EE C222 Project: Ball and Beam Trajectory Tracking

Jinsheng Li, Miranda Xie, Leo Huang, and Dylan Lee
University of California, Berkeley
(Dated: April 5, 2025)

I. INTRODUCTION

We developed a controller for a Ball and Beam dynamical system. This setup consists of a track along which a metal ball can roll freely. One side of the beam is attached to a lever arm coupled to the load gear of a rotary servo unit. By controlling the servo's position, the beam angle is adjusted to give the ball a desired acceleration.

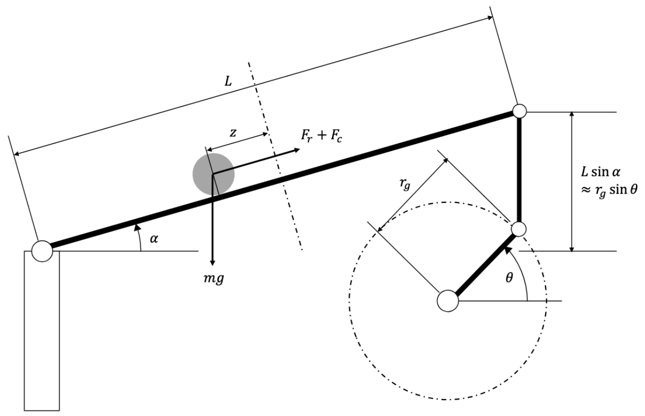

FIG. 1: The Ball and Beam Setup

The controller is to be evaluated based on three criteria: how accurately it tracks a given reference trajectory, its energy consumption, and any violations of physical or safety constraints. Each category incurs a penalty, and the overarching goal is to minimize the total score.

Our full code is uploaded to this Github repository: https://github.com/ljs-233233/EE222-Ball-and-Beam.

II. SYSTEM DYNAMICS AND SCORING SCHEME

The dynamics of the Ball and Beam system is given by the nonlinear equation $\dot{\mathbf{x}} = F(\mathbf{x}, u)$ where $\mathbf{x} \in \mathbb{R}^4$ and $u \in \mathbb{R}$. A list of parameters of the Ball and Beam system is given in I.

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{5g}{7}\frac{r_g}{L}\sin x_3 - \frac{5}{7}\left(\frac{L}{2} - x_1\right)\left(\frac{r_g}{L}\right)^2 x_4^2 \cos^2 x_3 \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -\frac{x_4}{\tau} + \frac{K}{\tau}u
\end{aligned} \tag{1}$$

We can Euler-discretize the system as follows:

$$\begin{aligned}
x_1[k+1] &= x_1[k] + x_2[k]dt \\
x_2[k+1] &= x_2[k] + \Bigg[\frac{5g}{7}\frac{r_g}{L}\sin x_3[k] \\
&\quad - \frac{5}{7}\left(\frac{L}{2} - x_1[k]\right)\left(\frac{r_g}{L}\right)^2 x_4[k]^2 \cos^2 x_3[k]\Bigg]dt \\
x_3[k+1] &= x_3[k] + x_4[k]dt \\
x_4[k+1] &= x_4[k] - \left(\frac{x_4[k]}{\tau} + \frac{K}{\tau}u[k]\right)dt
\end{aligned} \tag{2}$$

Where dt is the time step. We define this system as $\mathbf{x}[k+1] = \tilde{F}(\mathbf{x}[k], u)$.

TABLE I: Ball and Beam System Parameters

Parameter	Variable(s)	Value
Ball Position	x_1, z	
Ball Velocity	x_2, $\dot{z}$	
Beam Angle	x_3, θ	
Beam Angular Velocity	x_4, $\dot{\theta}$	
Motor Voltage	u, V	
Gravitational Acceleration	g	9.81 m/s^2
Servo Arm Length	r_g	0.0254 m
Beam Length	L	0.4255 m
Motor Constant	K	1.5 rad/sV
Delay Constant	τ	0.025 s

