TRAJECTORY-TRACKING CONTROL OF THE BALL-AND-PLATE SYSTEM

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

by

Dominic E. Riccoboni

March 2023

COMMITTEE MEMBERSHIP

TITLE: Trajectory-Tracking Control of the Ball-and-Plate System

AUTHOR: Dominic E. Riccoboni

DATE SUBMITTED: March 2023

COMMITTEE CHAIR: William R. Murray, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: Charles T. Refvem
Lecturer of Mechanical Engineering

COMMITTEE MEMBER: Clay McKell, Ph.D.
Lecturer of Electrical Engineering

ABSTRACT

Trajectory-Tracking Control of the Ball-and-Plate System

Dominic E. Riccoboni

The Mechatronics group in the Mechanical Engineering department of Cal Poly is interested in creating a demonstration of a ball-and-plate trajectory tracking controller on hardware. The display piece will serve to inspire engineering students to pursue Mechatronics and control theory as an area of study. The ball-and-plate system is open-loop unstable, underactuated, and has complicated, nonlinear equations of motion. These features present substantial challenges for control - especially if the objective is trajectory tracking. Because the system is underactuated, common nonlinear trajectory tracking control techniques are ineffective. This thesis lays out a theoretical foundation for controlling the hardware.

Several important concepts related to ball-and-plate trajectory tracking control are presented. Models of the system, with various assumptions, are given and used in deriving control law candidates. To limit project scope, reasonable control criteria are introduced and used to evaluate designs from the thesis. Several control architectures are explored, these being Full-State Feedback with Integral Action, Single-Input-Single-Output Sliding Mode, and Full-State Feedback with Feed Forward. The mathematical reasoning behind each is detailed, simulation results are shown to validate their practicality and demonstrate features of the architectures, and trajectory similarity measure studies are produced to evaluate controller performance for a wide range of setpoint functions.

The Full-State Feedback with Feed Forward controller is recommended based on its theoretical advantages and compliance with the control criteria over the competing

designs. The control architecture has a proof of asymptotic tracking in the linear model, has excellent performance in simulations that use a nonlinear plant model, and produces the most pleasing visual experience when viewed in animation.

ACKNOWLEDGMENTS

I would like to thank Charlie and Dr. Murray for many years of mentorship and friendship. Their guidance, knowledge, and wisdom have helped make me the person I am today, and I appreciate everything they have done for me. I would also like to thank Clay for the wealth of controls knowledge he imparted on me, and for all of his time and effort as a member of my committee.

I would like to thank my peers, Scott and Zach. Our conversations got me through many challenging road blocks, and I appreciated having friends to struggle alongside. I wish them both success in their engineering careers.

Finally, I would like to thank my friends and family for their love and support through many challenging times. A special thanks to my parents for always believing in me and encouraging me, and to my nonno, for pushing me to finish my thesis when I most needed the nudge.

TABLE OF CONTENTS

Page

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER

1 Background 1

1.1 Model 2

1.1.1 Coupled Nonlinear Model 5

1.1.2 Decoupled Nonlinear Model 6

1.1.3 Decoupled Linearized Model 12

1.2 Literature Review 15

1.2.1 The Asymptotic Tracking Problem 15

1.2.2 Trajectory Similarity Measures (TSMs) 17

1.2.3 Underactuation 22

1.2.4 Ball-and-Plate-Specific Trajectory Tracking Schemes 27

2 Control Criteria 33

2.1 Defining the Tracking Problem for the Ball and Plate 33

2.2 Selecting an Appropriate Trajectory Similarity Measure 35

2.3 Additional Control Criteria 36

2.4 Practical Considerations and Constraints 39

3 Control Architectures and Simulation Results 41

3.1 Linear Full-State Feedback with Integral Action 45

3.1.1 Control Law 46

3.1.2 Simulation Results 49

3.2 SISO Sliding Mode 57
3.2.1 Control Law 58
3.2.2 Simulation Results 63
3.3 Full-State Feedback with Feed Forward 76
3.3.1 Control Law 78
3.3.2 Simulation Results 84
3.3.3 Turning off the Feed Forward Torque 94
4 Lock-Step Euclidean Distance Study 104
5 Discussion, Conclusion, and Future Work 107
5.1 FSFB with Integral Action Discussion 107
5.2 SISO Sliding Mode Discussion 108
5.3 Recommendation: Full-State Feedback with Feed Forward 110
5.4 Conclusion 111
5.5 Future Work 112
BIBLIOGRAPHY 114
APPENDICES
A Additional LSED Studies 116

LIST OF TABLES

Table		Page
1.1	Parameter Definitions and Numerical Values	4
1.2	TSM Timing Categories	19
3.1	Error State Definitions	44
3.2	SISO Sliding Mode Control Parameters	65

LIST OF FIGURES

Figure		Page
1.1	Render of Ball-and-Plate Platform SolidWorks Assembly Mangin [3, p.25]	2
1.2	System 1 Schematic Richter [6, p.8]	3
1.3	System 2 Schematic Richter [6, p.8]	3
1.4	The Free-Body Diagram (FBD) Equated to the Kinetic Diagram (KD) for the Ball Alone Richter [6, p.21]	8
1.5	FBD = KD For the Ball and Plate Together Richter [6, p.24]	9
1.6	TSMs Demonstration. Tao et al. [9, p.649]	20
1.7	System 1 Schematic Richter [6, p.8]	25
2.1	3D Animation Example	38
2.2	Motor Actuation Mechanism Richter [6, p.16]	39
3.1	Quadrifolim Rose Curve with $r = 0.15$ $[m]$	42
3.2	Quadrifolim Rose Curve with $r = 0.15$ $[m]$	43
3.3	Outer Control Architecture - FSFB With Integral Action	49
3.4	x Axis Control Architecture - FSFB With Integral Action	49
3.5	Regulation Response - FSFB With Integral Action	51
3.6	Step Response, $x_s = -0.15$ $[m]$ - FSFB With Integral Action	53
3.7	Quadrifolium Response, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - FSFB With Integral Action	54
3.8	Quadrifolium Response, Ball Path, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - FSFB With Integral Action	56
3.9	Outer Control Architecture - SISO Sliding Mode	63

3.10 x Axis Control Architecture - SISO Sliding Mode 64

3.11 $\mathbf{s_1}$ sliding variable computation - SISO Sliding Mode 64

3.12 Quadrifolium Response, Output vs. State, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode 66

3.13 Quadrifolium Response, Ball Path, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode 67

3.14 Quadrifolium Response, $\mathbf{s_1}$, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode 67

3.15 Quadrifolium Response, T_β and β, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode 68

3.16 Quadrifolium Response, Output vs. State, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode 70

3.17 Quadrifolium Response, T_β and β, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode 71

3.18 Quadrifolium Response, $\hat{T}_\beta$, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode 72

3.19 Quadrifolium Response, s_1, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode 72

3.20 Regulation Response - SISO Sliding Mode 73

3.21 Regulation Response, T_β and β - SISO Sliding Mode 74

3.22 Step Response - SISO Sliding Mode 75

3.23 Step Response, T_β and β - SISO Sliding Mode 76

3.24 Outer Control Architecture - FSFB With Feed Forward 85

3.25 x Axis Control Architecture - FSFB With Feed Forward 85

3.26 Regulation Response - FSFB With Feed Forward 87

3.27 Step Response, $x_s = -0.15\ [m]$ - FSFB With Feed Forward 88

3.28 Quadrifolium Response, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - FSFB With Feed Forward 90

3.29 Quadrifolium Response, $r = 0.15$ $[m]$, $T_r = 3.75$ $[s]$ - FSFB With Feed Forward 92

3.30 Quadrifolium Response, Ball Path, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - FSFB With Feed Forward 93

3.31 Quadrifolium Response, Ball Path, $r = 0.15$ $[m]$, $T_r = 3.75$ $[s]$ - FSFB With Feed Forward 94

3.32 Quadrifolium Response, Linear Plant Model, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - Feed Forward Turned Off 95

3.33 Quadrifolium Response, Linear Plant Model, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - Feed Forward Kept On 96

3.34 e_x zoomed in - Feed Forward Turned Off 97

3.35 e_x zoomed in - Feed Forward Turned On 97

3.36 $T_\beta - \hat{T}_\beta$ - Feed Forward Turned Off 97

3.37 $T_\beta - \hat{T}_\beta$ Zoomed In - Feed Forward Turned Off 98

4.1 LSED Study, Quadrifolium Rose Curve, n = 2, d = 1 105

4.2 LSED Study, Dürer Folium Rose Curve, n = 1, d = 2 106

4.3 LSED Study, Trifolium Rose Curve, n = 3, d = 1 106

A.1 LSED Study, Circle Rose Curve, n = 1, d = 1 116

A.2 LSED Study, Limaçon Trisectrix Rose Curve, n = 1, d = 3 116

A.3 LSED Study, Rose Curve, n = 2, d = 3 117

A.4 LSED Study, Rose Curve, n = 3, d = 5 117

Chapter 1

BACKGROUND

The Mechatronics group in the Mechanical Engineering department of Cal Poly is interested in showcasing a ball-and-plate trajectory-tracking controller on hardware. The device will serve as a display piece intended to inspire engineering students to pursue Mechatronics and control theory as an area of study. The ball-and-plate system is also of scholarly interest, as it has properties that prove challenging for control algorithm development and can serve as a medium to explore mathematical techniques in control theory.

This section introduces the ball-and-plate system and outlines a few different dynamic models. It begins on the topic of trajectory similarity measures, details some properties relevant to the trajectory-tracking control problem, and reviews some trajectory-tracking control schemes already developed. These are done with reference to prior work in the Cal Poly Mechatronics group, and papers from literature.

A graduate mechatronics student at Cal Poly has developed prototype ball-and-plate hardware, Mangin [3]. A render of the SolidWorks model for the hardware created by Mangin is shown in Fig. 1.1. This thesis also builds upon model validation and simulation work done by Richter [6]. The dynamic models were developed by Richter, and are only detailed in this thesis to the extent necessary to facilitate control architecture discussion in Chapter 3. The same simplifying assumptions implemented

by Richter[1] were implemented for the plant model used in simulations here, but the physical parameters have changed since the prototype was completed by Mangin.

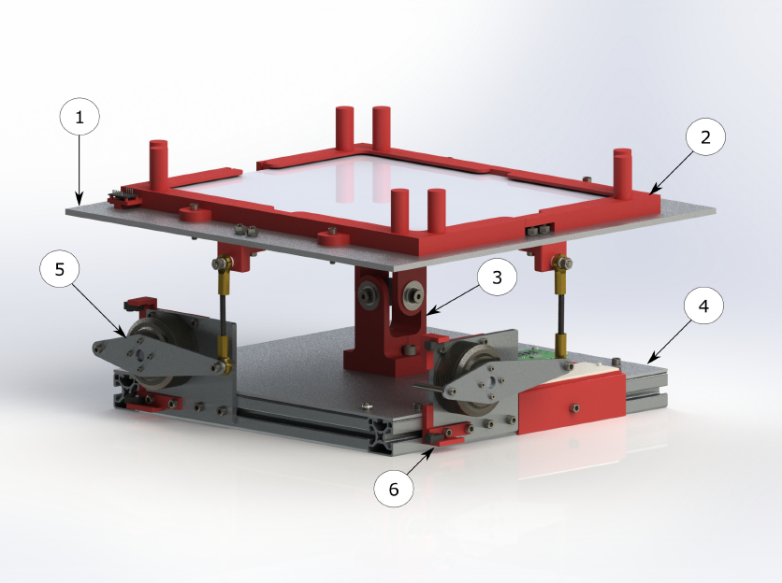


Figure 1.1: Render of Ball-and-Plate Platform SolidWorks Assembly Mangin [3, p.25]

1.1 Model

This section gives an overview of the system model used in analysis and simulation of the control systems presented. First, a coupled nonlinear model is introduced, though explicit formulas are omitted here. Then a decoupled nonlinear model is produced using some simplifying assumptions. From there, linearizing assumptions and the decoupled linear model are outlined. Schematics of the system, given by Richter [6], are shown in figures 1.2 and 1.3.

[1]The model primarily used assumes that there is no damping in the u-joint, no rolling friction for the ball, that the ball rolls without slip, and that the ball does not spin about its z axis in the plate frame.

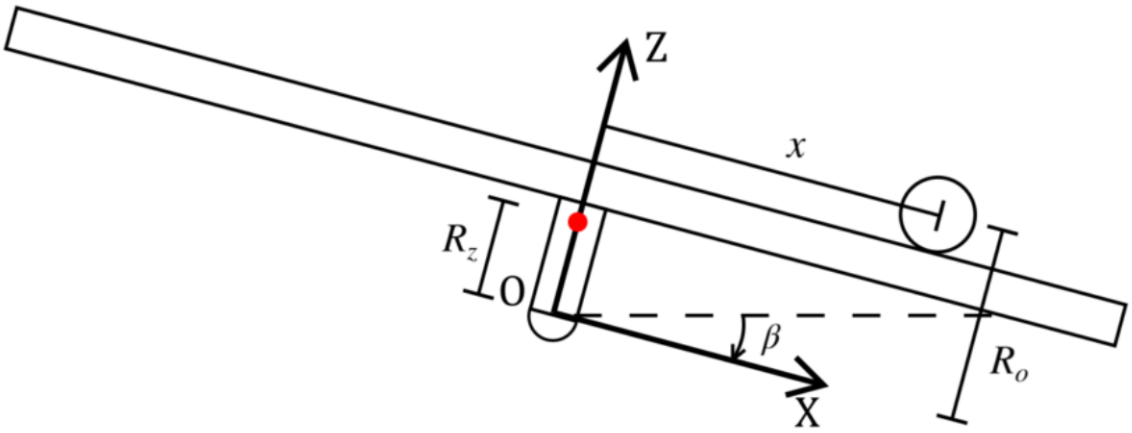


Figure 1.2: System 1 Schematic Richter [6, p.8]

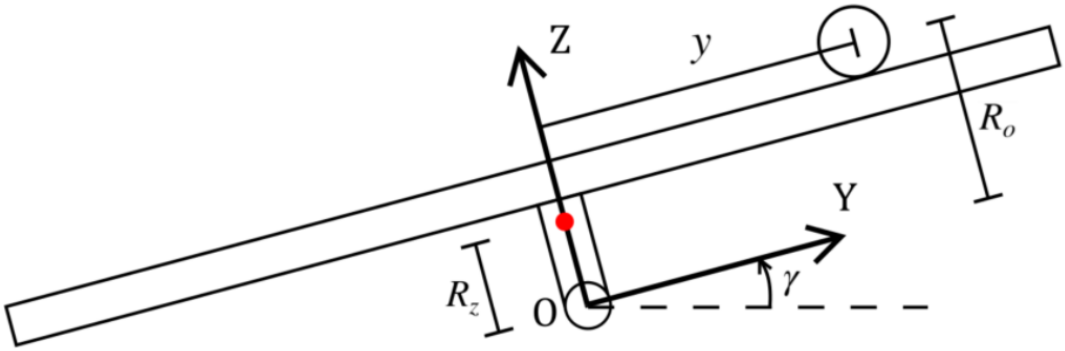


Figure 1.3: System 2 Schematic Richter [6, p.8]

Table 1.1 defines the parameters used in the models, and their approximate values for the physical system. Values in this table were used to produce numerical simulation results in later chapters.

Table 1.1: Parameter Definitions and Numerical Values

r_b	Radius of the ball	0.0195 [m]
m_b	Mass of the ball	0.225 [Kg]
z_b	Distance between center of the u-joint and center of gravity of the ball	0.122 [m]
z_p	Distance between center of the u-joint and center of gravity of the plate	0.0857 [m]
m_p	Mass of the plate	3.98 [Kg]
$I_{p,\mathrm{xx}}$	Mass moment of inertia of the plate about its x axis[2].	0.079 $[\frac{Kg\ m}{s^2}]$
T_{max}	Max u-joint torque. Applies to T_β and T_γ inputs.	1.3 [Nm]

The notation used in this thesis differs in that R_z and R_o of figures 1.2 and 1.3 (the z-direction distances from the center of the u-joint to the centers of mass of the plate and the ball, respectively) are replaced with z_p and z_b (defined in Table 1.1).

