Optimal Control 5413 Project: *Ball and Beam Optimal Control*

Charles O'Neill

7 May 2004

1 Introduction

This design project simulates and controls a beam and ball system. A ball rolls on a pivoting beam, see Figure 1. The beam is connected through DC motor through a linkage arm. The objective is to create an output feedback control system.

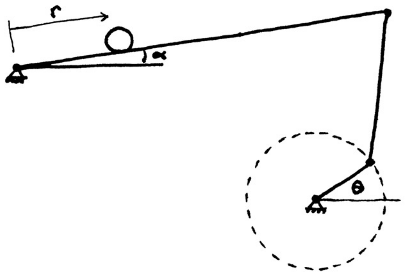

Figure 1: Ball-Beam Layout

The physical system consists of coupled linkages and a free-to-roll ball. The linkage motion is non-linearly coupled to the gear angle. The beam has a length of 1 meter. The gear arm has a radius of 0.03 meters, and the linking arm has a length of 0.2 meters. The ball is steel with a radius of 15 millimeters. The system has 2 energy storage components: beam inertia and ball inertia. System control is allowed through a voltage input into the DC motor, which provides a torque applied at the gear arm. The gear arm motion is harmonic; the gear can go past top-dead-center in the straight up and down positions.

1.1 Beam-Lever-Gear Angles

This section describes the beam motion derivation. Figure 2 was used for this derivation. The beam-lever-gear connection constraint is:

$$d \sin\theta + A(1 - \cos\beta) - L \sin\alpha = 0$$

Solving for the beam angle α yields:

$$\alpha = \arcsin\left(\frac{d}{L}\sin\theta - \frac{A}{L}(1 - \cos\beta)\right)$$

The lever arm angle β is:

$$A \sin\beta = d(1 - \cos\theta) - L(1 - \cos\alpha)$$

The general solution for α and β is complicated with

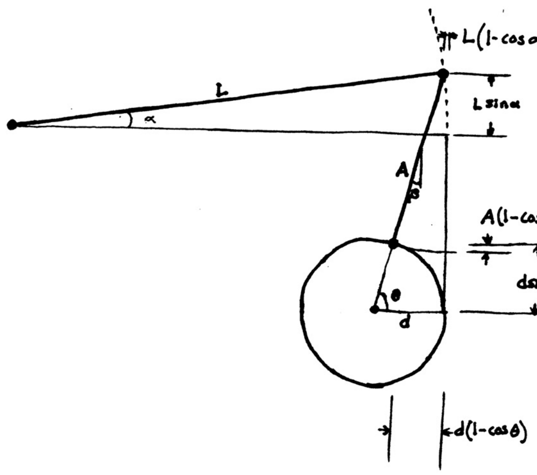

Figure 2: Beam-Lever-Gear Connections

respect to the input θ. The general, non-closed-form, solution to this 4-bar system is Freudenstein's equation[1].

The general solution to these governing equations was tested. It was noticed that the lever-arm motion is a higher order term in the overall beam angle because the gear arm length, d, is small. Also, noticing that the β term is a weak function of the beam angle α allows for a closed form $\alpha = f(\theta)$ expression.

$$\beta \approx \arcsin\left(\frac{d}{A}(1-\cos\theta)\right)$$

The beam angles and derivatives as functions of the input gear angle θ are:

$$\alpha = \arcsin\left(\frac{d}{L}\sin\theta\right) \qquad \dot{\alpha} = \frac{\frac{d}{L}\cos\theta}{\sqrt{1-\frac{d^2}{L^2}\sin^2\theta}}\,\dot{\theta}$$

$$\beta = \arcsin\left(\frac{d}{A}(1-\cos\theta)\right) \qquad \dot{\beta} = \frac{\frac{d}{A}\sin\theta}{\sqrt{1-\frac{d^2}{L^2}\left(1-\cos\theta\right)^2}}\,\dot{\theta}$$

For small d/L ratios as given in this problem, the derivative denominator terms can be ignored.

1.2 Beam-Lever-Gear Dynamics

1.2.1 Beam

The primary function of the beam is to allow a flat rolling surface for the ball. A sizing study based on deflection, inertia, and availability was performed to size the beam. 1/16 inch wall thickness 1 inch by 1/2 inch extruded aluminum tubing has sufficient mechanical properties and is readily available. Bending at mid-beam is about 1/10 *mm*. The 1 meter beam has a mass of 0.30 kg.

The beam rotates about the left end with an angle α. The beam's inertia is $J_{beam} = 1/3\ mL^2$ at the rotation point. Substituting yields an inertia of $0.10\ kg\ m^2$.

1.2.2 Lever Arm

A lever-arm length of 0.2 m was chosen based on a study of lever-arm motion versus lever-arm length. Shorter lever-arms are lighter but causes larger β angles. The 0.2 meter arm uses the same material size as the beam and has a mass of $0.060 kg$ and an inertia of $8.0 \cdot 10^{-4}\ kg\ m^2$.

1.2.3 Gear

The gear arm has a length of 0.03 m as specified in the problem statement. The "gear" is assumed to be a single lever similar to the lever-arm. The gear has a mass of $0.009 kg$ and an inertia of $2.7 \cdot 10^{-6}\ kg\ m^2$. The drive motor's inertia is considered separately.

1.2.4 Overall Beam Dynamics

The overall beam-lever-gear governing equation was derived through Lagrange's equation[2], which is based on Hamiltonian kinetic and potential energy minimization theory.[*] After some algebra the governing equation in terms of the input gear angle θ is:

$$\begin{aligned} J_\theta \ddot{\theta} &= J_T \sin(2\theta)\dot{\theta}^2 - J_L \frac{d^2}{A^2}\sin(2\theta)\dot{\theta}^2 \\ &\quad - \frac{J_T}{2}\sin(2\theta)\dot{\theta}^2 + \frac{J_L}{2}\frac{d^2}{A^2}\sin(2\theta)\dot{\theta}^2 \\ &\quad - g\cos\theta\left(\frac{Mrd}{L} + \frac{M_b d}{2} + M_L d + \frac{M_g d}{2}\right) \\ &\quad + T(t) \end{aligned}$$

The lumped inertia term J_θ is:

$$J_\theta = J_M + J_G + J_T\cos^2(\theta) + J_L\frac{d^2}{A^2}\sin^2(\theta)$$

Also, J_T is:

$$J_T = \frac{d^2}{L^2}\left(J + J_b + Mr^2 + M_L L^2\right)$$

The above governing equation contains both dynamic inertial terms and static force terms. By inspection, the zero-input *at-rest* condition due to gravity occurs at:

$$\theta = -\frac{\pi}{2} + n\,2\pi$$

1.3 Ball Dynamics

The governing differential equation for the ball dynamics was given in the project statement. However, the coordinate system for the differential equation and the corresponding figure in the handout do not match. The correction consists of moving the ball's 'r' origin to the beam's pivot end. Conceptually, the $\dot{\alpha}^2$ term should indicate that the ball accelerates *away* from the pivot point when the beam rotates. The governing differential equation is:

$$\left(\frac{J}{R^2} + M\right)\ddot{r} + Mg\sin\alpha - Mr\left(\dot{\alpha}\right)^2 = 0$$

One practical complication exists if the ball slips instead of rolls. For smooth beam surfaces, balls with large rotational inertias, and large beam angles, slipping is certain. For this project, any nonlinear ball-beam slipping behavior is neglected; rolling is linearly coupled with translation.