We aim to design a controller that computes the servo motor voltage V to drive the ball along a desired time-varying trajectory over a period of $T = 90$ s. The primary objectives are to minimize the tracking error and the motor's energy consumption, while ensuring safe operation. In particular, the ball must remain within the physical limits of the beam. The metrics used to quantify system performance are the Tracking Error:

$$J_{\text{tracking}} = \frac{1}{T}\int_0^T (x_1(t) - z_{\text{ref}}(t))^2 \, dt \tag{3}$$

Energy Consumption:

$$J_{\text{energy}} = \frac{1}{T}\int_0^T u^2(t) \, dt \tag{4}$$

And Safety-Constraint Violations:

$$J_{\text{safety}} = \begin{cases} 1, & z(t) \text{ is unsafe} \\ 0, & \text{else} \end{cases} \tag{5}$$

Where $z(t)$ is unsafe when:

$$\begin{cases} |z(t)| \geq 0.19 \text{ m}, & \text{collision imminent} \\ |\theta(t)| \geq 60^\circ, & \text{angle outside operation range} \end{cases} \tag{6}$$

The goal is to minimize the total score achieved by the controller:

$$J_{\text{total}} = w_t J_{\text{tracking}} + w_e J_{\text{energy}} + w_s J_{\text{safety}} \tag{7}$$

Where w_t, w_e, and w_s are the weight coefficients assigned to each cost.

III. OBSERVER DESIGN

The observer we designed is an Extended Kalman Filter (EKF), selected for two primary reasons. First, we anticipate the sensors having at least some degree of noise, and the EKF by construction aims to filter out measurement noise. Second, we bear in mind that the model given in the project document will almost certainly be inaccurate, as the true plant will have friction and most likely some delay; the EKF also permits some degree of modeling uncertainty, weighing the assumed model and measurements according to its tuning.

We begin by simply assuming additive process noise in the model:

$$\begin{aligned} \tilde{x} &= x + v \\ \Rightarrow \tilde{x}[k+1] &= F(\tilde{x}[k], u[k]) + v[k] \end{aligned} \tag{8}$$

where $v = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 \end{bmatrix}^\top$ is to be tuned. We also define our (noisy) measurements:

$$\tilde{h}(\tilde{x}, w) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x} + \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \tag{9}$$

where the measurement noise w is to be measured via calibration or tuned.

We can then use the standard EKF design as discussed in [1] to obtain the estimated state $\hat{x}[k]$ and variance $P[k]$ at any time step k. The initialization of the state is $\hat{x}(0) = \begin{bmatrix} x_1(0) & 0 & x_3(0) & 0 \end{bmatrix}^\top$ since we have access to the initial sensor measurements. The initialization of the variance is arbitrary, and we use $P(0) = 0.1\, I_{4x4}$. At each time step, we perform first a prediction/process update, obtaining

$$\begin{aligned} \hat{x}_p[k] &= F\left(\hat{x}[k-1], u[k-1]\right) \\ P_p[k] &= AP[k-1]A^\top + J\Sigma_{vv}J^\top \end{aligned} \tag{10}$$

where the subscript p indicates a predicted quantity, $A = \frac{\partial F}{\partial \tilde{x}}\big|_{\tilde{x}=\hat{x}[k-1], u=u[k-1], v=0}$, $J = \frac{\partial \tilde{h}}{\partial \tilde{x}}\big|_{\tilde{x}=\hat{x}[k-1], w=0}$, and Σ_{vv} is the covariance matrix of the process noise v. Note all these matrices may be time varying (indeed A is), but those which are not (in this case L and Σ_{vv} by design) can be pre-computed to improve runtime. If we then take measurements from the sensors, we can perform a measurement update, obtaining

$$\begin{aligned} K &= P_p[k]H^\top \left(HP_p[k]H^\top + M\Sigma_{ww}M^\top\right)^{-1} \\ \hat{x}[k] &= \hat{x}_p[k] + K\left(y[k] - \tilde{h}(\hat{x}_p, 0)\right) \\ P[k] &= (I - KH)\, P_p[k]\, (I - KH)^\top \\ &\qquad + KM\Sigma_{ww}M^\top K^\top \end{aligned} \tag{11}$$

where $H = \frac{\partial h}{\partial \tilde{x}}\big|_{\tilde{x}=\hat{x}_p[k], w=0}$, $M = \frac{\partial \tilde{h}}{\partial w}\big|_{\tilde{x}=\hat{x}_p[k], w=0}$, and Σ_{ww} is the covariance matrix of the measurement noise w. Again, these matrices may all be time varying, but in this case are all constant with time (either naturally by the plant or by design) so can be pre-computed.