More detailed derivations and model analysis than are present in this chapter are given by Richter [6]. However pertinent model details are included here to give readily available context to topics discussed later. These details are particularly helpful in Chapter 3 on control architectures. Additionally, the parameters of the physical system have since changed[3], and there exists notation and formulations specific to this thesis that must be included.

[2]The coordinate system used is introduced in Figures 1.2 and 1.3. $I_{p,\mathrm{yy}}$ is assumed to be identical to $I_{p,\mathrm{xx}}$ and $I_{p,\mathrm{zz}}$ never enters into the dynamics. All products of inertia are assumed to be zero.

[3]And therefore the matrices have changed.

1.1.1 Coupled Nonlinear Model

The coupled nonlinear model of the ball-and-plate system was developed by Richter [6] using a Newton-Euler modeling approach. This thesis uses a substantively identical parametric model. The full nonlinear model that couples the dynamics of the system is far too cumbersome to display in a fully expanded form. Despite the cumbersome nature of the full nonlinear model, it still has utility in that symbolic expressions for it, generated using the MATLAB® Symbolic Math Toolbox, can be converted into MATLAB® functions for use in numerical simulation.

The nonlinear equation of motion (1.1), most generally, is a nonlinear function of the state vector $\mathbf{x}$ and the input vector $\mathbf{u}$.

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}) \tag{1.1}$$

The state vector, rearranged from that of Richter, is (1.2).

$$\mathbf{x} = [\ x \quad \dot{x} \quad \beta \quad \dot{\beta} \quad y \quad \dot{y} \quad \gamma \quad \dot{\gamma}\ ]^{\mathsf{T}} \tag{1.2}$$

As defined by Richter, x and y states are the positions and velocities of the ball in the plate frame, and the β and γ states are the u-joint angles (one for each u-joint axle) and angular velocities. These states are shown in the schematics of figures 1.2 and 1.3. The inputs to the system are the torques about the u-joint axles and the input vector is (1.3).

$$\mathbf{u} = [\ T_\beta \quad T_\gamma\ ]^{\mathsf{T}}. \tag{1.3}$$

The system is control-affine, meaning that state-dependant coefficients on the input variables can be factored out and the nonlinear equations can be written as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + g(\mathbf{x})\mathbf{u} \tag{1.4}$$

The matrix $g(\mathbf{x})$ is, once again, far too cumbersome to display. Its dimensions, however, are known to be 8×2. This significant result is related to the fact that the system is uderactuated - a concept that will be discussed in Section 1.2.3.

1.1.2 Decoupled Nonlinear Model

The decoupled nonlinear model comes from imposing simplifying assumptions on the coupled nonlinear model. The decoupled system can be approximated as two entirely separate "teeter-totters", one for each of the u-joint rotations. System 1 involves x and β variables alone, and is shown in Fig. 1.2. System 2 involves y and γ and is shown in Fig. 1.3

In each "teeter-totter" system, the effect of the other system is considered negligible. For system 1, the pertinent state variables are given by (1.7), and for system 2, they are given by (1.8). The decoupled nonlinear state equations have the form:

$$\dot{\mathbf{x}}_1 = \mathbf{f}_1(\mathbf{x}_1) + g_1(\mathbf{x}_1)\mathbf{u}_1 \tag{1.5}$$

$$\dot{\mathbf{x}}_2 = \mathbf{f}_2(\mathbf{x}_2) + g_2(\mathbf{x}_2)\mathbf{u}_2 \tag{1.6}$$

where the decoupled state vectors are

$$\mathbf{x}_1 = [\ x \quad \dot{x} \quad \beta \quad \dot{\beta}\ ]^\mathsf{T} \tag{1.7}$$

$$\mathbf{x}_2 = [\ y \quad \dot{y} \quad \gamma \quad \dot{\gamma}\ ]^\mathsf{T} \tag{1.8}$$

and the decoupled inputs are

$$\mathbf{u}_1 = T_\beta \tag{1.9}$$

$$\mathbf{u}_2 = T_\gamma. \tag{1.10}$$

The decoupled nonlinear model can be derived by making assumptions when summing moments in the kinetic analysis for the full nonlinear model. For both "teeter-totter" systems, system 1 and system 2, the assumption is that the u-joint angle and angular velocity of the other system are zero. For system 1, γ and $\dot{\gamma}$ are set to zero, and for system 2, β and $\dot{\beta}$ are set to zero.

Free-body and kinetic diagrams for the ball alone are given in Fig. 1.4. The vectors $\mathbf{F}_W, \mathbf{F}_{fr}$ and $\mathbf{F}_N$ represent the weight of the ball, the friction force, and the normal force, respectively.

The sum of moments are taken about the contact point (point c) of the ball, i.e.,

$$\sum_{FBD} M_c = \sum_{KD} M_c.$$

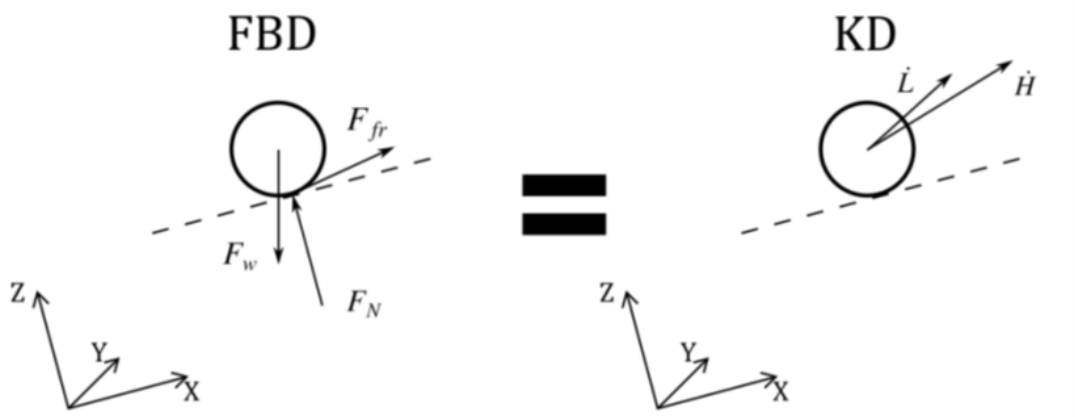


Figure 1.4: The Free-Body Diagram (FBD) Equated to the Kinetic Diagram (KD) for the Ball Alone Richter [6, p.21]

For the sum of moments about point c in the y direction, with the decoupling assumptions, the result is a moment equation that only contains variables of $\mathbf{x}_1$ (1.7). This moment equation is

$$g\, m_b\, r_b\, \sin(\beta) = \frac{7\, m_b\, r_b\, \ddot{x}}{5} + \ddot{\beta}\, m_b\, r_b\, z_b + \frac{2\, \ddot{\beta}\, m_b\, {r_b}^2}{5} - \dot{\beta}^2\, m_b\, r_b\, x,$$

which simplifies to

$$g\, \sin(\beta) = \frac{7\, \ddot{x}}{5} + \ddot{\beta}\left(z_b + \frac{2\, r_b}{5}\right) - \dot{\beta}^2\, x. \tag{1.11}$$

Doing the same for the sum of moments about point c in the x direction yields a moment equation that only contains variables of $\mathbf{x}_2$ (1.8). This moment equation is given by

$$g\, m_b\, r_b\, \sin(\gamma) = -\frac{7\, m_b\, r_b\, \ddot{y}}{5} + \ddot{\gamma}\, m_b\, r_b\, z_b + \frac{2\, \ddot{\gamma}\, m_b\, {r_b}^2}{5} + \dot{\gamma}^2\, m_b\, r_b\, y,$$

which simplifies to

$$g \sin(\gamma) = -\frac{7\,\ddot{y}}{5} + \ddot{\gamma}\left(z_b + \frac{2\,r_b}{5}\right) + \dot{\gamma}^2\, y. \tag{1.12}$$

For each system, there are two unknown accelerations, so an additional equation is needed. The additional equation for each system can be found by considering the free-body and kinetic diagrams for the ball and plate together, Fig. 1.5.

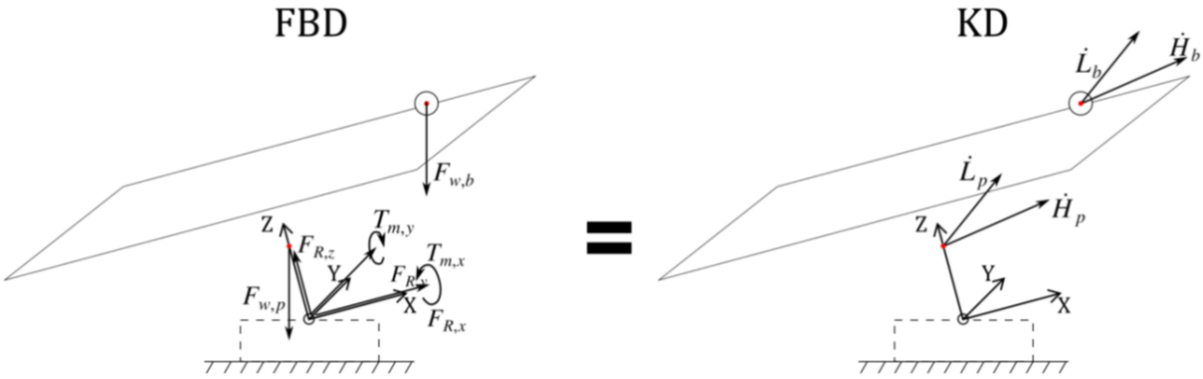


Figure 1.5: FBD = KD For the Ball and Plate Together Richter [6, p.24]

The the u-joint torques $T_{m,y}$ and $T_{m,x}$ in Fig. 1.5 correspond to T_γ and T_β respectively[4]. The x, y, and z-direction reaction forces at the u-joint are $F_{R,x}$, $F_{R,x}$, and $F_{R,z}$ and the force of the plate's weight is $F_{w,p}$. To get the two additional equations, moments are summed at the u-joint (point u), i.e.,

$$\sum_{FBD} M_u = \sum_{KD} M_u,$$

[4]These u-joint torques, are idealized, and the hardware actually contains actuators that push on the plate through connecting rods. As a consequence, the effective u-joint torques need to be estimated. This detail is discussed later in Section 2.4.

and the same decoupling assumptions used to derive (1.11) and (1.12) are implemented to yield

$$T_\beta + g\,m_b\,x\,\cos(\beta) + g\,m_b\,z_b\,\sin(\beta) + g\,m_p\,z_p\,\sin(\beta) =$$

$$\ddot{\beta}\left(m_b\,x^2 + \frac{2\,m_b\,{r_b}^2}{5} + I_{p,\mathrm{xx}} + m_b\,{z_b}^2 + m_p\,{z_p}^2\right)$$

$$+\ \ddot{x}\left(m_b\,z_b + \frac{2\,m_b\,r_b}{5}\right) + 2\,\dot{\beta}\,m_b\,x\,\dot{x} \quad (1.13)$$

and

$$T_\gamma + g\,m_b\,z_b\,\sin(\gamma) + g\,m_p\,z_p\,\sin(\gamma) - g\,m_b\,y\,\cos(\gamma) =$$

$$\ddot{\gamma}\left(m_b\,y^2 + \frac{2\,m_b\,{r_b}^2}{5} + I_{p,\mathrm{xx}} + m_b\,{z_b}^2 + m_p\,{z_p}^2\right)$$

$$+\ \ddot{y}\left(-m_b\,z_b - \frac{2\,m_b\,r_b}{5}\right) + 2\,\dot{\gamma}\,m_b\,y\,\dot{y}. \quad (1.14)$$

Together, (1.11) and (1.13) produce (1.5) when $\ddot{x}$ and $\ddot{\beta}$ are solved for and the system is put in the first-order, control-affine representation. Equations (1.12) and (1.14) produce (1.6) in the same fashion. For system 1, without carrying out the matrix operations, the accelerations in the state-space representation are given by

$$\begin{bmatrix} \ddot{x} \\ \ddot{\beta} \end{bmatrix} = M_1^{-1}\ b_1 \quad (1.15)$$

where

$$M_1 = \begin{bmatrix} \frac{7}{5} & z_b + \frac{2\,r_b}{5} \\ m_b\,z_b + \frac{2\,m_b\,r_b}{5} & m_b\,x^2 + m_b\,{z_b}^2 + m_p\,{z_p}^2 + \frac{2\,m_b\,{r_b}^2}{5} + I_{p,\mathrm{xx}} \end{bmatrix}$$

and

$$b_1 = \begin{bmatrix} \dot{\beta}^2\,x + g\,\sin\left(\beta\right) \\ T_\beta + g\,m_b\,x\,\cos\left(\beta\right) + g\,m_b\,z_b\,\sin\left(\beta\right) + g\,m_p\,z_p\,\sin\left(\beta\right) - 2\,\dot{\beta}\,m_b\,x\,\dot{x} \end{bmatrix}.$$

For system 2, they are given by

$$\begin{bmatrix} \ddot{y} \\ \ddot{\gamma} \end{bmatrix} = M_2^{-1}\,b_2 \tag{1.16}$$

where

$$M_2 = \begin{bmatrix} \frac{7}{5} & -z_b - \frac{2\,r_b}{5} \\ -m_b\,z_b - \frac{2\,m_b\,r_b}{5} & m_b\,y^2 + m_b\,{z_b}^2 + m_p\,{z_p}^2 + \frac{2\,m_b\,{r_b}^2}{5} + I_{p,\mathrm{xx}} \end{bmatrix}$$

and

$$b_2 = \begin{bmatrix} \dot{\gamma}^2\,y - g\,\sin\left(\gamma\right) \\ T_\gamma + g\,m_b\,z_b\,\sin\left(\gamma\right) + g\,m_p\,z_p\,\sin\left(\gamma\right) - g\,m_b\,y\,\cos\left(\gamma\right) - 2\,\dot{\gamma}\,m_b\,y\,\dot{y} \end{bmatrix}.$$

The input vector for system 1 is then

$$g_1(\mathbf{x}_1) = \begin{bmatrix} 0 \\ -\frac{5\,z_b+2\,r_b}{7\,m_b\,x^2+2\,m_b\,{z_b}^2-4\,m_b\,r_b\,z_b+7\,m_p\,{z_p}^2+2\,m_b\,{r_b}^2+7\,I_{p,\mathrm{xx}}} \\ 0 \\ \frac{7}{7\,m_b\,x^2+2\,m_b\,{z_b}^2-4\,m_b\,r_b\,z_b+7\,m_p\,{z_p}^2+2\,m_b\,{r_b}^2+7\,I_{p,\mathrm{xx}}} \end{bmatrix} \tag{1.17}$$

and the input vector for system 2 is

$$g_2(\mathbf{x}_2) = \begin{bmatrix} 0 \\ \frac{5\,z_b+2\,r_b}{7\,m_b\,y^2+2\,m_b\,{z_b}^2-4\,m_b\,r_b\,z_b+7\,m_p\,{z_p}^2+2\,m_b\,{r_b}^2+7\,I_{p,\mathrm{xx}}} \\ 0 \\ \frac{7}{7\,m_b\,y^2+2\,m_b\,{z_b}^2-4\,m_b\,r_b\,z_b+7\,m_p\,{z_p}^2+2\,m_b\,{r_b}^2+7\,I_{p,\mathrm{xx}}} \end{bmatrix}. \tag{1.18}$$

1.1.3 Decoupled Linearized Model

The linearized model comes from taking the Jacobian of the coupled nonlinear equations of motion with respect to both the states and the inputs. The resulting matrices are evaluated at a stable operating point where the state vector (1.7) is zeroed. In this state, the plate is perfectly horizontal and the ball is in the center of the plate. A linear state-space model results (1.19) where A and B are matrices of constant coefficients (Richter [6]).