[*] Calculus of Variations is everywhere!

1.4 DC Motor

The motor is a DC motor. Figure 3 shows a schematic of the motor model. The overall torque is:

$$T = k_M \left(\frac{V - k_e \omega}{R} \right)$$

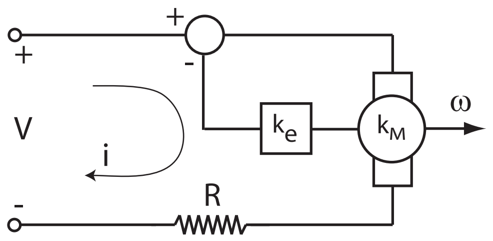

Figure 3: Motor Schematic

Adding a gearbox changes the effective torque and rotational constants by the gear ratio G. The effective torque constant is:

$$k_M = k_{M_0} \cdot G \cdot \eta$$

The effective back-emf constant is:

$$k_E = k_{E_0} \cdot G$$

1.4.1 Selection

The motor is a DC motor. Early on, intuition[†] suggested that a geared motor would be needed. θ has an effective operating range of $\pm\pi$, so small output motions and large torques are needed. An initial motor size was estimated from quasi-steady motion. For an input angular velocity of 1 rev/s, the maximum torque of 0.5 Nm occurs at $\theta = 0$. Maximum power is estimated to be below 7.5 W.

The Maxon RE-max 21 #250020 motor was selected. The continuous rated output, 6W, is probably higher than needed, but this allows for extra robustness. Also, the motor series comes with reduction gear drives and encoders. The Maxon GP 22 (53:1) reduction gear #134163 was selected. The motor, gear, and encoder specification sheets are attached in the appendix.

1.4.2 Motor Validation

The motor model was validated by disconnecting the linkages and starting the motor with a step input to its rated input voltage, 4 Volt. The spec sheet claims a 21 millisecond startup time constant and a no-load rotational velocity of 11500 rpm. Figure 4 shows the model's response: 11800 rpm and approximately a 20 millisecond startup time. The model does not include motor friction, which accounts for the slightly high no-load velocity.

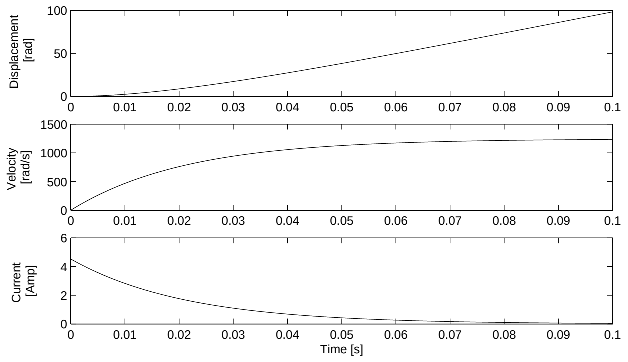

Figure 4: Motor Validation

1.4.3 Concerns

The motor model has some critical concerns. First, the gearbox losses are idealized as a torque efficiency (60%). This efficiency probably overestimates the losses at low rpms. Second, the planetary gearbox certainly has unmodeled non-linear behavior. For typical static torques caused by gravity, the gearbox will probably lock-up because of the high 51:1 gear ratio. Third, the control system can act as a generator, which implies shorted motor leads for the open loop response!

1.5 Simulink Model

The above ball-beam-motor system was implemented in Simulink. The simulink model is shown in Figure 5. There are three parts —from top to bottom— of the model: plant, controller, and observer. Also, the left side includes the inputs signals; the right side exports the states and animation.

1.6 Animation

An animation routine was created based on the supplied Matlab s-function. The animation shows ball location; beam, lever arm, and gear angles; and the ball tracking point. Figure 6 diagrams the pieces. The animation s-function (bb_animate) is attached

[†]Verified with the project's problem statement.

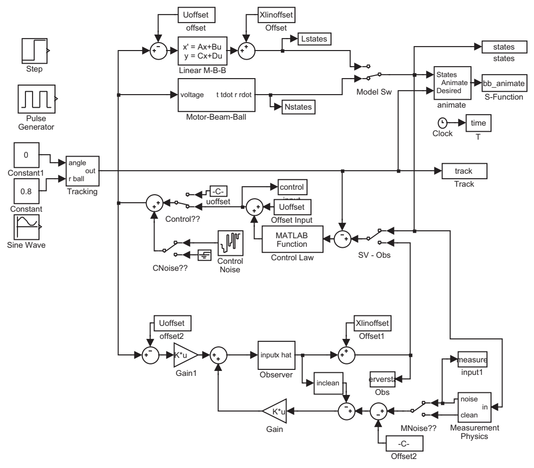

Figure 5: Ball-Beam Simulink Model

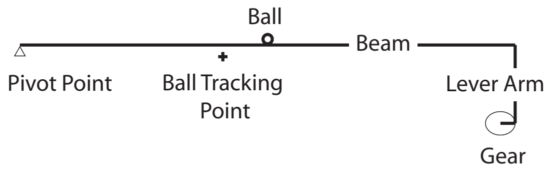

Figure 6: Animation Diagram

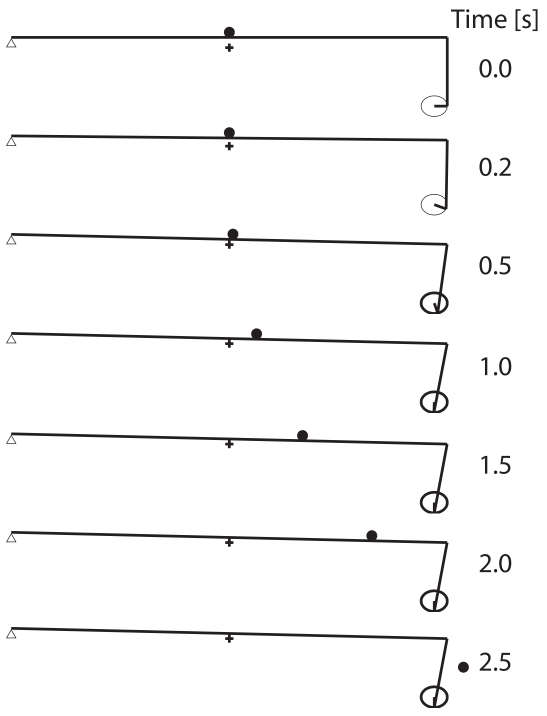

Figure 7: Open Loop Animation

in Appendix C.2.