In our simulation, we defined $\Sigma_{vv} = 0.01\, I_{4x4}, \Sigma_{ww} = 0.01\, I_{2x2}$. They may be tuned or calibrated during the hardware section of the project. For a full list of the matrices used in the observer, see Appendix A.

IV. DESIGN OF PROPOSED CONTROLLERS

A. PID-Wrapped LQR

The first controller we decided to implement was a PID-wrapped LQR controller. This decision was a result of some preliminary testing with a pure PID controller and a pure LQR controller. We observed that the PID controller tracked the reference trajectory $r_t(t)$ considerably better than the LQR controller but used significantly more energy. This qualitatively makes sense as the PID controller simply does not penalize energy usage, while LQR balances a quadratic cost for tracking and energy. We therefore decided to combine the two controllers.

Specifically, we noticed that the x_3, x_4 subsystem is in fact linear, and that the energy cost we are evaluated with is quadratic. These facts lend the x_3, x_4 subsystem to be controlled well by an LQR controller, so we implemented an LQR which determines its reference based on a cascaded PID wrapper. After tuning, we found that inclusion of integral and differential gains (at least in simulation) detracted from the performance of the controller. Therefore, the exact structure of the controller is

$$\begin{aligned} x_{2,d} &= \dot{r}_t - k_{p,1}\left(x_1 - r_t\right) \\ x_{3,d} &= \frac{7L}{5gr_g}\left(\ddot{r}_t - k_{p,2}\left(x_2 - x_{2,d}\right)\right) \\ x_{4,d} &= -k_{p,3}\left(x_3 - x_{3,d}\right) \\ u &= -G\begin{bmatrix} x_3 - x_{3,d} \\ x_4 - x_{4,d} \end{bmatrix} \end{aligned} \tag{12}$$

where the gains $k_{p,i}$ are to be tuned and G is the standard LQR gain obtained by solving the Discrete Algebraic Riccati Equation (DARE). Note that Simulink does not support MATLAB's built-in `lqr()` or `dlqr()` functions, so a fast DARE solver was explicitly implemented, with reference to [2].

B. Approximate Feedback Linearization

There are several reasons we chose to implement feedback linearization as one of our controllers. First, we note that the system is an affine system, which suggests that feedback linearization may be possible. Furthermore, we note that the reference trajectory is composed of either sinusoidal or square waves. The sinusoidal references are smooth and infinitely differentiable. Meanwhile, the square references are piecewise constant, and are differentiable except at a finite set of points; we therefore set all desired derivatives to zero at those points, making the derivatives smooth and infinitely differentiable. By taking up to the fourth-order derivatives of the reference trajectory, we may be able to, by selecting the output as

$$y = h(x) = x_1 \tag{13}$$

and have a relative degree of $r = n = 4$ without any zero dynamics. In reality, an analysis of the output derivatives shows that the relative degree is not well defined. However, we can follow the techniques in [3] to use feedback linearization for an approximate system, which does give a well-defined relative degree of 4.