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{1.19}$$

Evaluation of the Jacobian matrices at the operating point - with numerical parameters substituted in - yields numerical A and B matrices and the full linearized

state-space model (1.20). Results here differ from those of Richter and follow from the prototype designed by Mangin.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\beta} \\ \ddot{\beta} \\ \dot{y} \\ \ddot{y} \\ \dot{\gamma} \\ \ddot{\gamma} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1.8786 & 0 & 4.1026 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 20.2622 & 0 & 31.3281 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1.8786 & 0 & -4.1026 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -20.2622 & 0 & 31.3281 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \beta \\ \dot{\beta} \\ y \\ \dot{y} \\ \gamma \\ \dot{\gamma} \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 \\ -0.8511 & 0 \\ 0 & 0 \\ 9.1798 & 0 \\ 0 & 0 \\ 0 & 0.8511 \\ 0 & 0 \\ 0 & 9.1798 \end{bmatrix} \begin{bmatrix} T_\beta \\ T_\gamma \end{bmatrix} \quad (1.20)$$

The linearized system turns out to be decoupled, and can be determined via linearization starting from the nonlinear decoupled model of Section 1.1.2 rather than from the coupled nonlinear model of Section 1.1.1. To do this, one simply employs the same Jacobian linearization that produced (1.20) starting from the decoupled nonlinear model. The decoupled linear systems have the same two state vectors, $\mathbf{x}_1$ and $\mathbf{x}_2$,

and input vectors, $\mathbf{u}_1$ and $\mathbf{u}_2$, as Section 1.1.2. The linear decoupled, state equations are of the form

$$\dot{\mathbf{x}}_1 = A_1\mathbf{x}_1 + B_1T_\beta \tag{1.21}$$

$$\dot{\mathbf{x}}_2 = A_2\mathbf{x}_2 + B_2T_\gamma. \tag{1.22}$$

Expanded out, equations (1.21) and (1.21) are

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\beta} \\ \ddot{\beta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1.8786 & 0 & 4.1026 & 0 \\ 0 & 0 & 0 & 1 \\ 20.2622 & 0 & 31.3281 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \beta \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 \\ -0.8511 \\ 0 \\ 9.1798 \end{bmatrix} T_\beta \tag{1.23}$$

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{\gamma} \\ \ddot{\gamma} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1.8786 & 0 & -4.1026 & 0 \\ 0 & 0 & 0 & 1 \\ -20.2622 & 0 & 31.3281 & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \gamma \\ \dot{\gamma} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.8511 \\ 0 \\ 9.1798 \end{bmatrix} T_\gamma. \tag{1.24}$$

The effectiveness of this model is discussed by Richter and it is determined that there are potentially non-negligible differences between simulation results for the linear and nonlinear models. Despite these differences, certain control techniques presented in this thesis will utilize systems (1.23) and (1.24) in their derivation. Other control laws discussed will instead utilize the nonlinear decoupled equations in Section 1.1.2. These nonlinear decoupled equations will have less mismatch between the internal model of

the controller and the coupled nonlinear model of Section 1.1.1. These laws will be evaluated in the context of nonlinear model simulations (unless otherwise specified) as well as the physical system. So, in this thesis, the discrepancy between the models will only be discussed to the extent that control law effectiveness is concerned.

1.2 Literature Review

This section gives insight into the trajectory-tracking control problem for the ball-and-plate system and introduces some control schemes present in literature. The asymptotic tracking problem as defined in Applied Nonlinear Dynamics, Slotline and Li [8], is presented. Methods of evaluating control schemes that do not posses a proof of asymptotic tracking are shown in the Trajectory Similarity Measures section, referencing the work of Tao et al. [9]. Underactuation, the primary road block to a nonlinear trajectory-tracking controller for the system, is explored through the work of Tedrake [10]. There, common nonlinear trajectory-tracking control schemes are shown to be ineffective for the ball and plate.

1.2.1 The Asymptotic Tracking Problem

In Applied Nonlinear Dynamics, Slotline and Li [8] present the **Asymptotic Tracking Problem**:

"Given a nonlinear dynamics system described by"

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}) \end{aligned}$$

"and a desired output trajectory $\mathbf{y}_d$, find a control law for the input $\mathbf{u}$ such that, starting from any initial state in a region Ω, the tracking errors $\mathbf{y}(t) - \mathbf{y}_d(t)$ go to zero, while the whole state $\mathbf{x}$ remains bounded." The authors also note that, for practicality, one may require that "$\mathbf{x}$ actually remain 'reasonably' bounded, and, in particular, within the range of validity of the system model."

For our model, there is no consideration for where the edge of the plate is (where the ball would fall off), and there is no consideration for the fact that the plate cannot turn upside down (mechanical constraints in the u-joint). Even if the plate could turn upside down, the model does not capture the fact that the ball would no longer be constrained to the plate and would fall off. Although these unmodelled constraints of the system are not captured in the control criteria of Chapter 2, they are obeyed by simulation results presented in this thesis[5].

For some of the work done in this thesis, the distinction that the state remain bounded is redundant. In the feed-forward controller of Section 3.3.1, the state vector is comprised of error states alone, and the output is the state vector. In this case, if there is a proof that the tracking error will asymptotically approach zero then the state is clearly bounded and no further proof is required. However, for the single-input-single-output (SISO) sliding mode and full-state feedback (FSFB) with integral action controllers of Sections 3.2.1 and 3.1.1, there isn't a known desired trajectory in the entire state vector. As a consequence, a state vector of error states cannot be constructed and the output vector is necessarily a subset of the state vector. For the SISO sliding mode and FSFB with Integral action controllers, one would need to independently prove that both requirements are satisfied.

[5]Unless the simulation results are presenting an invalid or unstable controller response.

This thesis does not attempt to give proofs of asymptotic tracking for the control laws presented because, in all cases, it is impossible to do so[6]. However, it does present a means of evaluating tracking performance for laws where asymptotic tracking cannot occur. The concept of a Trajectory Similarity Measure (TSM) is introduced in Section 1.2.2 and a suitable choice of TSM is presented in Section 2.2.

1.2.2 Trajectory Similarity Measures (TSMs)

Trajectory similarity measures give us a way to quantify, in some sense, the distance between two trajectories in space and time. The various TSMs in the literature treat space and time differently, according to different sets of goals, and each with different definitions of "similar". In "A comparative analysis of trajectory similarity measures ", Tao et al. [9] provide an overview of trajectory similarity measures as well as a survey of the "most widely-known and frequently cited trajectory similarity measures". They give key use cases including "find the most similar trajectory in a collection to a given trajectory, e.g. (Buchin et al. 2011)", clustering, and trajectory classification.

It is clear from the article that TSMs have a wide range of applications and constitute a deep area of study. Our use for TSMs will be relatively simple: to evaluate performance of control systems that cannot achieve perfect asymptotic tracking of a reference signal. If asymptotic tracking could be achieved, there would be no need for TSMs. Any TSM comparing the response to the reference would evaluate to nearly zero[7]. Even if a controller could theoretically asymptotically track, TSMs would prove

[6]This is known immediately, as all controllers are derived starting from simplified models of the system. If there were a controller that contained an internal model of the full nonlinear dynamics, it would be a potential candidate for a proof of asymptotic tracking.

[7]This is, of course, ignoring the transient portion of the response where the initial conditions are being dealt with.

useful in evaluating performance on hardware, as opposed to simulation, where model uncertainties, noise, disturbances, etc. degrade the controller.

Tao et al. give some characterizing features of TSMs that will prove useful in selecting an appropriate one for our use case (the selection is done in Section 2.2). The most relevant concept is that of "Relative versus absolute measures". The authors state "In comparing two trajectories, one can consider space and time as either absolute (i.e. compared with an external spatial and/or temporal reference frame) or relative (i.e. intrinsic comparison, ignoring absolute times or positions)."

For the ball and plate, this means that, in evaluating a control system response, we may consider a TSM that allows for phase lag in time, absolute offset in plate location, or even rotations of the trajectory shape within the frame of the plate. It is difficult to imagine some of those scenarios being a reasonable response for our controllers to achieve. For instance, if the setpoint is to draw a four-leaf clover centered at the origin, it is not likely that the controller draw a perfect four-leaf clover out in the corner of the plate. Nevertheless, these types of scenarios may, for some given TSM return a relatively small measure. Other scenarios, however, may be perfectly reasonable. For instance, if the control objective is to track a perfectly centered circle of a certain radius and angular frequency, the response of the controller may be a centered, phase lagged circle with a bit of amplitude attenuation. For some TSMs, it may be appropriate to shift the response data in time such that the phase lag is eradicated and then the TSM will measure the effect of the amplitude difference.

With respect to relative and absolute time, TSMs come in three categories (Tao et al. [9]):

Table 1.2: TSM Timing Categories

1)	Rigid	Does not support relative time
2)	Flexible	Evaluates spatial similarity ignoring time shifts
3)	Semi-flexible	Evaluates spatial as well as temporal similarity

The authors give intuitive commentary on the differences between the categories: "A pair of trajectories that are spatially identical but vary in speed profile along the trajectory will be expected to have a higher similarity score when compared using a flexible measure than a rigid or semi-flexible measure."

The TSMs explored by Tao et al. are demonstrated in the Fig. 1.6

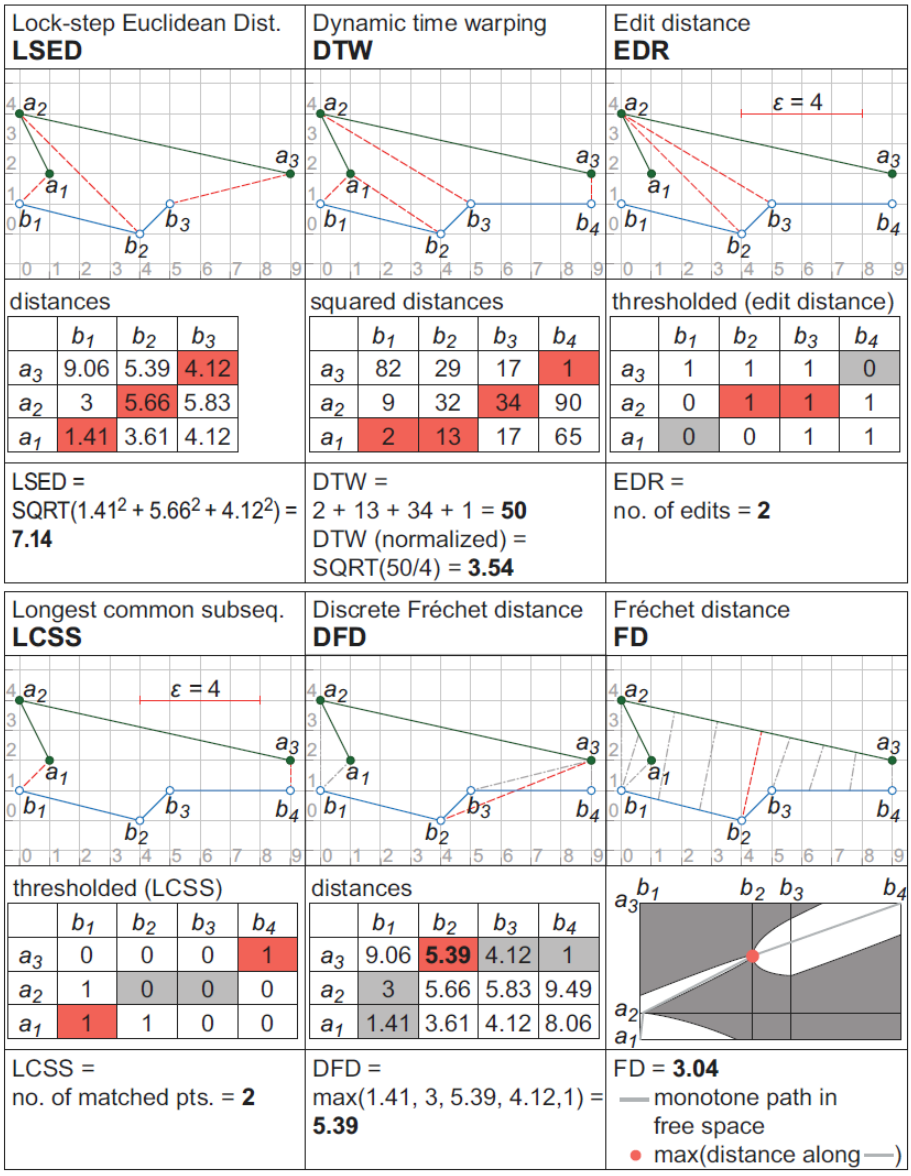


Figure 1.6: TSMs Demonstration. Tao et al. [9, p.649]

"Demonstration of trajectory similarity measures, aligning two trajectories where n=3 and m=4 (except for LSED, where n=m=3) according to the various measures, along with a corresponding distance matrix or free-space diagram." Tao et al. [9, Figure 1]

These examples use trajectories in 2D Euclidean space - the gridded plots represent the space that the compared trajectories (blue and green) live in. But, these measures can be easily extended to higher dimensions. For many of these measures, discrete points in the trajectory are used, and therefore a high enough sample rate to capture

the trajectory's features is necessary. The discrete nature allows the data to be collected in finite distance matrices containing the values used in computing the measure. As an example, for the Lock-Step Euclidean Distance (LSED), the values inside the distance matrices are the Euclidean distances $[dist_2(\mathbf{a}_i, \mathbf{b}_i)]$ between time aligned points in the trajectory, 1.25. The LSED measure between trajectories A and B is then

$$Eu(A, B) = \sqrt{\sum_{i=1}^{n} dist_2^2(\mathbf{a}_i, \mathbf{b}_i)}. \tag{1.25}$$

Because the points must be aligned in absolute time, this measure is a rigid TSM. It is worth noting that the LSED measure will increase with the sample rate and with the duration of time under consideration. There are however variations that do not have this property.

An example of a flexible TSM surveyed by the authors is Dynamic Time Warping (DTW), originally used for speech recognition [8]. Tao et al. state:

> Dynamic time warping can be seen as selecting a minimum cost path in the distance matrix. More precisely, DTW selects a path from the lower left corner to the upper right corner of the distance matrix that minimizes the sum of the squared distances.

This means that DTW locally distorts the time a particular change in either trajectory takes in order to minimize an LSED-like distance measure. This must be done in a manner that keeps the time values associated with points in the trajectories monotonically increasing. DTW essentially applies as many zero-order holds to points in

[8] Imagine your phone needing to understand that you spoke the words "take me to home" regardless of the pace at which you spoke them.

either trajectory as is necessary to make them as close in measure as possible. This can be done after a global time shift or scaling is applied to one of the trajectories - for instance, to make two periodic trajectories have the same number of cycles. The formal definition will not be included here, but it should be noted that this is an optimization problem where dynamic programming is applied.

1.2.3 Underactuation

The ball-and-plate system is underactuated, and as a consequence, well understood and effective nonlinear trajectory-tracking methods will not work - at least not perfectly. In a course on underactuated robotics at MIT, Tedrake [10], gives an overarching theme in the discussion of underactuation:

> Classical control techniques for robotics are based on the idea that feedback can be used to override the dynamics of our machines. These examples suggest that to achieve outstanding dynamic performance (efficiency, agility, and robustness) from our robots, we need to understand how to design control system which take advantage of the dynamics, not cancel them out.

Tedrake speaks of utilizing the natural dynamics of a system to achieve a control objective. A control method that's formulated to take advantage of the natural dynamics of the ball-and-plate system[9] is discussed in Section 1.2.4. Tedrake also speaks of cancelling out dynamics to achieve control objectives - referring to techniques like feedback linearization. A common nonlinear control technique, sliding mode control, is very similar to feedback linearization and also works by cancelling out the system

[9]The law is argues in this thesis to have limited effectiveness.

dynamics. Though they will not work perfectly, similar techniques can be made to work with compromise. For example, an output sliding mode controller is explored in Section 3.2 where the output variable is chosen to be something other than the x or y state.