1.7 Open Loop Response

The open loop response is simulated and plotted in Figures 7 and 8. Torques caused by beam, ball, and lever arm weights cause a non-zero beam angle of $\theta = -\pi/2$. The linkage system reaches its 90% equilibrium position in approximately 2/3 second; however, the ball continues to roll, never reaching equilibrium, and eventually falls off the beam's free —right— end. The gear equilibrium point is $\theta = -1.57$ as predicted. Significantly heavier beams will slightly cause a further *small* negative gear rotation due to the lever angle β at equilibrium. The open loop stable-beam and unstable-ball response is as expected.

Keeping the beam level requires an offset input voltage u_o. Trimming the control input voltage with the `bbtrim` Matlab script (Appendix C.6) determines u_o. The trim voltage increases as the r increases, which indicates an increasing ball's moment arm. For the mid beam ball position (r=0.5 m), the trim voltage is 0.69 Volt.

Next, the system is simulated open loop with an offset input voltage u_o of 0.688 Volt. Figure 9 shows the response. The system is unstable; because the ball's weight moment changes and also because the effective linkage arm " gear ratio" changes. From the figure, the trim voltage is slightly low. The ball accelerates away from the pivot point, which increases the required trim voltage. Eventually the ball falls off removing the ball's weight, which causes the gear to accelerate past top-dead-center ($\theta = \pi/2$).

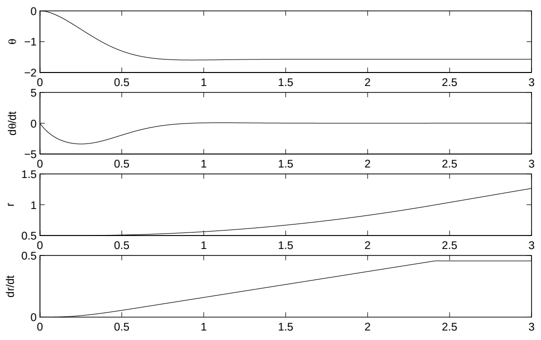

Figure 8: Open Loop States

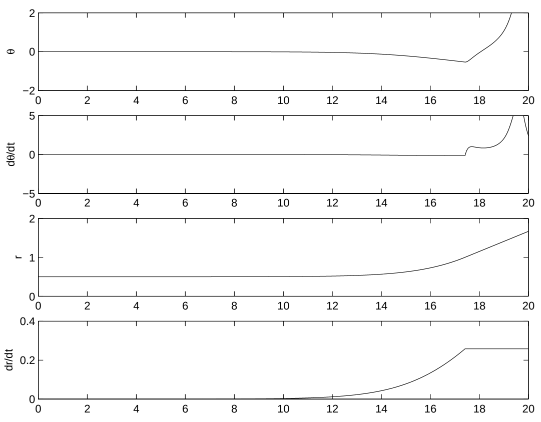

Figure 9: Open Loop with Input Offset

2 Linear Model

This section develops a linear model based on the nonlinear simulink model. The linear model is created about an operating point, so a perturbation representation must be formed first. The entire ball-beam system has 4 states: $\theta, \dot{\theta}, r, \dot{r}$. For an assumed linearization point x_{lin}, the system response is:

$$x' = x - x_{lin} \quad \text{and} \quad x = x' + x_{lin}$$

Also, the input to the linear model needs a linearization point:

$$u' = u - u_{lin} \quad \text{and} \quad u = u' + u_{lin}$$

The linear offset u_{lin} is determined from equilibrium with the `bbtrim` Matlab script (Appendix C.6).

2.1 Linearization with `linmod`

A linear state space model was created from the `sysidmodel.mdl` simulink model with Matlab's `linmod` command. `Sysidmodel.mdl` (Fig. 10) is an exact input-output representation of the ball-beam model used in the full simulink system (Fig. 5). The linmod routine outputs a system in the form:

$$\dot{x} = Ax + Bu \quad \text{and} \quad y = Cx + Du$$

The linearization requires linearization values for the states and the input. For a trimmed system ($u_o = 0.687$) with the ball at mid-beam with a level beam (r=0.5, $\theta = 0$), the state matrices are:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -9.02 & -14.82 & 0 \\ 0 & 0 & 0 & 1 \\ -0.21 & 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 52.91 \\ 0 \\ 0 \end{bmatrix}$$

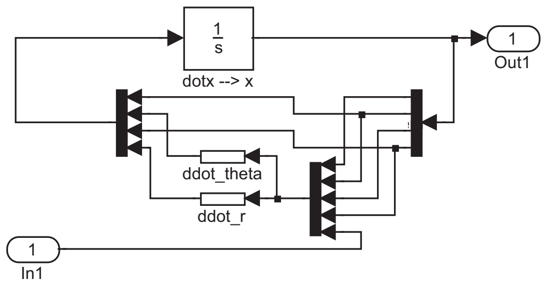

Figure 10: Linearization Model

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

The eigenvalues of the plant matrix A are:

$$\begin{bmatrix} 0.68 \\ -0.34 + 0.63i \\ -0.34 - 0.63i \\ -9.0 \end{bmatrix}$$

As seen in the open-loop non-linear simulation section, the system is unstable. As expected and seen, the unstable mode is the beam angle ($s = +0.68$). The input B matrix has a positive term in the $\dot{\theta}$ row as expected.

2.2 Simulation and Comparison

A comparison of the linear and nonlinear open loop response with the input offset is shown in Figure 11. The dark line is the linear response. The simulation runs to 17 seconds, which is when the ball falls off the beam for the nonlinear simulation. The linear response remains at zero, the nonlinear response drifts

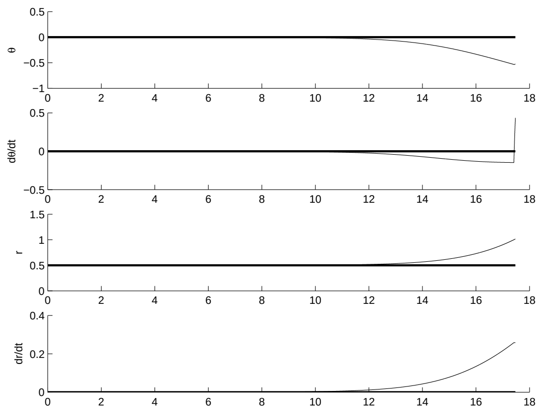

Figure 11: Nonlinear vs Linear Open loop Comparison

with the not-quite-perfect input offset. This particular linear simulation wrongly indicates that the system is stable.