We apply the standard feedback linearization change of variables and control law:

$$\begin{aligned} \xi_i &= y^{(i-1)} \\ u &= \frac{1}{L_g L_f^3 h}\left(-L_f^4 h - \sum_{i=1}^{4} k_i \xi_i\right) \end{aligned} \tag{14}$$

where $L_\star(\cdot)$ is the Lie derivative of $(\cdot)$ in the direction of $\star$. Hauser's work [3] presents two methods to approximate the system such that a well-defined relative degree is obtained. In both methods, we find $L_g L_f^3 h = \frac{5 g r_g K \cos x_3}{7 L \tau}$. $L_f^4 h$ using the "worse" approximation described by [3] is equal to $-\frac{5 g r_g}{7 L \tau} x_4 \cos x_3 + x_4^2 \sin x_3$, while using the "better" approximation yields an expression with considerably more terms. We select the gains k_i such that the matrix

$$A_k = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -k_1 & -k_2 & -k_3 & -k_4 \end{bmatrix} \tag{15}$$

is Hurwitz, and the dynamics via approximation is controllable, allowing the gains k_i to be arbitrarily tuned.

C. Safety

To guarantee that the ball remains safely on the beam, we augment both controllers with a safety filter based on a Control Barrier Function. The approach formally ensures that the system state remains within a predefined safe set. In our design following [4], we define a safe set for the ball's position using the barrier function

$$h(x) = (0.175)^2 - p_{\text{ball}}^2,$$

where p_{ball} is the ball's position. This choice reflects our observation that the ball trajectories remain within 15 cm, and thus we enforce a conservative safety margin by setting the safe limit to 17.5 cm. The ball is safe as long as $h(x) > 0$.

The CBF condition is formulated to ensure that the safe set is forward invariant. Specifically, we require that

$$L_f^2 h(x) + L_g^2 h(x)\, u + 2\lambda\, \dot{h}(x) + \lambda^2\, h(x) \geq 0,$$

where $\lambda > 0$ (set as $\lambda_{\text{cbf}} = 10$ in our code) determines the aggressiveness of the safety enforcement.

Our controller first computes a nominal control input. If the nominal control input does not satisfy the CBF condition, we project it onto the set of safe controls by solving the scalar inequality [4]

$$L_g^2 h(x)\, u + \left(L_f^2 h(x) + 2\lambda\, \dot{h}(x) + \lambda^2\, h(x)\right) \geq 0.$$

This safety filter guarantees that, regardless of the nominal command, the system will not drive the ball into an unsafe region (i.e., $h(x) \leq 0$). We test the safety of our controllers by setting the reference trajectory to be consistently unsafe ($z = 0.19$ cm). The controller drives the ball very close to the desired position but does not ever violate the safety constraints. For an illustration of the controller safety, see the media in the repository linked at the end of Section I.

V. SIMULATION RESULTS

The system dynamics was simulated on three reference trajectories. A sinusoidal (amplitude 0.075 m) and a square (amplitude 0.05 m) trajectories with a period of 8 seconds were set up. In addition, a tester reference trajectory composed of alternating square and sine waves was created. The period of the trajectory swept between 6 and 10 seconds, the amplitude of the sinusoidal parts swept between 0 and 15 cm, and the amplitude of the square parts swept between 0 and 10 cm, all as specified in the project requirement. For fair comparison, the ball was initiated at the origin when using the square and sine wave trajectories, but the starting position for the tester trajectory was initiated at $x_1 = -0.19$ m to test initiation robustness.