Tedrake describes underactuation as such:

> In words, underactuated control systems are those in which the control input cannot accelerate the state of the robot in arbitrary directions. As a consequence, unlike fully-actuated systems, underactuated system cannot be commanded to follow arbitrary trajectories.

Tedrake also gives a formal definition for underactuated mechanical systems. These second order systems can be represented in the form:

$$\ddot{\mathbf{q}} \;=\; \mathbf{f_1}(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{f_2}(\mathbf{q}, \dot{\mathbf{q}}, t)\mathbf{u} \tag{1.26}$$

where, for our ball-and-plate system, $\mathbf{q}$ is a vector containing the degrees of freedom. For the decoupled dynamics, system 1, the degrees of freedom are given by:

$$\mathbf{q} \;=\; \begin{bmatrix} x \\ \beta \end{bmatrix} \tag{1.27}$$

The system is then underactuated if (1.28) is satisfied.

$$rank\left[\mathbf{f_2}(\mathbf{q}, \dot{\mathbf{q}}, t)\right] < dim\left[\mathbf{q}\right] \tag{1.28}$$

Intuitively, this means that the system has fewer linearly independent actuators than the number of relevant degrees of freedom. Equation (1.28) makes it clear that the rank of the input matrix is dependant upon the state of the system. For the ball-and-plate system, since (1.17) and (1.18) are vectors, the max rank possible is one. Even so, it is conceivable that for some states, $\mathbf{f_2}$ maps $\mathbf{u}$ to the zero vector, causing the system to lose actuation entirely. This is not the case for (1.17) or (1.18) since the state-dependant terms are always positive and the remaining terms are constant.

Tedrake also tells us that the system can become underactuated if additional constraints such as $|\mathbf{u}| \leq T_{max}$ are imposed. Input saturation turns out to be an important constraint that limits the aggressiveness of trajectories achievable by the ball and plate. In many simulations conducted while the controllers were being tuned, the input would saturate and immediately cause the response to go unstable. A simple thought experiment can be construed that demonstrates one way input saturation can be an issue for the ball and plate. Consider the "teeter-totter" model for system 1, in the configuration shown in Fig. 1.7. With the plate tilted in the positive β direction, if the ball is located as it is in the figure, and rolling downhill (in the x direction), as the ball rolls, it gets further and further from the center of the plate. In the absence of actuation, the plate will continue to tilt and the ball will continue to roll downhill.

If the goal is to get the ball to move to the left of the plate's center, it is not difficult to imagine that, at some point, the plate would need to be tilted the other way. If there is a limit on how much torque can be applied to rotate the plate in the negative β direction, the torque may not be enough to counter the influence of gravity, the momentum of the ball, etc. What's occurring is, the influence of the state on the control objective is overpowering the influence of the input. In this case, the issue

is not that that the actuation is in an unfavorable direction, it's that it is not large enough in magnitude to actually tend the states toward the desired direction[10].

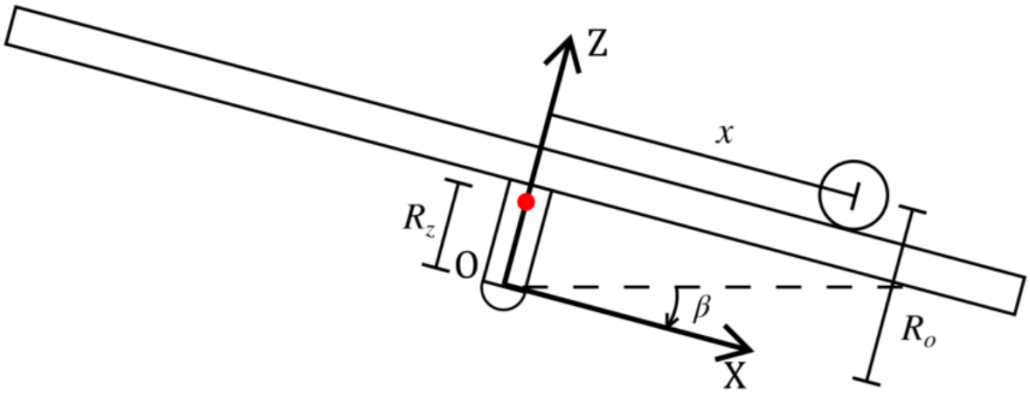


Figure 1.7: System 1 Schematic Richter [6, p.8]

As will be discussed in Chapter 2, the primary control objective is to track a desired time-dependant ball trajectory in the plate frame coordinates, x and y. Since we are not requiring that β and γ follow some time-dependant trajectory as well, i.e. we are not asking for the system to track arbitrary trajectories in the state-space, the control objectives are not in conflict with the physics. It is then conceivable for freedom in the angular trajectories to facilitate perfect tracking in the position trajectories. The recommended controller design, full-state feedback with feed forward , finds angular trajectories consistent with the linear representation of the dynamics, given desired x and y trajectories that meet certain requirements. A feed forward torque is then found that actuates this (approximately) consistent set of positional and angular trajectories.

If the system were fully actuated though, this method would not be necessary. There would be no need to find consistent angular trajectories because all angular trajectories would be consistent. It would be trivial to invert $\mathbf{f_2}$ in (1.26) - it would be

[10]Here, "direction" means "direction in the state space". For an underactuated system, the state vector's derivative cannot be rotated into an arbitrary direction, but if one has a desired direction that the input can be rotated to, the magnitude of its influence is limited by the saturation.

full rank and square - and isolate $u = T_\beta$. We could then satisfy the dynamics of trajectories in x and β separately by specifying $\mathbf{q}$ and computing its derivatives:

$$\mathbf{u} \;=\; \mathbf{f_2}(\mathbf{q}, \dot{\mathbf{q}}, t)^{-1} \left(\ddot{\mathbf{q}} - \mathbf{f_1}(\mathbf{q}, \dot{\mathbf{q}}, t)\right). \tag{1.29}$$

If $\mathbf{u}$ is enacted while the system has some state $\mathbf{q}, \dot{\mathbf{q}}$ that lies on the trajectory at time t, then the system will remain on the trajectory (in the idealized system). In an imperfect world, however, a control system would then be wrapped around the error states to reject disturbances, model uncertainty, noise, offset initial conditions, etc. Tedrake [10] replaces $\ddot{\mathbf{q}}$ in (1.29) with a new input $\mathbf{u}'$:

$$\mathbf{u} \;=\; \mathbf{f_2}(\mathbf{q}, \dot{\mathbf{q}}, t)^{-1} \left(\mathbf{u}' - \mathbf{f_1}(\mathbf{q}, \dot{\mathbf{q}}, t)\right). \tag{1.30}$$

When this actuation, $\mathbf{u}$ is plugged into (1.26), a fully actuated linear system (1.31) results.

$$\ddot{\mathbf{q}} \;=\; \mathbf{u}' \tag{1.31}$$

Here, $\mathbf{q}$ and $\dot{\mathbf{q}}$ are now measurements of the state (rather than desired trajectories) and $\mathbf{u}$ is the feedback linearizing input. The dynamics of the system can then be set,

through choice of $\mathbf{u}'$, to be some stable linear dynamics. If the definition of $\mathbf{q}$ (1.27) were in terms of error states,

$$\mathbf{q} = \begin{bmatrix} e_x \\ e_\beta \end{bmatrix}, \tag{1.32}$$

then the decaying linear dynamics produced would be chosen to asymptotically bring the error states to zero. This approach is subject to uncertainty since the feedback linearizing input does not have a perfect model of the system. Tedrake points out that there are "results which generalize this idea to the case where $\mathbf{f_1}$ and $\mathbf{f_2}$ are estimated, rather than known". One such means of handling the model uncertainty is sliding mode control (Slotline and Li [8, Ch 7]).

For the ball-and-plate system, full actuation could be achieved by attaching an actuator to the ball and allowing it to be independently driven along the plate. Then the trajectory in β could just be to maintain a value of zero and the ball would be driven around on a horizontal plate to achieve the control objective. This is a far less interesting scenario.

1.2.4 Ball-and-Plate-Specific Trajectory Tracking Schemes

There are plenty of examples of control systems for the ball and plate, or ball-and-beam systems in the literature. Many of the reviewed methods take advantage of simplifying assumptions that, if reasonable, render the system fully actuated and/or feedback linearizable. It is found that the controllers from the literature, generally speaking, start from decoupled nonlinear models, but additionally require that the system behave quasi-statically. For instance, Mohammadi and Ryu [4] begin from a nonlinear decoupled model and they assume, in similar equations to (1.11) and (1.12),

that the $\ddot{\theta}$ and $\ddot{\gamma}$ terms, the angular accelerations of the plate, are zero. The authors then develop a feedback linearizable system that follows from this, and related, assumptions. Ramírez-Neria et al. [5] start from similar equations and make the same $\ddot{\theta}$ and $\ddot{\gamma}$ assumption - though these authors use Jacobian linearization rather than attempt feedback linearization. These papers retain the motor as the input to the ball dynamics.

Other methods utilize a cascaded control approach while maintaining the quasi-static assumption. Liu and Liang [2] present a nonlinear, cascaded control architecture that utilizes a Lyapunov stability argument in its proof of asymptotic tracking. Jeon and Hyun [1] use similar techniques and derive the same control law - though these authors mention the use of a boundary layer to reduce chattering. It is shown below that this boundary layer is actually crucial for practical application as a chattering angular input is inconsistent with the necessary assumptions for the law. For these methods, if a switching law could somehow be implemented (without a boundary layer), the dynamics of the input are guaranteed to be significant because it would have impulsive derivatives and we wouldn't be able to consider them negligible.

Liu and Liang present a decoupled model of the ball-and-plate system. In Section 1.1.2, the ball moment equations, (1.11) and (1.12), relate the positional dynamics to the angular dynamics. Liu and Liang show these relationships, but frame some of the terms as an unknown torque, T_x. The $sin(\beta)$ term is retained and considered the input to the system.

For system 1 variables, this results in

$$\begin{aligned} \ddot{x} &= \frac{5}{7}\, g\, sin(\beta) + \frac{5}{7\, m_b\, r_b} T_x \\ \ddot{x} &= b\, u_x + \Delta f(t) \end{aligned} \tag{1.33}$$

where T_x is the unknown torque on the ball, $u_x = sin(\beta)$, and $b = \frac{5}{7}\, g$. Here, u_x is considered the input to the positional dynamics for the ball in a fully-actuated system. The term $\Delta f(t)$ is the unknown torque described as a "time-varying uncertainty". If this term were zero, the input would directly determine the acceleration of the ball, and creating the desired dynamics would be straightforward.

In the theme of Tedrake (Section 1.2.3 on underactuation), these control laws take advantage of the natural dynamics of the system. They use the fact that the ball has a natural tendency to roll down an angled plate and are essentially using gravity as the actuator for the ball.

According to the authors, the unknown torque in (1.33) comes from un-modelled friction. It is apparent that this term is not solely from un-modelled friction, but also from the normal force at the contact point of the ball. Regardless, it can be seen from (1.11) and (1.12) that this term is actually where the β dynamics enter the expression:

$$\Delta f(t) = \frac{5}{7} \left(-\ddot{\beta} \left(z_b + \frac{2\, r_b}{5} \right) + \dot{\beta}^2\, x \right). \tag{1.34}$$

The fact that this expression contains hidden information about the plate dynamics is why the plate could never truly serve as a switching input. The $\Delta f(t)$ term itself

contains the dynamics of the input ($sin(\beta)$) so instantaneously altering the input violently excites the ball.

Liu and Liang derive a sliding mode control law using the sliding surface

$$s = \dot{e}_x + \lambda e_x \tag{1.35}$$

where

$$e_x = x - x_s \tag{1.36}$$

and x_s is the desired x setpoint. In a common manner for sliding mode control, a Lyapunov function is chosen to be (1.37),

$$V = \frac{1}{2}s^2 \tag{1.37}$$

and its derivative is (1.38).

$$\dot{V} = s\dot{s} \tag{1.38}$$

Here, $\dot{s}$ brings in the dynamics of the system and a control law follows that intends to keep $\dot{V}$ negative definite.

$$\begin{aligned} \dot{s} &= \ddot{e}_x + \lambda \dot{e}_x \\ \dot{s} &= \ddot{x} - \ddot{x}_s + \lambda \dot{e}_x \\ \dot{s} &= (\ b\, u_x + \Delta f(t)\ ) - \ddot{x}_s + \lambda \dot{e}_x \end{aligned} \tag{1.39}$$

The control law given by the authors is (1.40),

$$u_x = b^{-1} (\ \ddot{x}_s - \lambda \dot{e}_x - c_1\, s - c_2\, sgn(s)\ ) \tag{1.40}$$

where c_1 and c_2 are constant controller gains. With sliding mode control, there is typically an additional term to cancel out the nonlinear dynamics and produce a linear response in the error states. Since this control scheme lumps the nonlinear dynamics in with the time varying uncertainty $\Delta f(t)$, this term is omitted. With this control law, the derivative of (1.37) is the following:

$$\dot{V} = -c_1\, s^2 - c_2\, |s| + \Delta f(t)\, s. \tag{1.41}$$

If $\dot{V}$ here can be guaranteed negative definite, then asymptotic tracking of the reference signal is proven. However, (1.34) clearly shows that $\Delta f(t)$ and u_x are related through the dynamics of β and therefore the assumed input must remain relatively calm if the $\Delta f(t)$ term is to be kept sufficiently small. Though not explicitly stated in the papers, this control would need to be achieved using an inner loop with sufficiently faster dynamics compared to the outer positional loop. Because u_x must be

quasi-static, a boundary layer is needed to prevent the inner loop control objective from constantly switching.

Ultimately, these angular input sliding mode papers assume an instantaneously achievable angular setpoint can serve as the input to the x and y dynamics. With the necessary addition of a boundary layer, the law degrades to a simple linear cascaded control architecture[11]. Cascaded control and full-state feedback are effectively equivalent for a linear system, but for a nonlinear system, one may have benefits over the other. Rather than explore this distinction, the cascaded controller will be tabled, and only the full-state feedback control scheme will be explored in Chapter 3.

As discussed in the underactuation section, 1.2.3, the classic techniques of nonlinear control cannot be utilized perfectly with the ball-and-plate system. The above examples are cases where they can work, to a degree. For this thesis though, we desire aggressive trajectories (discussed in Section 2.3). Therefore the quasi-static assumption will not hold well. The final controller recommended in Chapter 5 reasons about the dynamics without assumptions that frame the system as fully actuated.

[11]Where the inner angular control loop has a saturating setpoint.

Chapter 2

CONTROL CRITERIA

In this chapter, control criteria for the ball and plate are presented, often referring to topics covered in Section 1.2. Asymptotic tracking is expanded upon for our problem, an appropriate TSM is chosen, qualitative criteria are introduced, and some practical considerations are shown before getting into the controller designs of Chapter 3 .

2.1 Defining the Tracking Problem for the Ball and Plate

For the ball and plate, we are interested in the output equation:

$$\mathbf{y} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.1}$$

For the decoupled systems, there are two separate output equations,

$$\mathbf{y}_1 = \mathbf{x} \tag{2.2}$$

$$\mathbf{y}_2 = \mathbf{y}, \tag{2.3}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the vector representations of state variables x and y. These choices allow one to place some printed trajectory shape on the plate so that spectators can

easily see visual feedback on the tracking performance. Or perhaps, the plate could be fitted with an LCD touchscreen and the setpoint could be displayed in real time. This type of experience would likely be much more difficult for different output variables. In Section 3.2, the global x and y locations of the ball are used as output variables in deriving the control law, but the performance is judged on the behavior in the corresponding plate frame variable. [1]

We constrain ourselves to analytical, pre-planned output trajectories whose derivatives exist and are known a priori. An additional constraint is necessary for the feed-forward controller of Section 3.3. There we require that we can find an analytical solution to a linear ordinary differential equation (ODE) whose forcing function is the second derivative of the desired trajectory (3.49). These constraints are to keep the project within a reasonable scope. One may wish to design a control system with user interaction, where a user alters the setpoint, perhaps by means of a joystick. One can also envision organically drawn setpoints where the user instructs the controller by drawing a shape on a tablet. The shape could then be parameterized by time. These trajectories would likely be lookup tables rather than functions with infinite precision, and their derivatives would need to be arrived at through some non-trivial process, such as a spline or numerical differentiation. Though interesting, these scenarios are not considered.