A better comparison simulates the response to a step in input voltage. For this simulation, the input offset voltage was reduced by 0.02 volts; the gear angle should decrease and the ball should roll off the beam's far end. Figure 12 shows the nonlinear and linear responses. The dark line is the linear response. This simulation shows that the linear and nonlinear

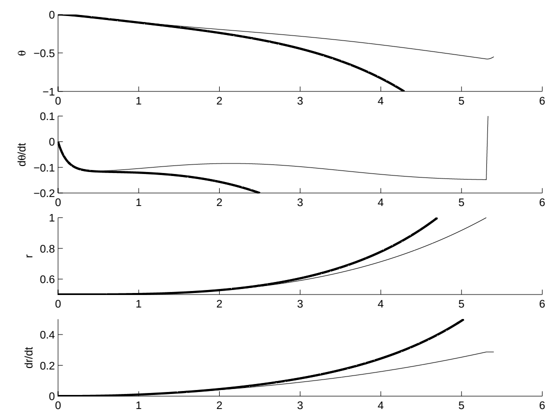

Figure 12: Nonlinear vs Linear Open loop Step Comparison

models are indistinguishable for small perturbations —up to about 1 second. When the ball moves toward the beam's end, the beam angle α increases which increases the ball's acceleration. The linear model predicts less linkage inertia. As expected, both the nonlinear and linear simulations predict the ball rolling off the beam's end.

3 State Variable Feedback

A state variable feedback controller is designed based on the steady state linear quadratic regulator[3,4]. The cost function is:

$$J = \int_0^\infty x^T R_1 x + \rho\, u^T R_2 u \, dt$$

The control signal based on the state vector x is:

$$u(t) = -R_2^{-1} \; B^T \; P \; x(t)$$

where P is the solution to the corresponding Riccati equation:

$$R_1 - PBR_2^{-1}B^TP + PA + A^TP = 0$$

This formulation requires specifying the performance index matrices R_1 and R_2. The state covariance for an input noise, w, is:

$$A\Sigma + \Sigma A^T + BS_wB^T = 0$$

3.1 Design Iteration

Here, the cost function design has a choice. By inspection, minimizing r deviations means leveling the beam. It is expected that only penalizing r will create a *good-enough* controller; however, the linear control may cause past top-dead-center gear angles, which are completely unwanted! Penalizing both ball distance, r, and gear angle, θ, should give a *better* controller.

3.1.1 Cost: Ball and Control

This design only penalizes ball location, r. The cost function is simple because the only tradeoff is between ball position and control input:

$$x^T R_1 x = r^2 \quad \text{and} \quad u^T R_2 u = \rho u^2$$

Thus, the penalty matrices are:

$$R_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R_2 = \rho$$

The expected best ratio of R_1 and R_2 is estimated from the maximum magnitudes of the r state and the u input. r is restricted to be less than 1.0 m and u must be less than 4 Volt. This ratio yields:

$$\frac{R_2}{R_1} \approx \frac{1^2}{4^2} \approx 0.06$$

Figure 13 shows a the time response for a sweep of $\rho = [1, 0.1, 0.01, 0.001]$ with a unit step in desired ball displacement from 0.5 m to 0.4 m. The linear system is linearized about 0.5 m. The resulting control scheme doesn't work very well. For large control penalties, the ball converges too slowly. When switching to larger displacements ($\Delta x > 0.1m$) or to the nonlinear model, the control scheme tends to force the gear angle past top-dead-center. The r-only cost function is rejected.

3.1.2 Cost: Ball, Gear angle, and Control

Adding the gear angle creates a cost function as:

$$x^T R_1 x = r^2 + \theta^2 \quad \text{and} \quad u^T R_2 u = \rho u^2$$

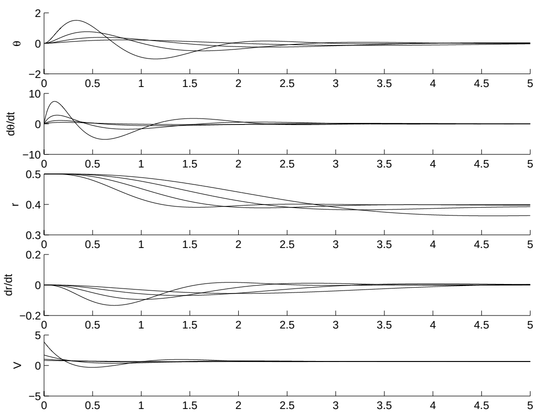

Figure 13: r: LQR Step Response

Now, ratios of θ and r must be considered. After some experimenting, an error of 1/4 degree in θ is considered equivalent to 3 mm in ball location. This gives a r/θ ratio of 2.15 Also, the maximum allowable values of r and θ are 1 m and $\pi/2$ radians. This gives a ratio of 2.47. A ratio of 2.2 was chosen. Thus, the penalty matrices are:

$$R_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2.2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R_2 = \rho$$

Figure 14 shows a the time response for a sweep of $\rho = [1, 0.1, 0.01]$ with a unit step in desired ball displacement from 0.2 m to 0.8 m. The linear system is linearized about 0.5 m. The overall performance has

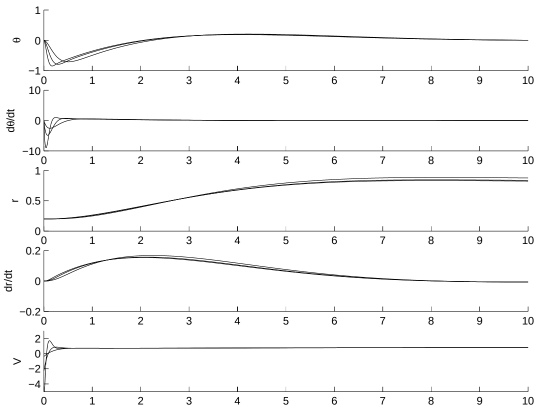

Figure 14: r θ: LQR Step Response (r/θ 2.2)

significantly improved compared to the r-only cost function. No over center behavior occurs because the cost function includes θ. However, the ball position appears to be converging slowly and always overshoots. Decreasing the ball tolerance in R_1 should help.

For an error of 2 mm in r, the r/θ cost ratio is 4. Figure 15 shows the system response. Increasing the r/θ ratio helps. The limitation now lies with the mo-

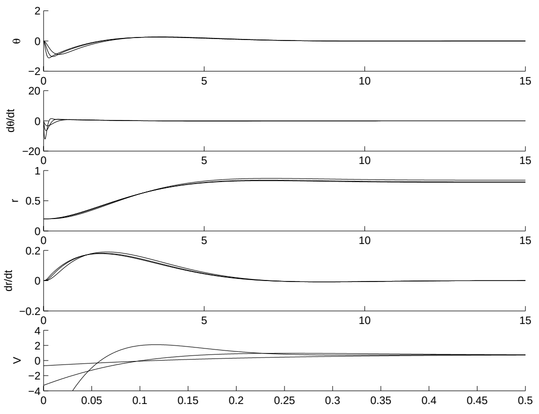

Figure 15: r θ: LQR Step Response (r/θ 4.0)

tor supply voltage. For this case, the input cost ρ is limited to 0.1 because of the motor limit of 4 Volts. Overloading the motor doesn't seem to improve the system response appreciably —especially when considering motor life degradation.