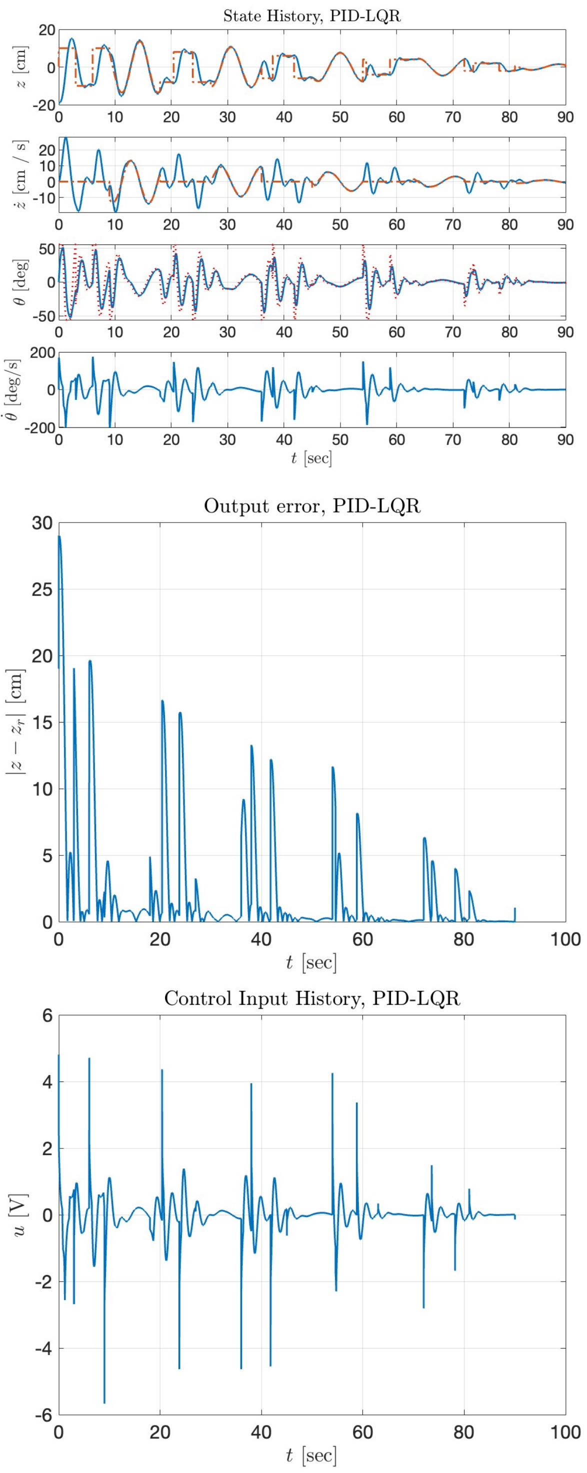

FIG. 2: Results: PID-Wrapped LQR control

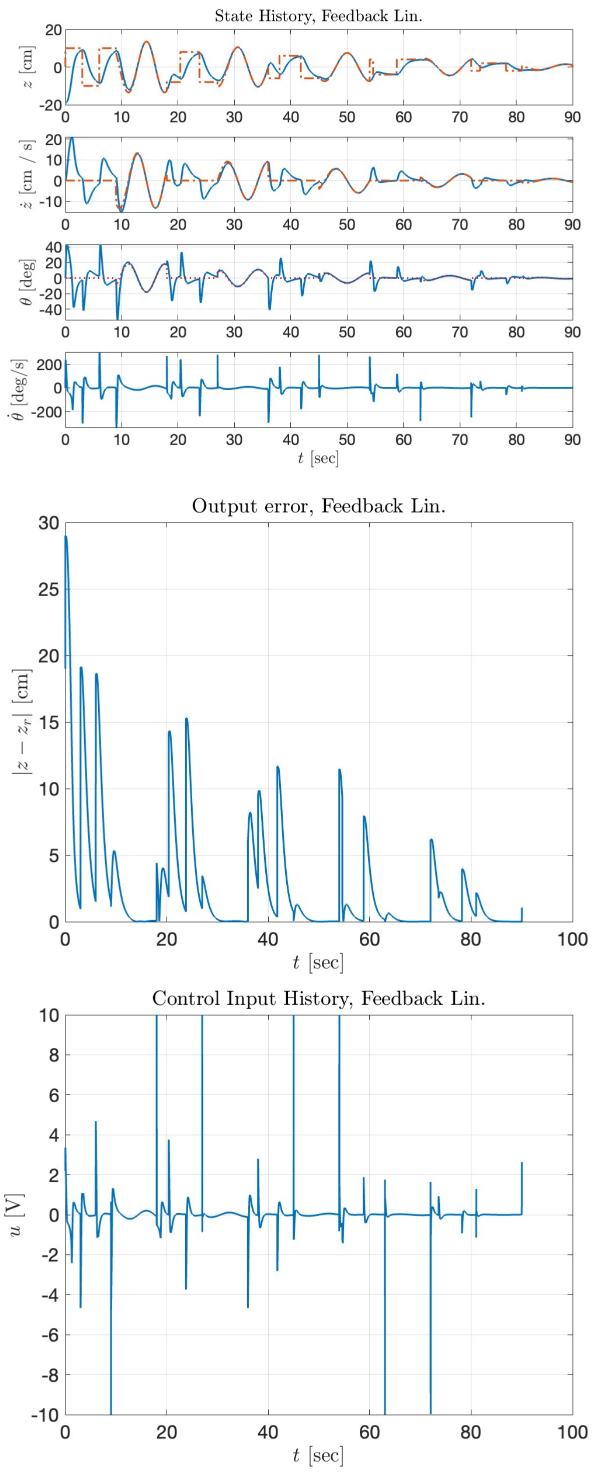

FIG. 3: Results: Approximate Feedback Linearization

We used the tester trajectory was used to tune the PID gains, and the Q, R matrices for the PID-LQR controller, as well as pole placements for the Feedback Linearization controller. Several results using both controllers are shown in Figures 2, 3, including the State History, Output Error, and Control Input in the duration of simulation. Note that all results of the Feedback Linearization controller were performed with the "worse" approximation technique; we did not observe significant improvements using the "better" approximation technique.

We also tabulate results for various reference trajectories below to facilitate the technical discussion of our controller performance.

TABLE II: Results for various controllers and references

Controller	PID-LQR			Feedback Lin.		
Trajectory	Tester	Sine	Square	Tester	Sine	Square
Tracking Cost	4.30	0.01	3.10	5.01	0.01	4.10
Energy Cost	1.21	0.03	1.34	1.34	0.02	0.57
Safety Cost	0.00	0.00	0.00	0.00	0.00	0.00
Total Cost	5.51	0.04	4.44	6.35	0.03	4.67

Overall, the PID-Wrapped LQR had a better performance on square waves, while the Approximate Feedback Linearization performs extremely well on sinusoidal waves. This is apparent both qualitatively from Figures 2, 3, and numerically from the costs in Table II. It seems the PID-LQR controller performs slightly better on the combined tester trajectory, but this may be due to the square waves with high amplitude and low period at the beginning of the simulation. Noting this, we consider to implement both controllers in hardware testing, along with a program to detect when the reference is square or sinusoidal, and choose to the better controller for that specific segment of the trajectory.

