn from that source, but the interested reader should refer back the textbook for a complete discussion. The LQR has its origins in work Wiener did in designing mean-square filtering. This problem was solved by minimizing the expected error criterion, V, over time:

$$V = E\{e^2(t)\}$$

If the system is linear time-invariant, has a transfer function $T(s)$, and one is interested in the steady state performance this expression can be re-written as:

$$E\{e^2(t)\} = \frac{1}{2\pi}\int_{-\infty}^{\infty} |T(j\omega)|^2 \Phi(\omega) d\omega$$

The $\Phi(\omega)$ term is the input noise spectrum. Combined with Parseval's theorem and some generalization of terms, the problem becomes the minimization of the quadratic performance measure, V:

$$V = \int_t^T (\boldsymbol{x}'\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{u}'\boldsymbol{R}\boldsymbol{u}) d\tau + \boldsymbol{x}'(T)\boldsymbol{M}\boldsymbol{x}(T)$$

This measure is subject to the following linear dynamical constraint:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$

This can be recast into an optimization equation and, by including feedback, written as the algebraic Riccati equation [14]:

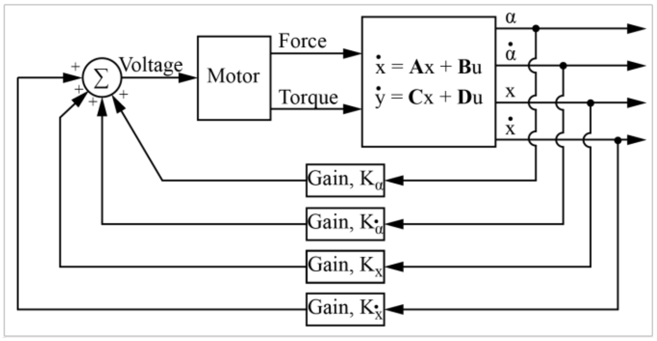


Figure 9 - LQR Controller Block Diagram.

$$\boldsymbol{A'P + PA + C'QC - PBR^{-1}B'P = 0}$$

If matrix $\boldsymbol{P}$ can be found that satisfies this equation then it has a minimum equal to zero and the closed loop gain is:

$$\boldsymbol{K = R^{-1}B'P}$$

In the Riccati equation all parameters are known, except for $\boldsymbol{Q}$ and $\boldsymbol{R}$. The term $\boldsymbol{Q}$ relates to controller accuracy and the term $\boldsymbol{R}$ relates to controller effort. The matrices can be diagonal with entries positive and large to make the associated variables small [13]. The actual values for the matrices are derived empirically. It was found that with the following weighting factors the robot balanced [11]. The matrix $\boldsymbol{Q}$ is the weight function for the states and the matrix $\boldsymbol{R}$ is the weighting function for the inputs.

$$\boldsymbol{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix} \qquad \boldsymbol{R} = [1]$$

Using Matlab to solve the equations, this controller had the following pole locations and gains [9]. With these gains the robot balanced.

$$Poles, LQR = [-1065, -3.7 + 2i, -0.4, -3.7 - 2i]$$

$$\boldsymbol{K} = [-95.0 \quad -18.2 \quad -10.0 \quad -53.4]$$

Up to this point the model has been evaluated only in the continuous time domain; however it will be implemented on a discrete time system. The MinSeg control cycle runs at 200 hertz. For most control systems the zero-order hold (ZOH) method provides accurate results to convert from continuous time to discrete time [12]. The discrete time system results are also included in the plots to confirm that the responses are similar. Figure 9 shows the block diagram for the controller.

This controller was implemented as a Simulink model, shown in Figures 10 and 11 and also available online [9]. The wheel angles are derived from the encoders on the motor. The red blocks indicate values associated with the angle, θ, and the yellow block indicate values associated with the angle rate, $\dot{\theta}$, (wheel speed).