The final best control law for the r and θ cost function is with a r/θ of 4. Figure 16 shows the comparison between the states and control for the linear and nonlinear models for *two separate* step responses: 0.5 to 0.6 and 0.2 to 0.8.

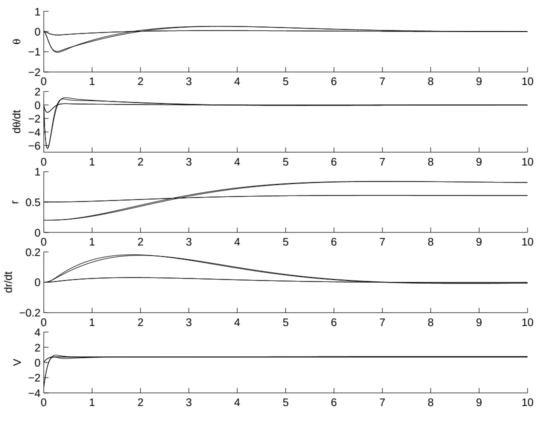

Figure 16: Linear-Nonlinear State Comparison

The state error between the linear and nonlinear model are shown in Figure 17 for their respective simulations. The linear model has a maximum ball location error of 2 cm at the highest ball velocity near 2.5 seconds.

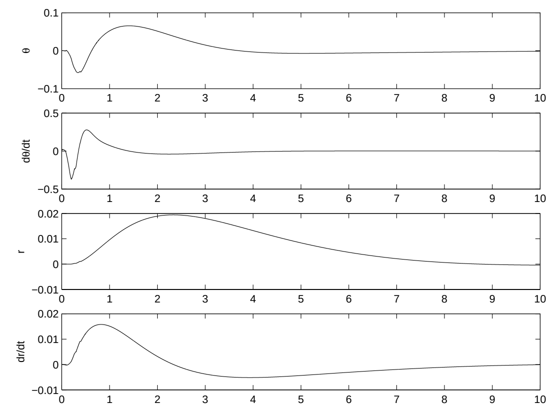

Figure 17: Linear-Nonlinear Error Comparison

3.1.3 Controller Stochastic

The steady state covariance matrix for the state vector is determined from the Lyapunov equation.

$$A\Sigma + \Sigma A^T + B\, S_w\, B^T = 0$$

where S_w is the equivalent white noise intensity of the plant input w. The motor disturbance is assumed to be white noise with a standard deviation as a percentage of the rated motor torque.

For a 1% standard deviation based on the rated motor torque of 14.6, which corresponds to a voltage of 0.04 Volts, the state covariance matrix is:

$$\Sigma = \begin{bmatrix} 6.2 \cdot 10^{-4} & 0.0000 & 1.7 \cdot 10^{-5} & 0.0000 \\ 0.0000 & 0.11 & 0.0000 & 1.3 \cdot 10^{-4} \\ 1.7 \cdot 10^{-5} & 0.0000 & 9.1 \cdot 10^{-6} & 0.0000 \\ 0.0000 & 1.3 \cdot 10^{-4} & 0.0000 & 3.6 \cdot 10^{-6} \end{bmatrix}$$

So 67% of the time, the ball location should be within $3mm$ of the desired location and the angle θ is within 1.4 degrees. Simulation inside simulink requires specifying a band limited white noise model. The intensity is σ^2. The sample time should be small enought to approximate white noise. From the linear system, the maximum eigenvalue with control (A-BK) is approximately 10 rad/s. So, an approximate system sample time is:

$$\Delta t \approx \frac{2\pi}{f_{max}}$$

A ratio of system to noise sample time should be at least 10:1.

The simulink model is simulated with a 100:1 ratio of noise to system time and a 1% motor σ. Figure 18 shows the state time history with the nonlinear system model. The experimental covariance matrix based on the nonlinear system's states is:

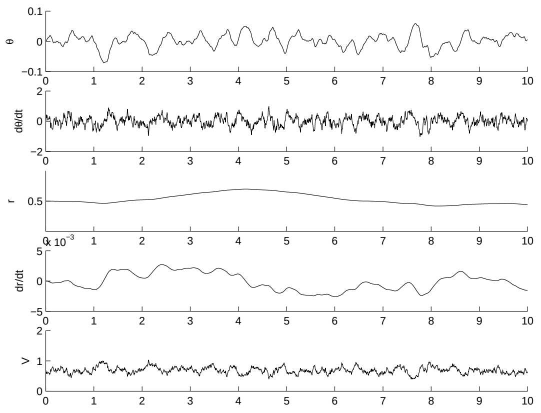

Figure 18: Controller Noise 1%

$$\Sigma = \begin{bmatrix} 5.2 \cdot 10^{-4} & -2.8 \cdot 10^{-5} & 8.3 \cdot 10^{-6} & -4.8 \cdot 10^{-7} \\ -2.8 \cdot 10^{-5} & 0.097 & 1.4 \cdot 10^{-7} & 1.0 \cdot 10^{-4} \\ 8.3 \cdot 10^{-6} & 1.4 \cdot 10^{-7} & 2.9 \cdot 10^{-6} & 1.5 \cdot 10^{-7} \\ -4.8 \cdot 10^{-7} & 1.0 \cdot 10^{-4} & 1.5 \cdot 10^{-7} & 2.0 \cdot 10^{-6} \end{bmatrix}$$

The interesting difference between the covariance matrices is that the nonlinear experimental covariance has stronger off-diagonal terms. The diagonal covariance terms are slightly lower for the nonlinear simulation.

For 10% motor torque noise in the nonlinear model, the standard deviations for r and θ are 1.6 cm and 13 degrees. The control voltage variance is 1 Volt! This much noise in the output is severely degrading performance. The control system is working at its limit as seen in Figure 19, routinely reaching a 4 Volt control input.

3.1.4 Comments

1. Switching to the time varying Riccati based gain would improve the convergence rate. Reducing the input penalty R_2 exceeds the motor voltage for the initial startup, so varying the gain near the desired tracking point would certainly be beneficial.

2. The linear model works reasonably well for this controls system even away from its linearization point. The closed loop linear system match with the nonlinear system is better than the corresponding open loop comparison.

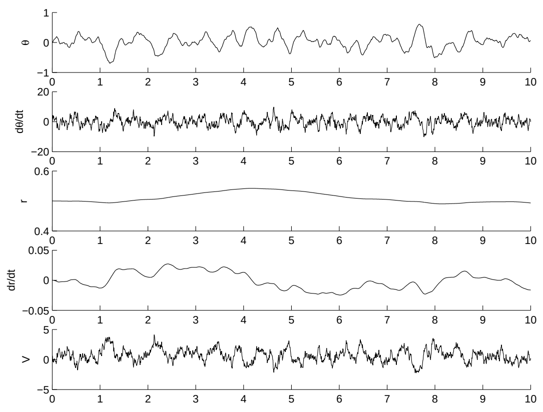

Figure 19: Controller Noise 10%

3. Constraining the control gain by adding additional state penalties helped prevent overshoot and past top-dead-center behavior.