We also note that the Feedback Linearization controller, despite generally using less energy, occasionally saturates the control input. This occurs specifically at times when the reference changes from square to sinusoidal or vice versa.

Appendix A: EKF Matrices

Listed below are the matrices used in the EKF observer as defined in Section III.

$$A = \begin{bmatrix} 1 & dt & 0 & 0 \\ \alpha\, dt & 1 & \beta\, dt & \gamma\, dt \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 - \frac{dt}{\tau} \end{bmatrix}$$

$$J = I_{4x4}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$M = I_{2x2}$$

with

$$\alpha = \frac{5}{7}\left(\frac{r_g}{L}\right)^2 x_4^2 \cos^2 x_3^2$$

$$\beta = \frac{5g}{7r_gL}\cos x_3 + \frac{5}{7}\left(\frac{L}{2} - x_1\right)\left(\frac{r_g}{L}\right)^2 x_4^2 \sin(2x_3)$$

$$\gamma = -\frac{10}{7}\left(\frac{L}{2} - x_1\right)\left(\frac{r_g}{L}\right)^2 x_4 \cos^2 x_3$$

[1] M. Mueller, ME C231B lecture notes, University of California, Berkeley (2024).

[2] E. K. W. Chu, H. Y. Fan, W. W. Lin, and C. S. Wang, Structure-preserving algorithms for periodic discrete-time algebraic Ricatti equations, International Journal of Control **77**, 767 (2003).

[3] J. Hauser, S. Sastry, and P. Kokotovic, Nonlinear control via approximate input-output linearization: the ball and beam example, IEEE Transactions on Automatic Control **37**, 392 (1992).

[4] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, Control barrier functions: Theory and applications, European Control Conference **18**, 3420 (2019).