The broadly defined trajectory-tracking problem presented in Section 1.2.1 is considered the ideal scenario for any control law explored in this thesis. It is found that asymptotic tracking is a very difficult objective to meet for this underactuated system. The equations of motion are too cumbersome and complicated, and the input matrix for the full nonlinear system cannot be inverted. Ultimately, none of the control

[1] In other words, even though the feedback law uses feedback on an output that differs from ultimate control goal, the goal remains unchanged. The tracking performance will be evaluated in the same way as other controllers in this thesis.

systems presented in this thesis can achieve perfect asymptotic tracking. None have an associated proof in the context of the nonlinear models - though the feed-forward controller does have a proof in the linear model. Even if a control law that proves asymptotic tracking were found, the model of the system is imperfect. The nonlinear model ignores damping, noise, and, potentially, many more factors that contribute to uncertainty. Therefore it is necessary to loosen the requirement and judge the controllers on different criteria.

We are interested in controllers that get us as close as possible to asymptotic tracking, and we give a very high weighting to timely performance. We want the aggressiveness of the desired trajectory to be reflected in the actual trajectory, and we are not satisfied that the trajectory merely have the correct shape. These qualitative criteria lead us to consider the trajectory similarity measures of Section 1.2.2, and in Section 2.2, they guide the choice of TSM by which tracking performance is gauged.

2.2 Selecting an Appropriate Trajectory Similarity Measure

In Section 1.2.2, Table 1.2, the timing categories described by Tao et al. are given. Because we are interested in timely performance, we need a TSM restricted to absolute time - i.e. we would like a rigid measure. Furthermore, we are not interested in trajectories whose relative location is inconsequential. It isn't acceptable for the control system to draw a square in the corner of the plate if a centered square is desired. The Lock-Step Euclidean Distance (LSED) TSM, (1.25), is the measure given by Tao et al. that meets these criteria.

This measure, however, isn't normalized and so its magnitude may depend on the amount of time a controller runs for [2]. Tao et al. give a normalized version of the LSED - this is the TSM that meets the criteria given above in Section 2.1. This modified version of the LSED that gives "the average distance between corresponding measurements" and is shown in (2.4) [3], Tao et al. [9]:

$$Eu'(A, B) = \frac{1}{n} \sum_{i=1}^{n} dist_2(a_i, b_i) \tag{2.4}$$

2.3 Additional Control Criteria

It is desirable to have highly aggressive trajectories. These types of trajectories are impressive to watch and more difficult to achieve. For slow trajectories, many of the controllers introduced in Chapter 3 have relatively good performance. To push the limits of the ball-and-plate hardware, and to find a control architecture distinguished from those that make limiting assumptions (like a quasi-static plate), faster trajectories are held in high regard.

Step and regulation control responses will be considered particularly important. If the controller cannot produce these, it is unlikely to be recommended. Good step and regulation responses are fundamental and expected out of a controller.

The final criteria is that the controller have qualitatively pleasing behavior, meaning that overly jolty ball-and-plate motion is penalized. Though this is certainly subjective, it was determined to be an important element to include given that the purpose

[2]Say we are comparing a control architecture's ability to handle circular setpoints at various frequencies. We would like to be able to compare the TSM between frequencies regardless of how long we must run the controller for to capture the full shape.

[3]Remember that $dist_2(a_i, b_i)$ is the Euclidean distance between points a_i and b_i

of the design is to be an enjoyable and inspiring display piece. In Chapter 5, this criterion is used to discuss control architectures presented in Chapter 3 and to support a final design recommendation. To judge controllers by this criterion, animations were produced in MATLAB® utilizing 3D images of the ball-and-plate system. A snapshot of one such animation is shown in Fig. 2.1. The animations gave a sense for what bystanders of the ball-and-plate display will experience. These animations were viewed throughout the project, for each controller, even for trajectories not included in this thesis. Discussion regarding qualitatively pleasing behavior is based on a multitude of these animated responses.

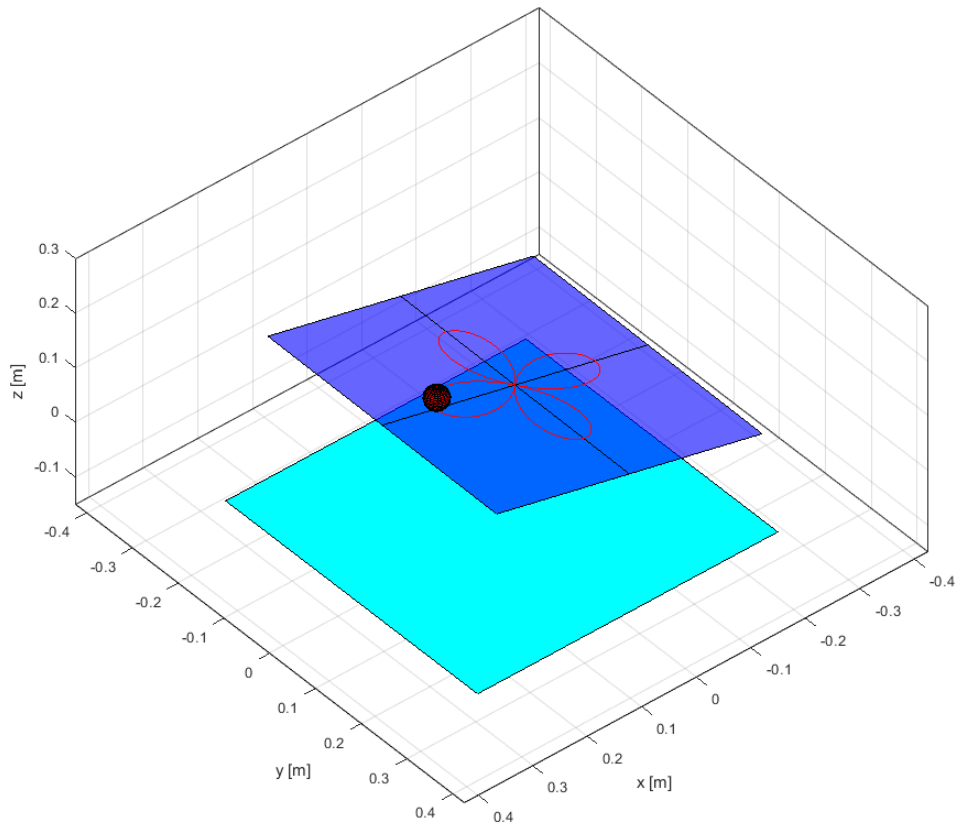


Figure 2.1: 3D Animation Example

The response of the controller to noise and disturbances is important to consider for real world control systems. However, the thesis focuses on finding a control architecture that is theoretically suitable for the system. It seeks to find a good foundation on which to begin that kind of practical work.

2.4 Practical Considerations and Constraints

Richter [6] considers the input to the system to be the torques about the u-joint axles. However, he also notes that the true input torques are supplied by motors attached to the plate through a crank arm and connecting rod. This implementation is depicted in Fig. 2.2.

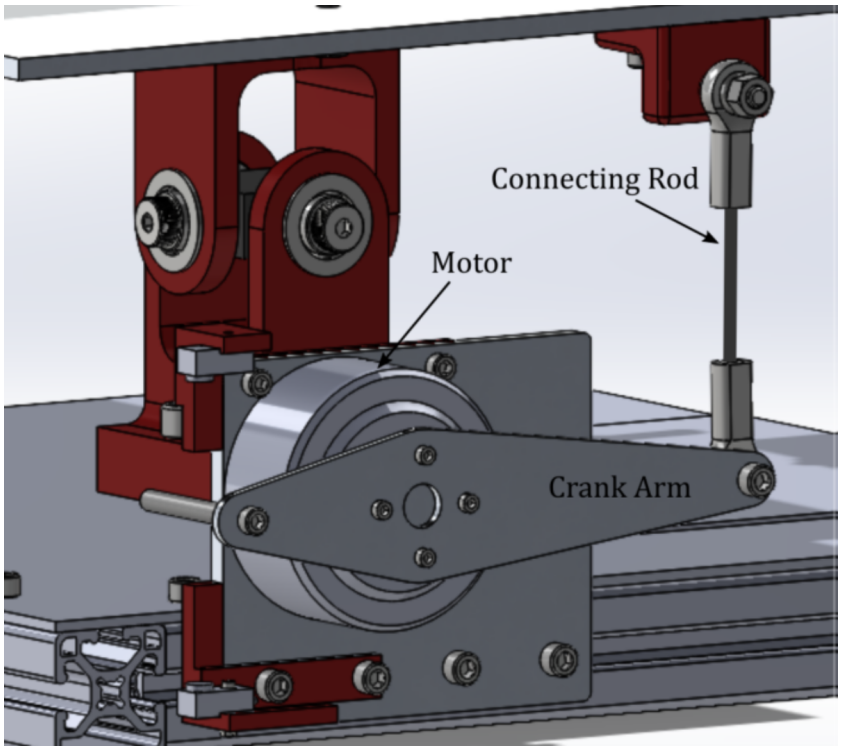


Figure 2.2: Motor Actuation Mechanism Richter [6, p.16]

Richter acknowledges that there is additional complexity if one chooses to consider the way in which a torque is truly applied to the u-joint axles. We bring in a fraction of that complexity by considering the crank arm and connecting rod to be a gear ratio between the applied motor torque and the effective u-joint torque. We assume that the crank arm, connecting rod, and motor armature are massless. We also assume that the crank arm and connecting rod are nominally at 90° when the platform is perfectly level. From there, a small angle approximation is employed and the resulting

expression is that of a constant gear ratio between the motor and the associated u-joint angle. This gear ratio is computed from the length of the crank arm (about 0.113 $[m]$), and half the width of the plate (about 0.25 $[m]$), and takes on a value of about 2.21. In combination with the maximum continuous torque of the motors, 0.6 $[Nm]$, the maximum u-joint axle torque is found to be 1.33 $[Nm]$. This torque value is used in simulation and saturates the input to the plant.

Chapter 3

CONTROL ARCHITECTURES AND SIMULATION RESULTS

Several control architectures are explored in this chapter, and simulation results for each are presented.

1. Full-State Feedback (FSFB) with Integral Action
2. Single-Input-Single-Output Sliding Mode
3. Full-State Feedback with Feed Forward

This chapter also gives a motivation for each architecture, details them mathematically, and discusses simulation results that support the final recommendation in Chapter 5. For each architecture, several types of trajectories are simulated, namely the step, regulation, and the quadrifolium rose curve responses. The rose curve is given by (3.1) and (3.2), where r is the radius of the rose curve (the furthest x or y will be from zero), n and d select a particular rose curve, and ω is the angular frequency. The angular frequency is chosen so that the overall shape evolves over a certain period, T_r - the smaller the period, the more aggressive the trajectory. For the quadrifolium, $n = 2$, and $d = 1$. Figure 3.1 shows that the curve is similar to a four-leaf clover.

$$x(t) = \frac{r}{2}[cos((\frac{n}{d} + 1)\omega t) + sin((\frac{n}{d} - 1)\omega t)] \tag{3.1}$$

$$y(t) = \frac{r}{2}[cos((\frac{n}{d} + 1)\omega t) - sin((\frac{n}{d} - 1)\omega t)] \tag{3.2}$$

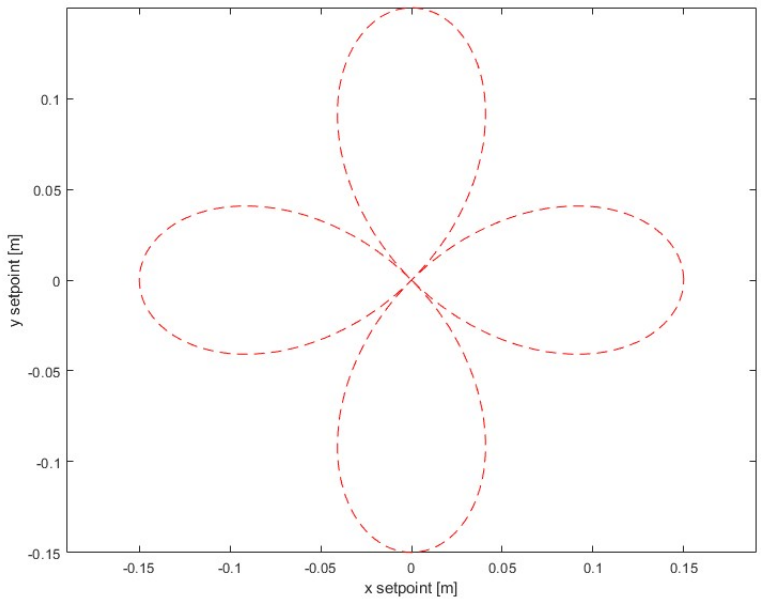


Figure 3.1: Quadrifolim Rose Curve with $r = 0.15\ [m]$

The responses are simulated from offset initial conditions at $x_0 = 0.075\ [m]$ and $y_0 = 0.075\ [m]$; half way to the edge of the plate in the first quadrant. The initial plate angles and angular velocities, as well as ball velocities were set to zero. Tuning each controller was done through manual iteration, input saturation being the most common and difficult obstacle to overcome. For the tunings that produced reasonable behavior, large initial errors would cause the torques to saturate and the system would go unstable. To combat this initial error, the setpoints are eased onto the desired trajectory through an exponential smoothing function, (3.3), applied to the initial condition vector. Here, $\bar{\mathbf{x}}_\mathbf{s}$ is the desired setpoint vector, $\mathbf{x}_\mathbf{s}$ is the smoothed setpoint

vector, and τ is the time constant. The smoothed setpoint function shape overlaid with the original is given in Fig. (3.2).

$$\mathbf{x}_s \;=\; \bar{\mathbf{x}}_\mathbf{s}(1 - e^{-\frac{t}{\tau}}) \;+\; \mathbf{x_0}(e^{-\frac{t}{\tau}}) \tag{3.3}$$

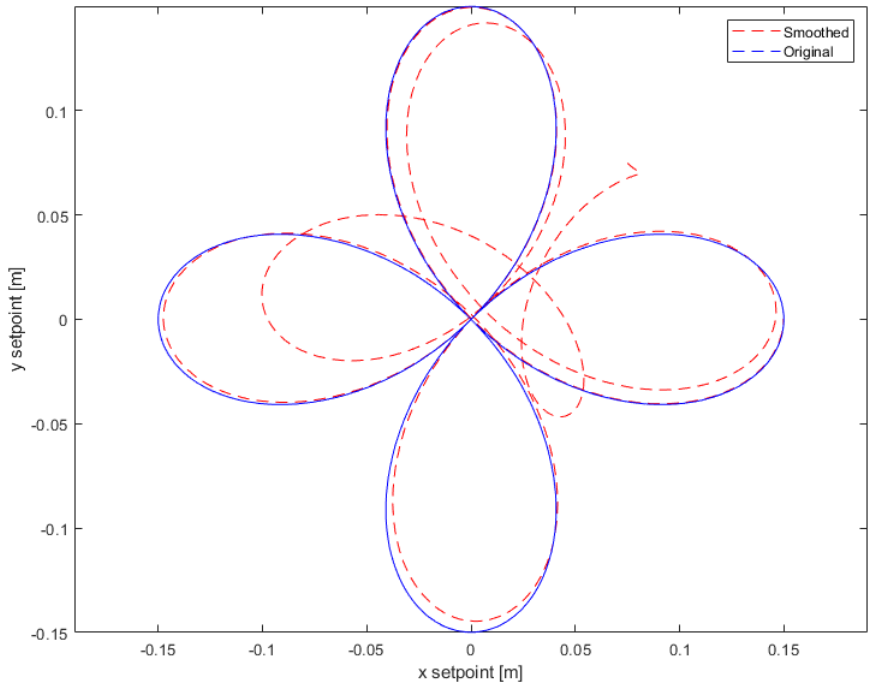


Figure 3.2: Quadrifolim Rose Curve with $r \;=\; 0.15\ [m]$

It is clear that at time zero, the setpoint is the initial conditions, and at time infinity, it is the desired setpoint. The time constant of this process was a parameter to tune and varied between controllers. This solution saves the feedback elements of the controllers from the initial jolt that results from offset initial conditions. For disturbance rejection, a similar process might be utilized that smooths the disturbed

state of the ball back onto the desired trajectory. This scenario is not investigated in this thesis and is left to future work.