4 Output Feedback

Output feedback creates a control signal based on measurable outputs. This requires estimating states based on outputs that may not be *clean* signals. A state observer is needed. The Kalman observer governing equation is[3,4]:

$$\dot{\hat{x}} = A\hat{x} + Bu + K_o\,(y - c\hat{x})$$

where the gain is:

$$K_o = \Sigma C^T S_v^{-1}$$

where Σ is found from the corresponding steady state Riccati equation:

$$A\Sigma + \Sigma A^T + B_w\, S_w\, B_w^T - \Sigma C^T S_v^{-1} C\Sigma = 0$$

The outputs and measurements have white noise intensities of S_w and S_v.

The corresponding stochastic linear system is:

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

$$y = Cx + Du + v$$

where w is the control noise and v is the measurement noise.

4.1 Measurement Equipment

Measurement equipment for this project needs to be chosen and characterized. This ball-beam system has 4 states: 2 second order systems. The measurement equipment needs to measure ball motion and gear motion. From the problem statement, the outputs are ball position r, and gear angle θ.

4.1.1 Gear Angle

The gear angle θ is measured with a rotary encoder on the DC motor. The selected encoder is a Maxon MR type M #201937 (Appendix D), which is a factory supported option for the Maxon motor. The encoder gives 512 counts per turn at a sampling rate up to 320 kHz.

The encoder's angle error probability function is assumed to be flat with a symmetric width of (rev/count)= $\pi/512$. With this assumption, the variance is:

$$\sigma_v^2 = \int \lambda^2 f(\lambda)\, d\lambda = \frac{\pi^2}{3c^2}$$

where c is the number of counts. So, $\sigma_v^2 = 1.255 \cdot 10^{-5}$. The covariance function is assumed to be a symmetric triangle with height σ^2 and a encoder sampling time width. The encoder is capable of 320 kHz, but for this project 100 Hz ($\Delta t = 0.01$) will be used. The white noise approximation intensity is:

$$S_v = \int_{-\infty}^{\infty} R_v(\tau) = \sigma_v^2\, \Delta t$$

So, the white noise intensity is: $S_v = 1.26 \cdot 10^{-7}$.

4.1.2 Ball Position

The ball location r is measured with an ultrasonic *time-of-flight* sensor as suggested. The selected sensor is a SICK UM 30-13113. Data sheets are given in Appendix D. Selecting a suitable ultrasonic sensor was slightly troublesome for range and response time. The selected sensor claims a 0.36mm resolution but with a 110 ms response time! It is suggested to look in more depth for a better sensor technology for this ball-beam system. Some laser sensors were especially tempting. However, the distance measuring sensor for this project will be the above SICK ultrasonic sensor.

The process of converting the position measurement system errors to a white noise is similar to the rotary encoder's. The sensor claims a resolution of 0.00036 m, but also states an accuracy of $\leq 2\%$. The noise estimation will use a 1% resolution —based on beam length— with a sampling time of 0.1 seconds. So, $\sigma^2 = \frac{1}{12}(\Delta x)^2 = 8.3 \cdot 10^{-6}$ and $S_v = 8.3 \cdot 10^{-7}$.

4.2 Validation Simulation

The observer gains were calculated with the `observer.m` script. The noise intensities are those calculated above. The observer gain is:

$$K_o = \begin{bmatrix} 100.56 & -0.00105 \\ 5055.9 & -0.07408 \\ -0.0069 & 0.4042 \\ -0.211 & 0.0816 \end{bmatrix}$$

For a step in ball location from 0.5 to 0.8, the observer estimated the following states (Figure 20. The measurements are noisy based on the resolution of the measurement system. The observer appears to be working correctly.

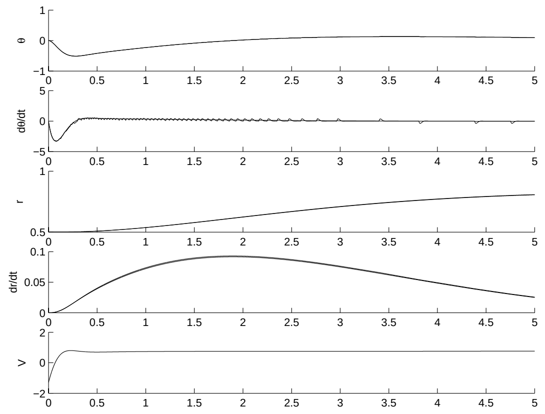

Figure 20: Observer Step

Output feedback with measurement and control noise is simulated in Figure 21. No large visible differences could be seen between the output and state control laws. The output feedback is more robust because it considers measurement and output noises.

5 Conclusions

The ball-beam system was investigated in this project. A linear system, state feedback control law, and a output feedback control law were developed. Both control laws stabilied the system. The output feedback gives similar performance, but is more robust.

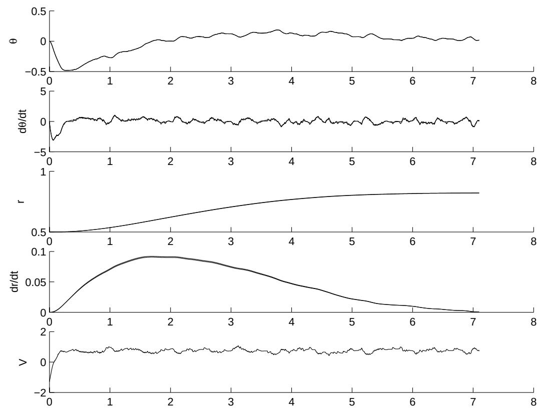

Figure 21: Observer Step Comparison

References

[1] Erdman, A. and Sandor, G., *Mechanism Design: Analysis and Synthesis*, Prentice-Hall, Upper Saddle River, NJ, 3rd ed., 1997.

[2] Moretti, P. M., *Modern Vibrations Primer*, CRC Press, Boca Raton, FL, 1st ed., 2000.

[3] Burl, J., *Linear Optimal Control*, Addison Wesley Longman, Menlo Park, CA, 1st ed., 1999.

[4] Hagan, M., "5413 Optimal Control Notes," Stillwater, OK, Spring 2004.