The robot includes a 3-axis gyro and 3-axis accelerometer in a single MEMS package (MPU-6050). This sensor provided rate of pitch angle, $\dot{\alpha}$, (green blocks) and pitch angle, α, (orange blocks). Measuring this pitch angle using the MPU-6050 provided to be a challenging part of designing the controller. As the students worked through

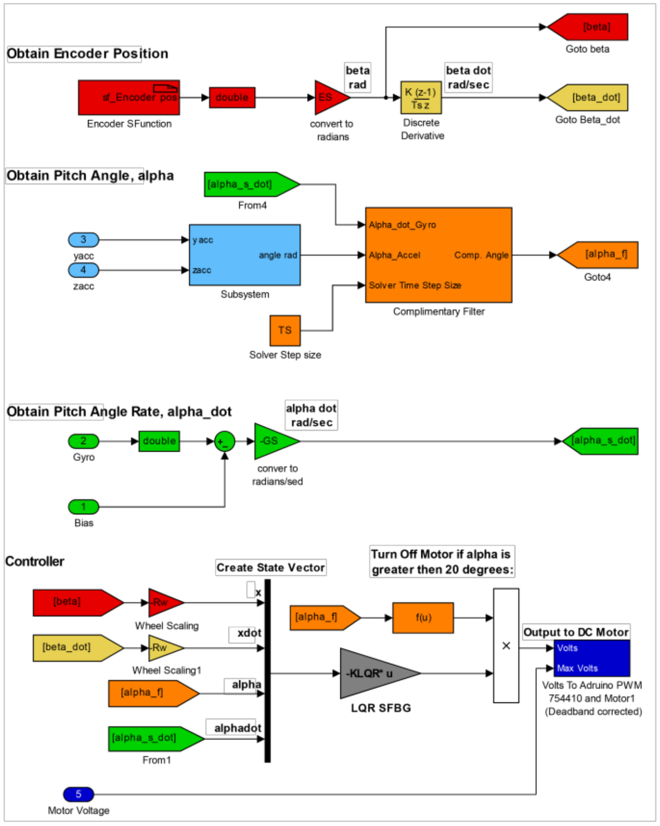


Figure 10 – Controller as implemented in the Simulink environment.

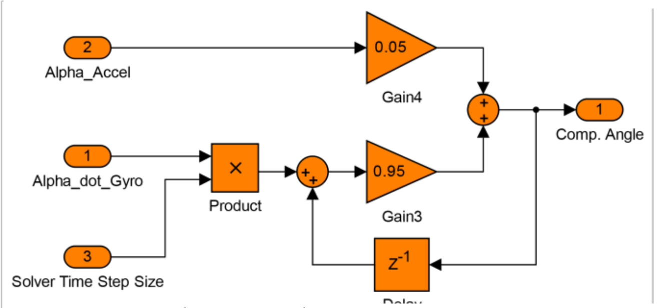


Figure 11 - Complimentary Filter.

the labs they observed that the accelerometers were relatively slow to respond and could not be used by themselves to measure pitch angle. The gyro signals have noise and a DC offset (bias) and could not be used alone to measure the angle. Ultimately, a complimentary filter was used to weigh the accelerometer and gyro signals together to determine the pitch angle. Figure 11 shows the complimentary filter design used in the MinSeg.

All four of the state variables are passed into a vector and multiplied by the gain vector (grey box in Figure 10). The output of the gain is then passed onto the motor controller (blue box in Figure 10). The controller also includes logic to shut the motor off if the robot pitch angle is greater than 20° in either direction.