For all control laws presented, the error terms are defined as follows:

Table 3.1: Error State Definitions

Error Term	Definition
e_x	$x_s - x$
e_y	$y_s - y$
e_β	$\beta_s - \beta$
e_γ	$\gamma_s - \gamma$
e_{y_1}	$y_{1s} - y_1$
e_{y_2}	$y_{2s} - y_2$

Here, x_s, y_s, β_s, γ_s, y_{1s}, y_{2s} and are the setpoint variables - the desired trajectories in the associated state or output variables. The variables y_{1s} and y_{2s} are the setpoints associated with output variables used in the SISO sliding mode controller of Section 3.2.1 - they are the global x and y position of the ball and are defined in that section.

Optimal tuning over a range of setpoint functions was not conducted as this would be very challenging and is out of scope for this thesis. However, the tunings found were enough to introduce the advantages and limitations of the architectures. In Chapter 4, normalized LSED studies are shown with the same gains presented here. In that chapter, the controller behavior shown here is given additional supporting evidence using a wide range of trajectories. In Chapter 5, the FSFB with feed forward controller is recommended. The controller has a proof of asymptotic tracking in the linear model, has excellent performance in simulations that use a nonlinear plant model, and produces the most pleasing visual experience when viewed in animation.

3.1 Linear Full-State Feedback with Integral Action

The Full-State Feedback with Integral Action controller serves as a kind of baseline for comparison. It is similar to the PID controller - the standard method that will work well for many applications. For this system, however, a simple PID scheme relating the u-joint torques to the ball position would prove challenging. A PID with left-hand plane poles (and no input saturation, noise, etc) would have a proof of bounded input-bounded output, BIBO, stability. But there may still be internal states for which stable output transfer function poles cannot guarantee stability. For a PID in our system, it may be possible to choose gains that produce stable poles in the transfer function relating x to x_s while also providing for stable angular states. Ultimately, one would need to place the eigenvalues of the entire state coupling matrix using only feedback on x - i.e., without feedback on the angular states[1]. Full-state feedback eases this burden. In this method, error feedback on the ball states is used alongside absolute feedback on the angular states to tend the system toward the desired trajectory while maintaining stability in all the states[2].

It is important to note that stability and asymptotic tracking are two very different things. Imperfect tracking with this method manifests as attenuation/amplification of frequency components of a signal, and inherent phase lag. These features are necessary for the scheme to work at all, as error and integral action determine the

[1]One can choose PID gains (or some other partial-state feedback approach) and analyze where that feedback places the poles of the state-transition matrix in the equivalent state space representation of the system. If those poles all happen to be in the left-hand plane, then the angular states will also be stable. If the poles associated with the angular states are not in the left-hand plane, then, from the perspective of the PID for x, there are unstable internal states that are unaccounted for.

[2]Simon [7] describes the difference between BIBO stability of a linear transfer function and overall stability of the system in the state space representation. A transfer function may have stable poles, but unstable, unobservable states may be present in the complete representation. In such a scenario, not all eigenvalues of the closed-loop state coupling matrix are in the left-hand plane.

applied torque. For non-constant setpoints, the error integrator cannot supply the ever-changing signal required to keep the ball on the trajectory. These issues are characteristic of PID-like trajectory-tracking control schemes.

3.1.1 Control Law

The controller utilizes the linear, decoupled model of Section 1.1.3 in its derivation. Error integrators on x and y are included to handle constant setpoint situations. They provide the steady state torque needed to balance the ball away from the equilibrium point. To bring in the integral action, an augmented state vector for each system is defined and used to modify the state equations presented there. For system 1, the augmented state vector is (3.4)

$$\mathbf{x}_{1a} = \begin{bmatrix} \int_0^t e_x \, \mathrm{d}t \\ e_x \\ \dot{e}_x \\ \beta \\ \dot{\beta} \end{bmatrix} \tag{3.4}$$

A setpoint vector, $\mathbf{x}_{1s}$, must be introduced to describe the dynamics of the error states[3]:

$$\mathbf{x}_{1s} = \begin{bmatrix} x_s \\ \ddot{x}_s \end{bmatrix}. \tag{3.5}$$

[3]Here, $\dot{x}_s$ is omitted because $\dot{x}$ does not show up in equations for $\ddot{x}$ and $\ddot{\beta}$. However, $\dot{x}_s$ is needed to compute $\dot{e}_x$.

The open loop dynamics, (3.6), are found by differentiating $\mathbf{x}_{1a}$. Here, A_{1a} and B_{1a}, are the augmented A and B matrices closely related to the original matrices.

$$\dot{\mathbf{x}}_{1a} = A_{1a}\mathbf{x}_{1a} + B_{1a}T_\beta + S_1\mathbf{x}_{1s} \tag{3.6}$$

$$A_{1a} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -1.8786 & 0 & -4.1026 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & -20.2622 & 0 & 31.3281 & 0 \end{bmatrix} \tag{3.7}$$

$$B_{1a} = \begin{bmatrix} 0 \\ 0 \\ 0.8511 \\ 0 \\ 9.1798 \end{bmatrix} \tag{3.8}$$

The setpoint vector enters the error dynamics through the setpoint matrix S_1.

$$S_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1.8786 & 1 \\ 0 & 0 \\ 20.2622 & 0 \end{bmatrix} \tag{3.9}$$

Unlike the absolute states, the error states are influenced both by the torque and the desired trajectory. The augmented matrices have additional rows/columns due to the addition of the integral state. There are also sign differences from the original dynamics that arise from the error definitions in Table 3.1.

Because the system is fully controllable (Richter [6]) a full-state feedback law can be introduced to place the poles of the system in the left half plane. This is shown in (3.10). The control law stabilizes the system and tends it toward the trajectory by actuating against the error.

$$T_\beta = -K_1 \mathbf{x}_{1a} \tag{3.10}$$

The law results in a closed-loop system whose input is the setpoint vector:

$$\dot{\mathbf{x}}_{1a} \;=\; [A_{1a} - B_{1a}K_1]\mathbf{x}_{1a} + S_1 \mathbf{x}_{1s} \tag{3.11}$$

It is clear from (3.10) that if the error in x is zero, the torque is proportional to the integral of the error, and the absolute angular states. Unless the time evolution of these terms is precisely what is required to keep the system on the trajectory, the system will not stay at zero error. For a constant x_s, and implied angular setpoint of zero[4], the law can work perfectly. The torque will be proportional to the integrated error alone, and the term will find the torque needed to hold the plate horizontal and the ball in place at x_s. This is the only scenario where perfect tracking can theoretically be achieved.

[4]There is an implied angular setpoint of zero when absolute β variables are used in the feedback law. If error states were used for instead, a setpoint of zero would produce the same term in the feedback law (with a sign change that is accounted for in the pole placement).

3.1.2 Simulation Results

The control law is implemented in Simulink®. The outer loop is shown in Fig. 3.3. Here, the setpoints are generated in the MATLAB® symbolic toolbox and brought into Simulink® as MATLAB® function blocks - they are contained in the "Setpoint Vectors" subsystem. The setpoint smoothing is also implemented in this subsystem; it contains MATLAB® function blocks that act on the raw setpoint signals. Each axis is controlled separately, in correspondence with the linear decoupling assumptions of Section 1.1.3. For system 1, the controller is contained in the "Controller x dimension" subsystem, as shown in Fig. 3.4.

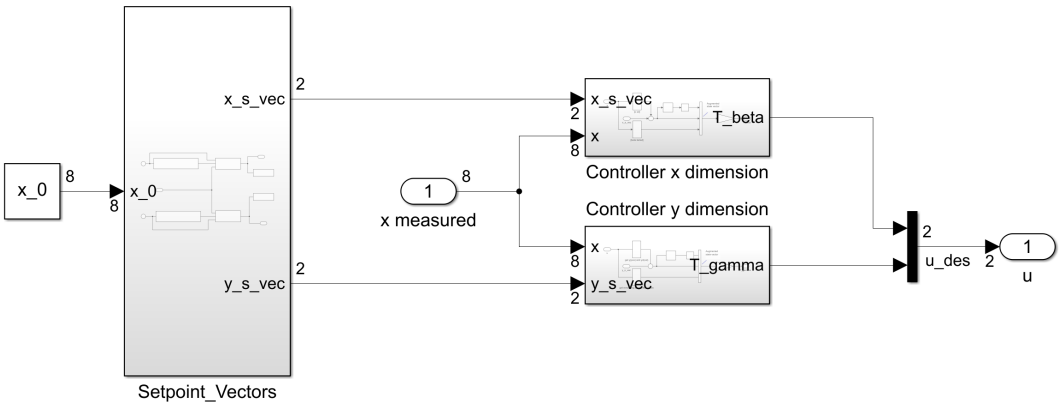


Figure 3.3: Outer Control Architecture - FSFB With Integral Action

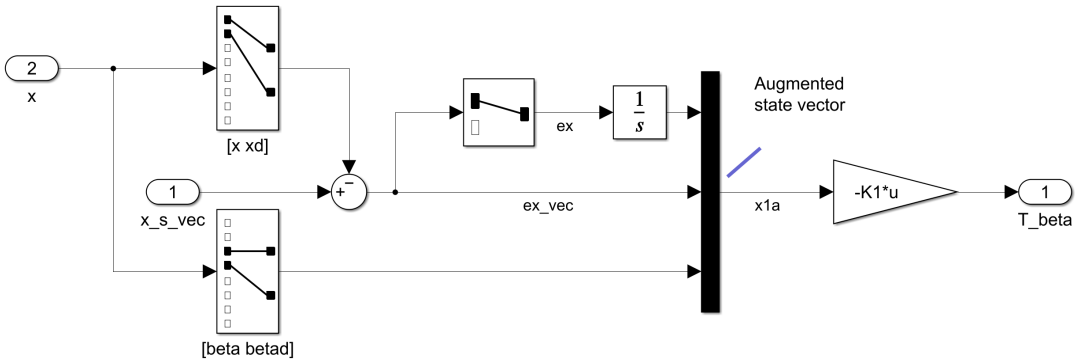


Figure 3.4: x Axis Control Architecture - FSFB With Integral Action

The lqr (Linear-Quadratic Regulator) function in MATLAB® was used to find stabilizing gain matrices (3.12) and (3.13).

$$K_1 = [\ -516.2 \quad -1209.2 \quad -378.7 \quad 409.1 \quad 47.6\ ] \tag{3.12}$$

$$K_2 = [\ 516.2 \quad 1209.2 \quad 378.7 \quad 409.1 \quad 47.6\ ] \tag{3.13}$$

The Q and R matrices, (3.14) and (3.15), were manually tuned to prevent instability due to input saturation, and to produce reasonable regulation, step, and quadrifolium responses in simulation.

$$Q = \begin{bmatrix} 2.5 \times 10^7 & 0 & 0 & 0 & 0 \\ 0 & 10^8 & 0 & 0 & 0 \\ 0 & 0 & 10^6 & 0 & 0 \\ 0 & 0 & 0 & 0.5836 & 0 \\ 0 & 0 & 0 & 0 & 0.1111 \end{bmatrix} \tag{3.14}$$

$$R = 93.8262 \tag{3.15}$$

The time constant τ of the smoothed setpoint, (3.3), was set to 1. Faster values lead to instability caused by input saturation[5]. The responses for system 1, the x direction[6], are shown in figures 3.5, 3.6, and 3.7. The regulation response of Fig. 3.5 was stable and can be seen to asymptotically approach zero error, as expected.

[5]With the gain matrices chosen for the controller and the setpoints explored, τ of 1 was about as fast as possible. This value may not be good for all types of rose curves or for different tunings.

[6]The responses for system 2, the y direction, are qualitatively equivalent and are not shown.

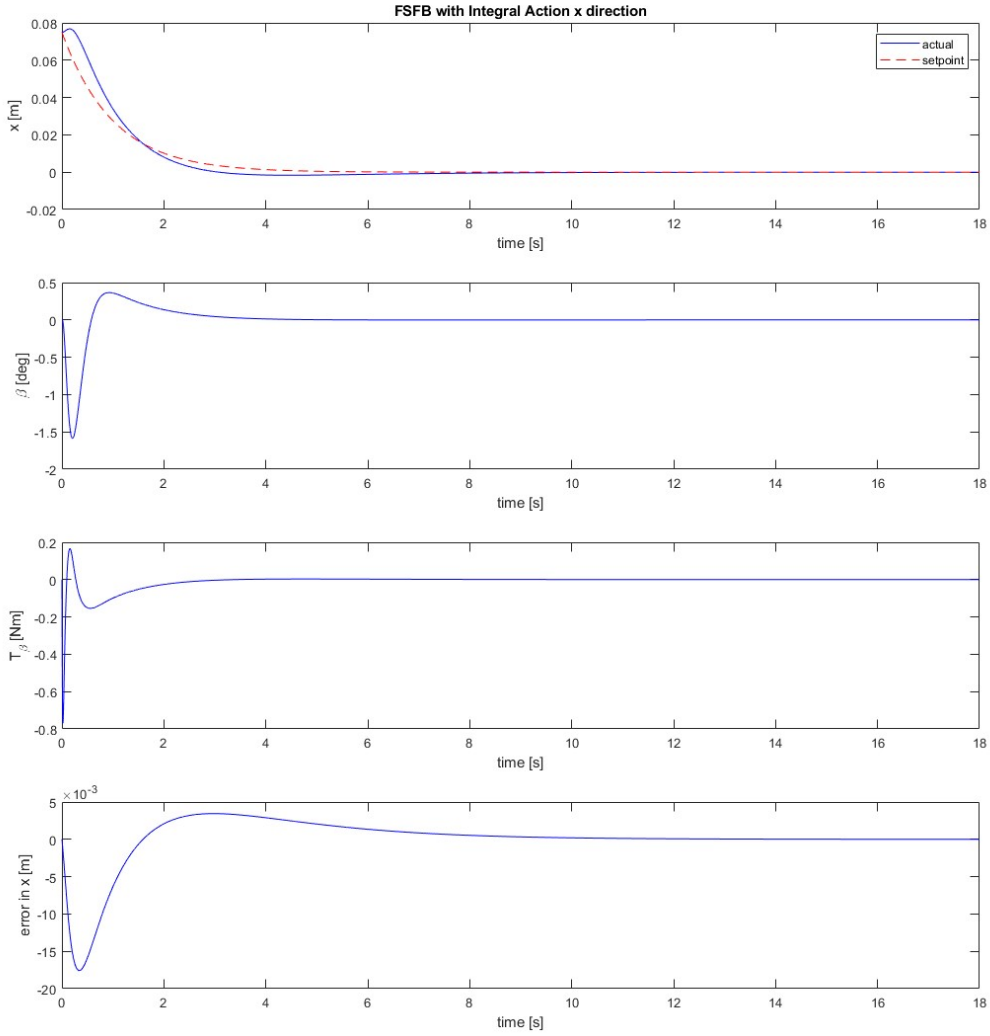


Figure 3.5: Regulation Response - FSFB With Integral Action

Figure 3.5 shows that the controller is able to bring the ball from the initial conditions to the origin. The response does not rely on the integrator in this case, as the steady state torque required to keep the ball at the origin is zero. The implied angular setpoint of zero is dynamically consistent with the ball setpoint, so the objective for the controller is actually achievable. With the quadrifolium setpoint, Fig. 3.7, this is not the case, as the control law contains a dynamically inconsistent angular setpoint that cannot possibly be achieved if the ball is to stay on the trajectory. The step response of Fig. 3.6 also achieves its objective.