APPENDICES

A Nomenclature

M	Ball Mass 0.11 kg
R	Ball Radius 0.015 m
J	Ball Rotational Inertia $9.99 \cdot 10^{-6}$ $kg \cdot m^2$
r	Ball Position from Beam Pivot m
L	Beam Length 1 m
M_b	Beam Mass kg
J_b	Beam Inertia $kg \cdot m^2$
α	Beam Angle rad
A	Lever Arm Length m
M_L	Lever Arm Mass kg
J_L	Lever-Arm Inertia $kg \cdot m^2$
β	Lever-Arm Angle rad
d	Gear Length 0.03 m
M_G	Gear Mass kg
J_G	Gear Arm Inertia $kg \cdot m^2$
θ	Servo Gear Angle rad
G	Gearbox Ratio
T	Motor Torque $N \cdot m$
J_M	Motor Inertia $kg \cdot m^2$
η	Motor Efficiency
g	Gravitational Acceleration 9.81 $m \cdot s^{-1}$

B Instructions for Simulating the Ball-Beam System

1. Setup the simulation parameters: execute the `bbsetup.m` Matlab script. The bbsetup file contains the linkage, motor, and sensor specifications.

2. Load the Simulink model: load `bb.mdl`

3. Trim the system with `bbtrim.m`. The control voltage offset is found by trimming the system at the particular initial conditions specified in the bbsetup file.

4. Linearize the system with `linearize.m`. This routine requires the `sysidmodel.mdl` simulink model, which is an exact replication of the nonlinear model in `bb.mdl`. The linearization is performed about the set point specified in the bbsetup file. Linear state space matrices are stored in Alinear, Blinear,... Warning: The linear matrices in the bbsetup file are not automatically changed when `linearize.m` is run.

5. Set Simulink model switches: All simulations are made with the same Simulink model. Switches turn on or off the tracking signal, controller, noise, and observer.

6. Start the Simulink model. The animation automatically brings up a new window.

7. Plot the state variables with `plotterc.m`.

8. Synthesize a LQR controller with `bblqr`. The linear system is specified in the bbsetup file. The control penalty multiplier ρ must first be specified at the command line. Control gains are automatically loaded when the simulink model is run again.

9. Sythesize an observer with `observer.m`. Plot the observer states with `plottero.m`.

C Matlab Programs

This project used the Matlab codes given below.

C.1 Setup file bbsetup.m

```
%
%    Setup Ball and Beam
%
clear all
global g
global R M  J
global L Mb Jb
global A ML JL
global d MG JG
global km ke RM JM GR eta
global Kcontrol

% Initial Condition
x0=[0.0 0.0 0.5 0.0];% theta thetadot r rdot
Xlinoffset=[0.0 0.0 0.5 0.0];
Xlin0=x0-Xlinoffset;

% Linear System
Alin=[...
          0           1           0           0;...
          0     -9.0169     -14.818           0;...
          0           0           0           1;...
   -0.20967           0           0           0;...
 ];
Blin=[      0;...
       52.913;...
            0;...
            0;...
 ];
Clin=[      1           0           0           0;...
            0           1           0           0;...
            0           0           1           0;...
            0           0           0           1;...
   ];
Dlin=[0;0;0;0];

% Control System
Uoffset=0.68772;
Kcontrol=[3.5229       0.23233       -6.6108       -14.586];
Sw=(0.01*4)^2; % Motor noise power
Sw_time=0.006;

% Observer
QuanAngle=(2.0*pi)/512.0;
QuanDist=0.36/1000.0;
```

```
Kobserver =[ 100.56      -0.0010507;
         5055.9       -0.074048;
    -0.0069212           0.4042;
         -0.211        0.081691;
    ];


%%%%%%%%%%%%%%%%%%%%%%%%
% Mechanical Properties

% Gravity
g=9.81;

% Ball Properties
R=0.015;
M=0.11;
J=9.99E-6;

% Beam Properties
L=1.0;
Mb=0.301;
Jb=0.1;

% Lever Arm Properties
A=0.2;
ML=0.060134;
JL=8.01792E-4;

% Gear Properties
d=0.03;
MG=9.020156E-3;
JG=2.70605E-6+0.002;

% Motor Properties Maxon RE-max 21
GR=53.0;% Gear Ratio
JMGEARBOX=0.4*(1000.0*100.0*100.0)^-1;%  kg m^2
JMMOTOR=2.48*(1000.0*100.0*100.0)^-1;%  kg m^2
JM=JMGEARBOX+JMMOTOR*GR;% Motor+Gearbox Inertia
km=3.21*(1.0/1000.0)*GR; % Nm/amp <--(7.062E-3) oz-in/amp
ke=2970.0^-1.0*GR*(60.0/2.0/pi); % V s/rad <-- (2*pi/60*10E-3) mV/rpm
RM=0.883; %Ohms
eta=0.60; %Motor-Gearbox Torque Efficiency
```

C.2 Animation bb_animate.m

```
% Animation Function for Ball-Beam Project in 5413 Opt. Control
function [sys, x0, str, ts]=animdemo(t, x, u, flag,L,A,R,d,x0)
%%%
% Rewritten by Charles O'Neill
% This file is based on:
```

```
%==
% ANIMDEMO S-function animation demo for ECEN/MAE 3723
% Written by: Jason Horn
% 11/24/03
% Revised 3/31/04
%===

% Declare global variables.
global xbeam ybeam xball yball vball xgear ygear xlever ylever xballwish

% Global variables for handles of drawings
global beam ball gear lever spindle ballwish

% Global variable for the handle of the animation figure.
global AnimDemoFigure

% Set variables str and ts according to S-function specifications
str=[];
% ts=[time between samples, start time]  Decreasing time between
% samples will slow the simulation down if it runs too fast.
ts=[0.1 0];%[0.05 0];

% Ball Velocity due to gravity --- off the beam!
vball=0;

% Check the value of flag.
if flag==2

    % Update the actual angles from the states
    theta=u(1);
    thetadot=u(2);
    thetadotdot=0;
    [alpha,alphadot,beta,betadot]=fourbar_motion(theta,thetadot);
    r=u(3);
    xballwish=u(5);

    % Make sure correct figure is selected and bring it to the front.
    if any(get(0,'Children')==AnimDemoFigure)
        set(0, 'CurrentFigure', AnimDemoFigure);
        if any(get(gca, 'Children')==beam)

            % Calculate new coordinates for each figure.

            % Beam
            xbeam=[0 L]*cos(alpha);
            ybeam=[0 L]*sin(alpha);

            % Ball
            if( (r<0) || (r>1))
                vball=vball-9.81*ts(1);
                xball=r;
                yball=max(-1,yball+vball*ts(1));
            else
                xball=r;
                yball=r*sin(alpha)+R/cos(alpha);
            end

            % Gear
            xgear=[1-d, 1-d+d*cos(theta)];
            ygear=[-0.2, -0.2+d*sin(theta)];

            % Lever Arm
            xlever=[xgear(2),xbeam(2)];
            ylever=[ygear(2),ybeam(2)];

            % Set new coordinates for each figure and redraw.
            % Note: Damper in Stretch was drawn as two pieces,
            %   so each piece must be set. Draw figures as one
            %   piece whenever possible to simplify.
            set(beam, 'XData', xbeam, 'YData', ybeam);
            set(ball, 'XData', xball, 'YData', yball);
            set(ballwish, 'XData', xballwish, 'YData', -0.03);
            set(gear, 'XData', xgear, 'YData', ygear);
            set(lever, 'XData', xlever, 'YData', ylever);
           drawnow
        end
    end

    % Specify sys according to s-function specifications.
    sys=[];

elseif flag==0

    % Initialization - setup figure, create and draw base shapes.
    [alpha,alphadot,beta,betadot]=fourbar_motion(x0(1),x0(2));

    % Check for existing figure.
    [fig, flag]=figflag('Animation_Demo_Figure', 0);
    % If figure exists, clear it.
    if flag
        AnimDemoFigure=fig;
        cla reset;
    % If not, create new figure.
    else
        AnimDemoFigure=figure;
    end

    % Set title of figure.
    set(AnimDemoFigure, ...
        'Name', 'Ball-Beam_Animation',...
        'NumberTitle', 'off')
```

```
     % Set properties and limits of the axes.
     set(gca, ...
         'Visible', 'off',...
         'DrawMode','fast',...
         'XLim', [-0.2 1.2],...
         'YLim', [-1 0.4]);

    % Beam
    xbeam=[0 1];
    ybeam=[0 0];

    % Ball
    xball= x0(3);
    yball= x0(3)*sin(x0(1))+R/cos(x0(1));

    % Ball Wish
    xballwish=x0(3);

    % Gear
    xgear=[1-d 1];
    ygear=[-0.2 -0.2];
    angle=0:pi/20:2*pi;
    xspindle=d*cos(angle)+xgear(1);
    yspindle=d*sin(angle)+ygear(1);

    % Lever Arm
    xlever=[xgear(2),xbeam(2)];
    ylever=[ygear(2),ybeam(2)];

    % Draw base shapes at initial positions.
    hold on;
    ball=plot(xball, yball,'ko');
    ballwish=plot(xballwish, -0.03,'+');
    beam=plot(xbeam, ybeam, 'k');
    gear=plot(xgear, ygear, 'k');
    lever=plot(xlever, ylever, 'k');
    spindle=plot(xspindle,yspindle, 'k');

    % Draw Misc Visual Support Shapes
    plot(0,-0.02,'k^'); % Beam Rotation point

    % Define sys and x0 according to S-function specification.
    % sys=[0 0 0 (# of inputs) 0 0 1]
    sys=[0 0 0 5 0 0 1];
    x0=[];
end
```