Figure 12 shows data collected from the robot as the red line and compares it to the continuous-time model (blue line) and discrete-time model (green line). As expected, the controller frequency is fast enough that the discrete-time

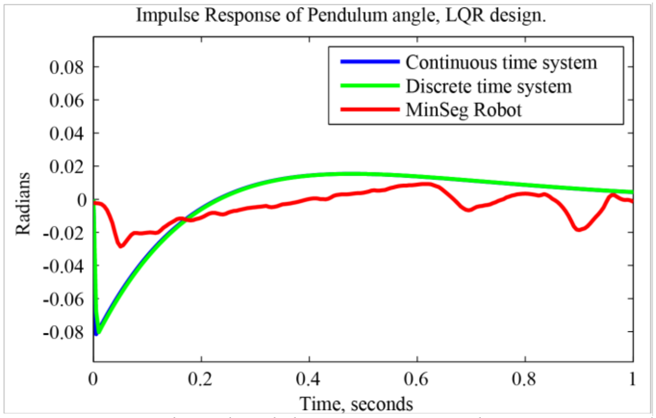


Figure 12 - Robot closed loop response with LQR optimized controller for the pendulum angle, α. Discrete time sampling frequency of 200 hertz.

response is nearly identical to the continuous-time response. The actual response, however, does not match the response of the model. One reason is that the amplitude of the impulse event on the robot is slightly different from that of the model. But that doesn't explain the steady-state difference.

The students observed, and the data shows, that the robot jittered while balancing. This movement can be seen in the difference of the MinSeg response compared to the model response. The jitter is a result of noise from the MEMS gyro that was amplified by the gains on the robot pitch angle. The ability to see noise in the physical motion of the robot as it balanced demonstrated how important both sensor selection and signal conditioning are to the performance of control systems.

In addition to the LQR optimized controller, a proportional-integral-derivative (PID) controller was also developed [9]. Unlike the LQR controller design, in which analytics were used to arrive at the gains, in this approach the gains were determined one by one. Figure 13 shows the block diagram for the PID controller. The first loop added was the proportional gain. With the gain, students could see the robot respond as it was tipped, but noted that it would not return to true vertical because of the steady state error. Next, the integral gain was added to eliminate the steady state error; however the control system response was not fast enough to prevent the robot from falling. Finally, the derivative gain was added to get the needed reaction time.

The experience of adding PID control to the robot was quite a bit different than expected based on the theoretical analysis. The biggest surprise was that the theory predicted a stable robot with just PID control on the pendulum angle, α. In the physical robot this approach did not result in a stable robot because the wheel velocity cannot increase to infinity. With just PID control on the angle, the robot would balance for a few seconds, but eventually fall over. This meant that feedback would be needed on wheel speed as well.

Adding proportional control to the wheel velocity was an interesting modeling problem because realistic gains were not enough to completely move the poles from the RHP to the LHP for the wheel control. To achieve a stable robot,

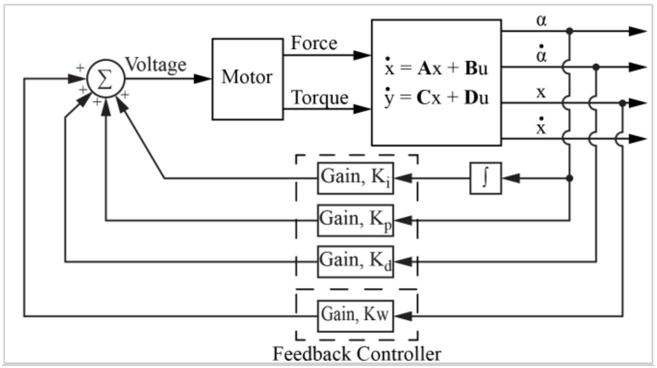


Figure 13 – PID Controller Block Diagram.

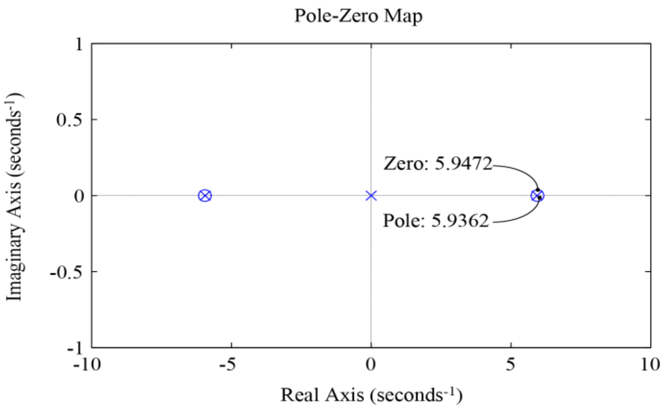


Figure 14 – Pole-zero plot showing approximate cancelation of poles in the right hand plane.