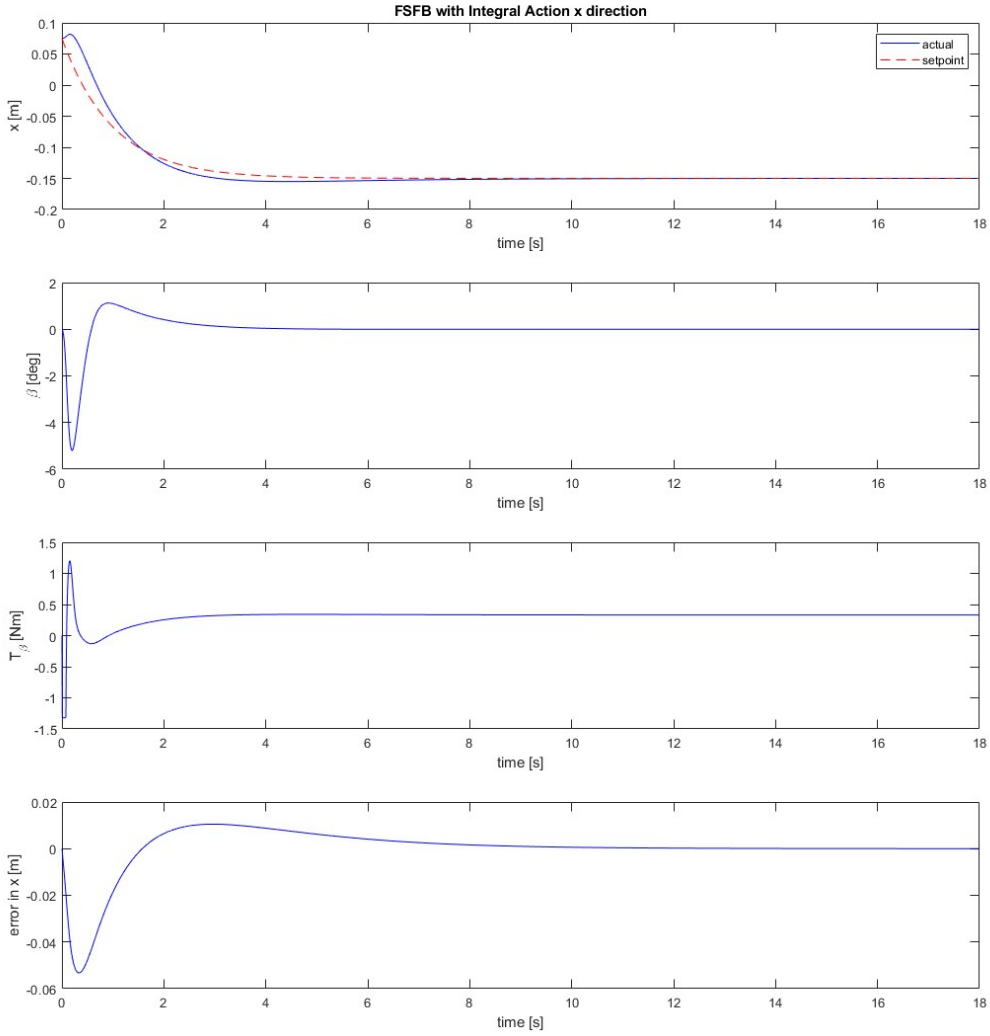


Figure 3.6: Step Response, $x_s = -0.15\ [m]$ - FSFB With Integral Action

This response is simulated with a setpoint of $x_s = -0.15\ [m]$; the edge of the plate[7]. Here, the integrator is supplying the non-zero steady state torque required to hold the ball at the setpoint, away from the origin. Because the angle associated with this scenario is zero, the controller can once again achieve the dynamically consistent angular

[7]Overshoot past the edge of the plate does occur in some simulations, but this is ignored. For the real system, the setpoint can simply be placed inward a bit to prevent the ball from falling off, or the tuning can be adjusted to prevent overshoot.

setpoint. Just as with the regulation controller, the error is seen to asymptotically approach zero. The quadrifolium response, Fig. 3.7, most clearly demonstrates the issues with the architecture. Poor performance is achieved, even for a quadrifolium setpoint that is slow relative to those tracked by the the FSFB with feed forward controller.

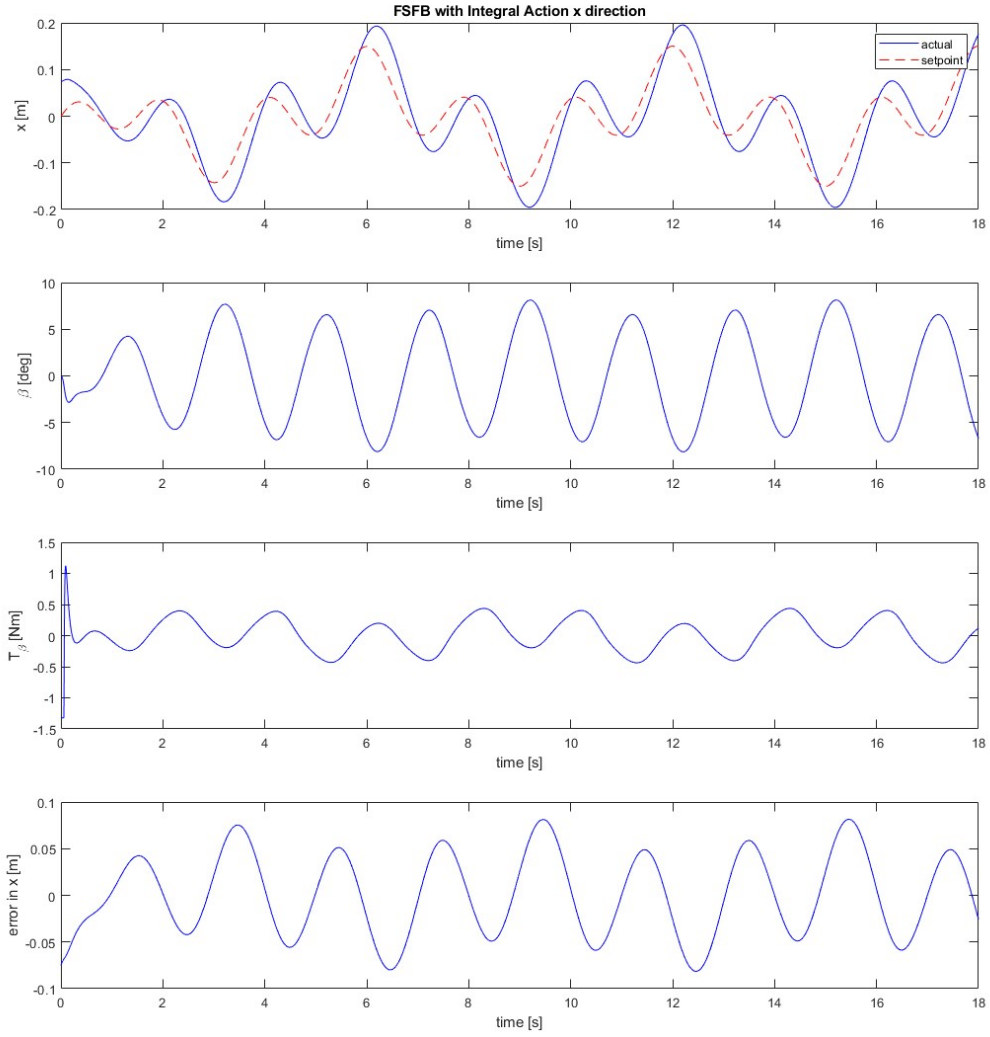


Figure 3.7: Quadrifolium Response, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - FSFB With Integral Action

The quadrifolium period achieved by this control law was limited by the input saturation. Because the frequency components of the setpoint are amplified in the response, the accelerations involved are higher and the achieved trajectory requires higher torque values than would be required at zero tracking error[8]. The ball also lags behind the setpoint, a behavior that, along with the amplification, will produce a larger lock-step Euclidean distance. The result for β is similar to that of T_β and e_x in that the response is some altered combination of the two frequency components of the setpoint. The upside to this is that the plate is in sync with the ball and produces a relatively pleasing response per control criteria laid out in Chapter 2.

Figure 3.8 shows another way of visualizing the quadrifolium response[9]. The shape is clearly exaggerated, and some key features are missing - namely that the ball passes through the origin between each pedal of the rose curve. The response is also a bit skewed, and the degree to which these features are present changes with the particular curve and the tuning.

---

[8]Faster trajectories, ones involving more torque, are easily achieved with the FSFB with Feed Forward controller of Section 3.3. This means that if the FSFB with Integral action is attempting to produce an achievable trajectory given the system's physical limitations.

[9]The small circles in this figure are to indicate the desired location and actual location of the ball at a particular time - they do not represent the size of the ball.

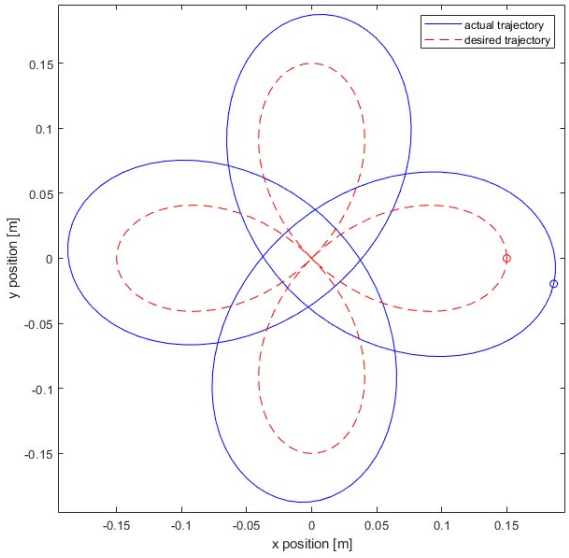


Figure 3.8: Quadrifolium Response, Ball Path, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - FSFB With Integral Action

One might inquire about altering the setpoint such that this architecture's response lands on the actual desired trajectory. For instance, if the setpoint is a circle, the law could first filter the setpoint and produce some phase and amplitude difference. This would be done in such a way that whatever phase lag or attenuation/amplification the law produces places the response precisely on the desired trajectory. Since the rose curve is comprised of two frequencies, and other trajectories may be comprised of even more, the setpoint conditioning would need to alter each frequency component differently. For more complicated signals, this would be incredibly cumbersome. If this could be achieved, the controller would have no way of combating model inaccuracies. The true transfer functions may not be exactly what were used to filter the setpoint and the scheme ultimately would have an open-loop element to it.

3.2 SISO Sliding Mode

The angular input sliding mode controller of Section 1.2.4 utilizes an input that has its own, slow, internal dynamics. If a sliding mode architecture can be found that takes advantage of a more fundamental input to the system, the torque, then the controller can truly utilize the Lyapunov stability argument. The problem is that the torque directly influences the entire state vector. Because the system is underactuated, there is no direct path between an input and the ball that does not involve the dynamics of the plate. In Section 1.2.4, the derivation of the control law of Liu and Liang is shown. If the time varying uncertainty term $\Delta f(t)$ were left untouched, the controller could attempt to cancel those nonlinearities.

Following the authors same reasoning, for system 1, the sliding variable would be (3.16) and its derivative would be (3.17).

$$s_1 = \dot{e}_x + \lambda_1 e_x \tag{3.16}$$

$$\begin{aligned} \dot{s}_1 &= \ddot{x}_s - \ddot{x} + \lambda_1 \dot{e}_x \\ \dot{s}_1 &= \ddot{x}_s - [\,1\;0\,]\left(\, M_1^{-1}\, b_1\right) + \lambda_1 \dot{e}_x \end{aligned} \tag{3.17}$$

Here, rather than $\ddot{x}$ being given by (1.33), the formulation containing $\Delta f(t)$, it would be represented as it is in Section 1.1.2, as row one of (1.15). In this way, T_β would be included explicitly. The control law would then utilize this fundamental input whose dynamics are not strongly tied to the dynamics of interest, as they are with Liu and Liang.

With the feedback law implemented, the derivative of the Lyapunov function could be controlled to (3.18)[10] - but only if T_β could be made arbitrarily large.

$$\dot{V} \;=\; -\,c_2\,|s_1| \tag{3.18}$$

The issue is that the actuation that cancels the nonlinear dynamics in $\ddot{x}$ would need to be unbounded if the angular terms are unbounded. This control law attempts to cancel these dynamics using feedback on the entire state vector, but its objective has no regard for the plate behavior. Because of this, the dynamics would essentially be guaranteed to be unbounded[11].

3.2.1 Control Law

In this section, an alternative output variable, y_1, is considered as a way of circumventing the issues in Section 3.2, and this output serves as the starting point for deriving the single-input-single-output (SISO) sliding mode controller presented in this chapter. If an output variable can be chosen that brings in both x and β, then it is conceivable to design a sliding mode control law where the control objective actually involves the plate dynamics. Because our ultimate goal is to control x to x_s, the output variable, y_1, needs to be similar enough to x that controlling y_1 to x_s produces an acceptable response.

[10]The $-c_1 s_1^2$ term is not present here, as it is with Liu and Liang, but $\dot{V}$ is still negative definite.

[11]Without stabilizing feedback on β states, the natural instability of the system will be present. Even if x is tracking perfectly, β states will be unbounded. There are however, unlikely initial conditions that may produce a stable solution.

A major issue with this idea is that of stability. Because there is a many-to-one relationship between the internal states and the output variable[12], unstable states may be coalescing into a bounded output. So a proof of stability in the output variable is not necessarily a proof of stability in the states. This is a very similar problem to the one discussed in Section 3.2.

While selecting an output variable whose stability proves internal state stability is impossible[13], an output that usually produces stable states is given by (3.19). This output has physical significance - it is the global x location of the ball for the decoupled nonlinear model. If the ball were to follow this trajectory, one could look down on the plate along the z-direction and see the ball tracing out the shape of the setpoint.

$$y_1 = xcos(\beta) + z_b sin(\beta) \tag{3.19}$$

This output was chosen intuitively, and a proof of state stability was not found for the SISO sliding mode controller. However, simulation results in Section 3.2.2, show that the law can produce stable trajectories with reasonably good performance in many scenarios. The reason for this may be that if e_{y_1} is zero, large variations in the states are too costly and take the system off the path of the tracked output. Small variations in the states also imply small variations in T_β. Therefore, if T_β is not saturated at one point along the trajectory, adjacent points may be unlikely to saturate and asymptotic tracking of the output can be maintained.

[12]The state space essentially projects onto the output space.

[13]A stable output variable is not in-and-of-itself a proof of internal state stability. However, it may still be possible to formulate a Lyapunov stability argument for the control law.