C.3 Beam Motion fourbar_motion.m

```
%
% Motion Terms for a 4 bar
```

```
%
function [alpha,alphadot,beta,betadot]=fourbar_motion(theta,thetadot)

global A d L

% Displacements
alpha=asin(d/L*sin(theta));
beta=asin(d/A*(1-cos(theta)));

% Velocities
alphadot=d/L*cos(theta)*thetadot;
betadot=d/A*sin(theta)*thetadot;

return
```

C.4 Ball Governing Equation ddotr.m

```
%
%
%
function output=ddotr(theta, thetadot, r, rdot, motor)
global g Jb L J M R d



[alpha,alphadot,beta,betadot]=fourbar_motion(theta,thetadot);
if(r<1 && r>0)
    output=(M*r*alphadot^2-M*g*sin(alpha))/(J/R^2 + M);
else
    output=0;
end

return
```

C.5 Beam Governing Equation ddottheta.m

```
%
%
%
function thetadotdot=ddottheta(theta, thetadot, r, rdot, voltage)

global g
global R M  J
global L Mb Jb
global A ML JL
global d MG JG
global km ke RM JM eta


if(r>1 || r<0)
```

```
    r=0;
end

[alpha,alphadot,beta,betadot]=fourbar_motion(theta,thetadot);

% Dynamic Terms
JT=(J+Jb+M*r^2+ML*L^2)*d^2/L^2;
JTHETA=JM+JG+JT*cos(theta)^2+JL*d^2/A^2*sin(theta)^2;
thetadotdot=JT*sin(2*theta)*thetadot^2/JTHETA-JL*d^2/A^2*sin(2*theta)*thetadot^2/JTHETA-si

% Static 'Gravity' Terms
thetadotdot=thetadotdot-cos(theta)*(M*r*d/L+Mb*d/2+ML*d+MG*d/2)*g/JTHETA;

% Motor Terms
%if(voltage~=0)
    torque=km*(voltage-ke*thetadot)/RM*eta;
    thetadotdot=thetadotdot+torque/JTHETA;
    %end

return
```

C.6 Trim trim.m

```
%
% Trim the system
% (find control voltage for gravity offset)
%
%

% Trim
x = [0;0;0.5;0];
u = 0;
y = [0;0;0.5;0];
ix = [];        % Don't fix any of the states
iu = [];        % Don't fix the input
iy = [1;2;3;4];    % Fix both output 1 and output 2
[x,u,y,dx] = trim('sysidmodel',x,u,y,ix,iu,iy)
```

C.7 Linearize linearize.m

```
%
%
% Linearize
%
bbsetup

u=Uoffset

x0p=[0;0;0.5;0];
```

```
[Alinear,Blinear,Clinear,Dlinear]=linmod('sysidmodel',x0p,u)
```

C.8 State/Control Plotter plotterc.m

```
%
%
%
rows=5
cols=1

subplot(rows,cols,1)
hold on;
plot(time,states(:,1))
ylabel('\theta');

subplot(rows,cols,2)
hold on;
plot(time,states(:,2))
ylabel('d\theta/dt');

subplot(rows,cols,3)
hold on;
plot(time,states(:,3))
ylabel('r');

subplot(rows,cols,4)
hold on;
plot(time,states(:,4))
ylabel('dr/dt');

subplot(rows,cols,5)
hold on;
plot(time,control(:,1))
ylabel('V');
```

C.9 Control Law Sythesis bblqr.m

```
%
% LQR routine
%

% States
R1=[1.0 0 0 0;
    0 0 0 0;
    0 0 20 0;
    0 0 0 0;
]

% Control
R2=rho;
```

```
%LQR
[Kcontrol,SS,EE]=lqr(Alin,Blin,R1,R2)
```

C.10 Control Law control_law.m

```
%
%
%
function voltage=control_law(theta, thetadot, r, rdot, voltage)

global g
global R M  J
global L Mb Jb
global A ML JL
global d MG JG
global km ke RM JM
global Kcontrol

voltage=-Kcontrol*[theta thetadot r rdot]';

return
```

C.11 Control Law Sythesis observer.m

```
%
%
% bblyap
%

% Control noise
Sw=Sw;

% Measurement
Ceffective=[1 0 0 0;
    0 0 1 0;
];

RR=[1.26E-7  0; 0 8.3E-7;
];

% Control
QQ=Blin*Sw*Blin';

%LQR
[Kobserver,SSS,EEE]=lqr(Alin',Ceffective',QQ,RR);
Kobserver=Kobserver'
```

D Equipment Data Sheets