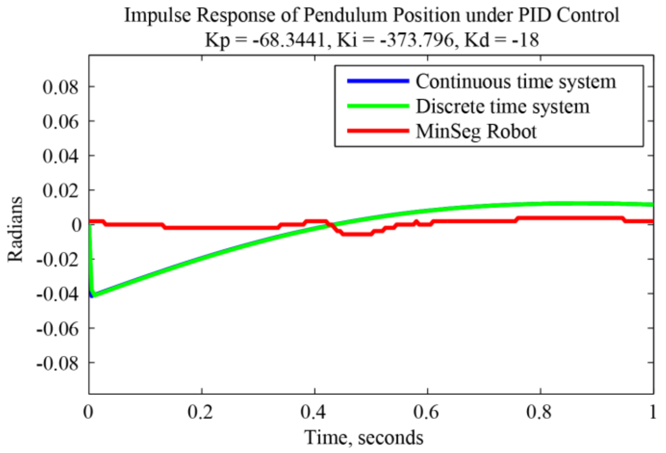


Figure 15 - Robot closed loop response with PID controller for the pendulum angle, α and proportional control on wheel speed. Discrete time sampling frequency of 200 hertz.

the gains of the PID controller were adjusted to cancel the RHP poles, shown in Figure 14.

This did result in a stable robot, although it tended to drift across the surface as the controller zeros never perfectly canceled the plant poles. Figure 15 shows the response of the robot compared to the predicted response. The transient responses are different because it is hard to provide the same impulse to the physical system that is included in the model. The steady state response has the same noise that was observed in the LQR response.

Modifying the robot by adding sensors slowed the control cycle. For example, adding an ultrasonic sensor to the robot

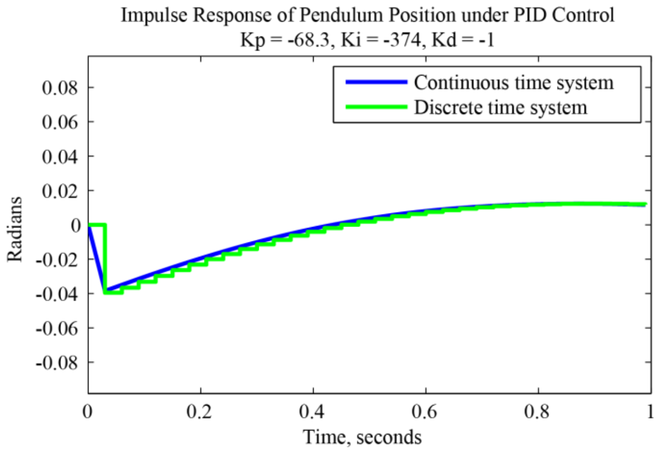


Figure 16 - Robot impulse response with PID controller for continuous and discrete time. Discrete time sampling frequency of 33 hertz.

slowed the control cycle time. The sensor had a maximum range of about 3 meters and it takes about 26 ms for the ultrasonic sound to complete the journey. To accommodate this travel time, the controller cycle time had to be dropped from 200 hertz to 33 hertz. Figure 16 shows an example of the impulse response for a continuous time system and a discrete system running at a relatively slow sampling rate of 33 hertz. Unlike Figure 15, in which discrete and continuous time systems had nearly identical responses, the two systems showed a difference with this slower sampling rate. This modelled response corresponds well to the sluggish behavior of the robot observed when an ultrasonic detector was added.

VI. Conclusion

Adding the lab to the LST regular format class proved to be beneficial, enhancing the students learning experience and giving depth to the theoretical concepts of the lecture materials. The lab section will continue to be a part of the PMP offering of the LST class.

There were a number of lessons learned from adding this lab. Learning to read data from the sensors and to use LST theory to control the actuators of a physical plant takes time and experience. The lab curriculum must include lessons that let the students gain this understanding. For example, from LST theory it was known that the robot pitch angle had to be measured, but it took considerable effort to design and implement the complimentary filter. Tuning plays an important role in any control system and allowing time in the labs and final project for the students to adjust gains and parameters of the controller gives the students a chance to explore and expand the application of the theory presented in lecture.

These findings, along with noise in the gyro, suggest some future work for the robot. For example, the usefulness of the LQR in finding gains indicates that it might be helpful to include the LQR in the LST lecture material. Research into the MEMS gyro behavior suggests that Kalman filtering would be helpful in attenuating the noise from the gyro. Reconfiguring the robot to operate with two motors, instead of one, is another area to explore. This would allow the robot to be steerable. In addition, adding a controller, such as a PID or finite-step, to the motor itself is another avenue to explore in future activities.

The inclusion of a lab that allowed students to design and build a control system on a physical plant proved valuable both in the student learning experience and in exposure to relationships and concepts that would not have been apparent from the theory alone.

Acknowledgment

The authors would like to thank Professor Joshua Hurst at RPI for his insightful and helpful discussions on the MinSeg robot, Simulink libraries, and laboratories.

References

[1] J.J. Rodriguez-Andina, L. Gomes, and S. Bogosyan, "Current Trends in Industrial Electronics Education," *Industrial Electronics, IEEE Transactions on* , vol.57, no.10, pp.3245-3252, Oct. 2010.

[2] J.E. Froyd, P.C. Wankat, and K.A. Smith, "Five Major Shifts in 100 Years of Engineering Education," *Proceedings of the IEEE* , vol.100, no. Special Centennial Issue, pp.1344-1360, May 2012.

[3] D.S. Bernstein, "Control Experiments and What I Learned From Them: A Personal Journey," *Control Systems, IEEE* , vol.18, no.2, pp.81-88, Apr 1998.

[4] M. Prince, "Does Active Learning Work? A Review of the Research," *Journal of Engineering Education*, Vol. 93, pp. 223–232, Jul. 2004.

[5] B. Bonafilia, N. Gustafsson, P. Nyman, and S. Nilsson, "Self-balancing two-wheeled robot". Web. < http://sebastiannilsson.com/wp-content/uploads/2013/05/Self-balancing-two-wheeled-robot-report.pdf>, Jan. 2015.

[6] M.F. Golnaraghi and B.C. Kuo, *Automatic Control Systems.* Hoboken, NJ: Wiley, 2010.

[7] F.P. Beer and R.E. Johnston Jr., *Vector Mechanics for Engineers*. 5th ed. New York : McGraw-Hill, 1988.

[8] C. Chen, *Linear System Theory and Design*. 4th ed. New York : Oxford UP, 2012.

[9] B.F. Howard, "MinSeg State-space LQR Controller Development." *MinSeg State-space LQR Controller Development.* Web. 25 Sept. 2014. <http://ihhome.com/~brian.howard/lqr/MinSegModel_LQR.html>, Aug. 2014.

[10] J. Hurst, "Rensselaer Mechatronics." *Rensselaer Mechatronics*. Rensselaer Polytechnic Institute, Web. <http://homepages.rpi.edu/~hurstj2/>, Sep. 2014.

[11] "Inverted Pendulum: State-Space Methods for Controller Design." Control Tutorials for MATLAB and Simulink -. Carnagie Mellon University, Web. <http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum§ion=ControlStateSpace>, Feb. 2015.

[12] R. Poley, *Control Theory Fundamentals*. Seminar Notes. Version 9.2. Dallas : Texas Instruments, 2012.

[13] P. Dorato, C.T. Abdallah, and V. Cerone. *Linear Quadratic Control. An Introduction*. Malabar, Florida : Krieger Publishing Company, 1995.

[14] J. Hespanha. *Linear Systems Theory*. Princeton, NJ: Princeton University Press, 2009.