The control law is formulated using the usual techniques of sliding mode control. The sliding variable is given in terms of the output variable (3.19) using numerical parameters in Table 1.1.

$$
\begin{aligned}
s_1 &= \dot{e}_{y_1} + \lambda_1 \, e_{y_1} && (3.20)\\
s_1 &= \dot{y}_{1s} - \dot{y}_1 + \lambda_1 \, (y_{1s} - y_1) \\
s_1 &= \dot{y}_{1\mathrm{s}} - \cos(\beta) \, \left(\dot{x} + 0.1220 \, \dot{\beta} \right) + \dot{\beta} \, x \, \sin(\beta) && (3.21)\\
&\qquad + \lambda_1 \, (y_{1\mathrm{s}} - 0.1220 \, \sin(\beta) - x \, \cos(\beta))
\end{aligned}
$$

The derivative of the sliding surface (3.23), once again brings in the system dynamics.

$$
\begin{aligned}
\dot{s}_1 &= \dot{e}_{y_1} + \lambda_1 \, e_{y_1} \\
\dot{s}_1 &= \ddot{y}_{1s} - \ddot{y}_1 + \lambda_1 \, (\dot{y}_{1s} - \dot{y}_1) \\
\dot{s}_1 &= \ddot{y}_{1\mathrm{s}} + \sin(\beta) \, \left(\ddot{\beta} \, x + 2 \, \dot{\beta} \, \dot{x} + 0.1220 \, \dot{\beta}^2 \right) && (3.22)\\
&\qquad - \cos(\beta) \, \left(-\dot{\beta}^2 \, x + \ddot{x} + 0.1220 \, \ddot{\beta} \right) \\
&\qquad + \lambda_1 \left(\dot{y}_{1\mathrm{s}} - \cos(\beta) \, \left(\dot{x} + 0.1220 \, \dot{\beta} \right) + \dot{\beta} \, x \, \sin(\beta) \right) \\
\dot{s}_1 &= \ddot{y}_{1s} + \lambda_1 \dot{y}_{1s} + h_1(\mathbf{x}_1, \lambda_1) + P_1 \begin{bmatrix} \ddot{x} \\ \ddot{\beta} \end{bmatrix} && (3.23)
\end{aligned}
$$

Where P_1 is given by (3.24) and $h_1(\mathbf{x}_1, \lambda_1)$ is given by (3.25).

$$
P_1 = \begin{bmatrix} -\cos(\beta) & \sin(\beta) \, x - \cos(\beta) \, 0.1220 \end{bmatrix} \qquad (3.24)
$$

$$h_1(\mathbf{x}_1, \lambda_1) = \sin(\beta)\left(x + 2\,\dot{\beta}\,\dot{x} + 0.1220\,\dot{\beta}^2\right) + \dot{\beta}^2\,x\cos(\beta) + \lambda_1\left(-\cos(\beta)\left(\dot{x} + 0.1220\,\dot{\beta}\right) + \dot{\beta}\,x\,\sin(\beta)\right) \tag{3.25}$$

In (3.23) T_β enters the equation when $\ddot{x}$ and $\ddot{\beta}$ are replaced with their definitions in (1.15). With the length and complexity of these equations, writing this step out explicitly is cumbersome and unhelpful. A more concise representation is presented here, but a bit of a notation shift must occur. The general form of the decoupled nonlinear equations for system 1 are given by (1.5)[14]. For this analysis, (3.26) gives an example of the notation to be used. Here, "k" denotes the k$^{\text{th}}$ row of the decoupled nonlinear state equations, and all terms in (3.26) are scalars.

$$\dot{x}_{1,k} = f_{1,k} + g_{1,k}T_\beta \tag{3.26}$$

Now (3.23) can be rewritten as (3.27) using the new notation.

$$\dot{s}_1 = \ddot{y}_{1s} + \lambda_1\dot{y}_{1s} + h_1(\mathbf{x}_1, \lambda_1) + P_1\left(\begin{bmatrix} f_{1,2} \\ f_{1,4} \end{bmatrix} + \begin{bmatrix} g_{1,2} \\ g_{1,4} \end{bmatrix} T_\beta\right) \tag{3.27}$$

The Lyapunov function and its derivative are given by (3.28) and (3.29).

$$V_1 = \frac{s_1^2}{2} \tag{3.28}$$

$$\dot{V}_1 = s_1\,\dot{s}_1 \tag{3.29}$$

[14]Along with (1.15) one can write out a very large $f_1(\mathbf{x}_1)$ expression. To clarify, here $\mathbf{x}_1$ is given by the state vector definition for the decoupled nonlinear model, (1.7).

To produce a negative definite $\dot{V}_1$, T_β is formulated in two parts (3.30). Here,

$$T_\beta = \hat{T}_\beta + \tilde{T}_\beta \tag{3.30}$$

Where $\hat{T}_\beta$, the "nullifying torque", is given by (3.31) and cancels the nonlinear dynamics and makes $\dot{V}_1$ zero. The switching torque, $\tilde{T}_\beta$, makes $\dot{V}_1$ negative definite through a saturation boundary layer on s_1, and is given by (3.32). The switching torque makes the sliding surface s_1 attractive for the system. When at or near $s_1 = 0$, the output error states decay according to (3.20).

$$\hat{T}_\beta = -\left(P_1 \begin{bmatrix} g_{1,2} \\ g_{1,4} \end{bmatrix}\right)^{-1} \left(\ddot{y}_{1s} + \lambda_1 \dot{y}_{1s} + h_1(\mathbf{x}_1, \lambda_1) + P_1 \left(\begin{bmatrix} f_{1,2} \\ f_{1,4} \end{bmatrix}\right)\right) \tag{3.31}$$

$$\tilde{T}_\beta = -\left(P_1 \begin{bmatrix} g_{1,2} \\ g_{1,4} \end{bmatrix}\right)^{-1} sat\left(k_{BL} s_1\right) \tag{3.32}$$

The switching torque being implemented as a saturation on s_1 means that for a sufficiently small value of the term, the law contains linear feedback on s_1, with gain k_{BL}. Because the decoupled nonlinear model has inaccuracies[15], the the sliding variable will stay bounded near $s_1 = 0$, but will not be perfectly zero[16]. This is the trade off necessary to eliminate actuator chattering.

[15]The control law contains an internal model of the decoupled nonlinear dynamics - this is a source of uncertainty that sliding mode attempts to handle. If this control law contained an internal model of the coupled nonlinear dynamics, there still might be error in s_1 since there would be numerical error in the solver.

[16]So long as T_β itself does not saturate. There are SISO sliding mode responses that go unstable because the internal states become so violent that T_β saturates and s_1 can no longer be made an attractive surface

3.2.2 Simulation Results

As with Section 3.3.2, the Simulink® model for this controller is comprised of two distinct dimensions. The setpoints are once again analytical functions that enter in through MATLAB® function blocks in the "Setpoint Vectors" subsystem. The outer loop is shown in Fig. 3.9.

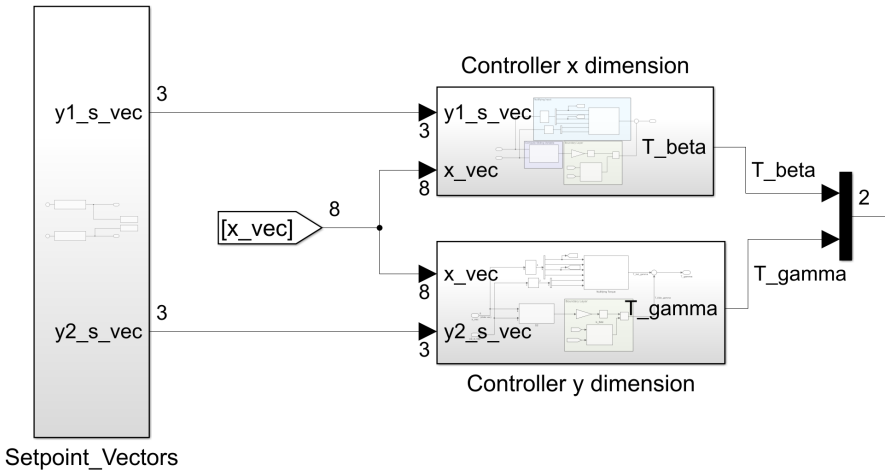


Figure 3.9: Outer Control Architecture - SISO Sliding Mode

The inner loop, seen in Fig. 3.10, shows a much more complicated scheme than for the linear controllers. The nullifying torque is a large nonlinear expression implemented as a MATLAB® function block. The sliding variable is computed in the "S1" subsystem, and is presented in Fig. 3.11. The output variables themselves are no longer trivial, and are computed in this subsystem using function blocks. While the law is derived using output definitions in the decoupled nonlinear model, the measurement of the outputs in the Simulink® model uses the outputs defined in the coupled nonlinear model. This is a potential source of error in the responses shown.

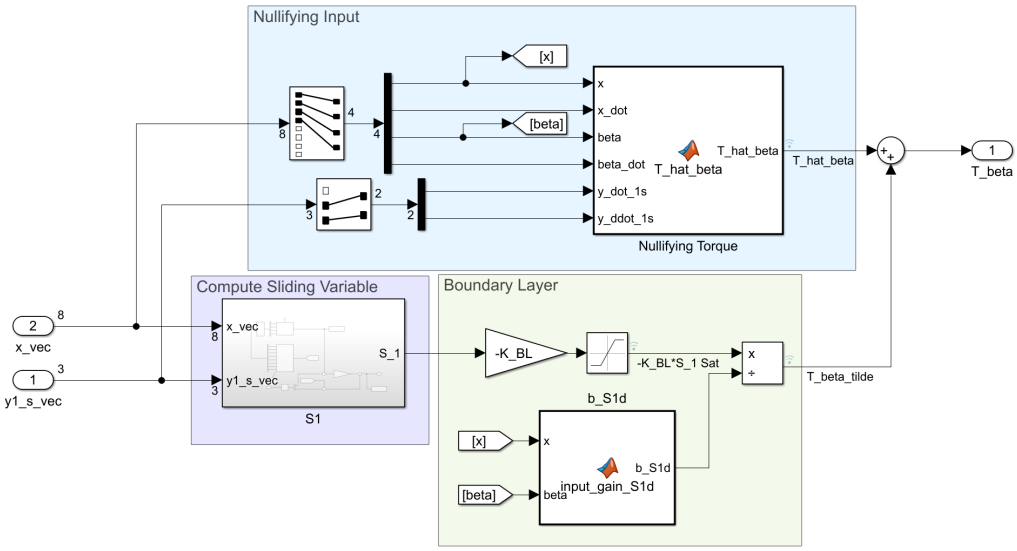


Figure 3.10: x Axis Control Architecture - SISO Sliding Mode

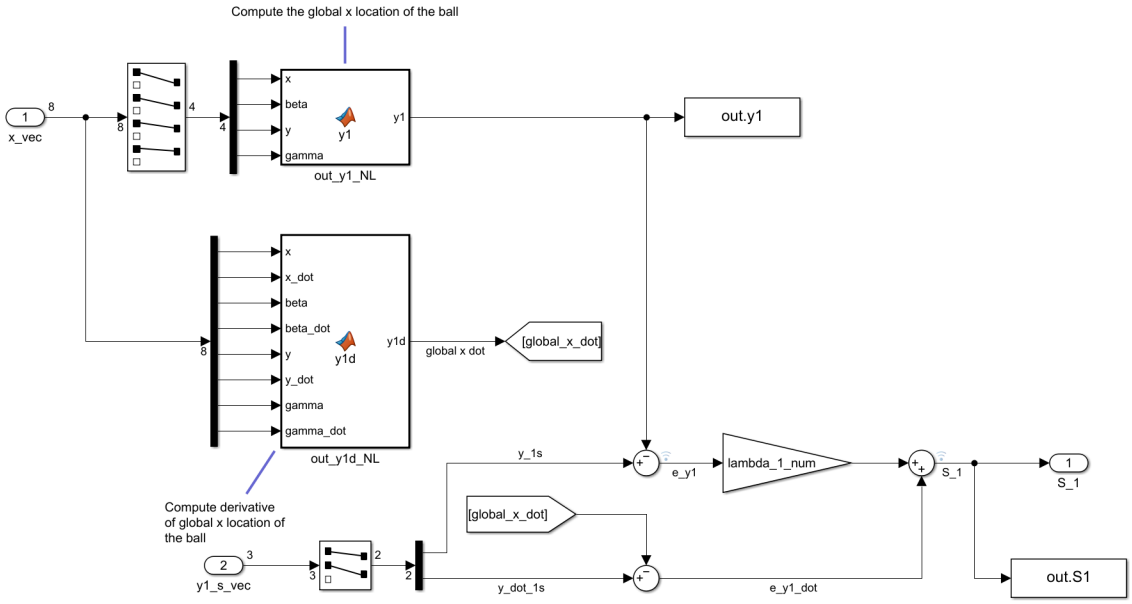


Figure 3.11: s_1 sliding variable computation - SISO Sliding Mode

Tuned parameters for the SISO sliding mode simulations are shown in Table 3.2. Here, k_{BL} is the linear feedback gain applied to s_1 prior to entering the unity slope saturation block in Fig. 3.10. The parameter BL_{size} sets the input value of s_1 at which saturation occurs, allowing us to set some notion of acceptable error. The

parameter BL_{sat} gives the saturated output value of s_1 given k_{BL} and the unity slope of the block, i.e. $BL_{sat} = k_{BL} \times BL_{size}$.

Table 3.2: SISO Sliding Mode Control Parameters

Parameter	Tuned Value
τ	2
λ_1	2
k_{BL}	100
BL_{size}	0.001
BL_{sat}	0.1

The setpoint that the SISO Sliding mode controller handled best was the quadrifolium with a period of 6 seconds. This is shown in Fig. 3.12. The output y_{1s} tracks very well, but x_1, the actual objective for the controller, shows issues. The result is in phase with the setpoint though, and Fig. 3.13 shows that the shape is less exaggerated than that of Fig. 3.8, the ball path of the integral controller. Figure 3.14 shows that the law is able to make $s_1 = 0$ an attractive sliding surface. The response stays bounded within BL_{size} around 0. The regulation and step responses for the controller, however, are shown to be fundamentally flawed, and show performance far inferior to those of more dynamics trajectories.

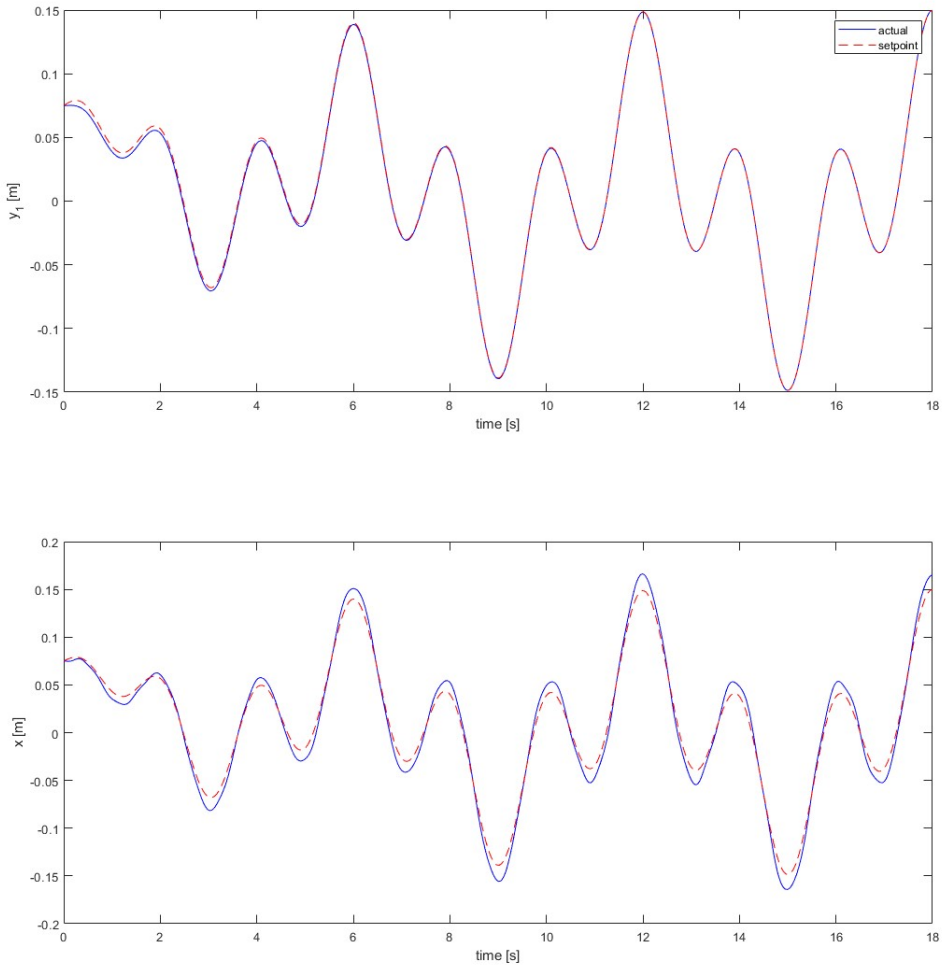


Figure 3.12: Quadrifolium Response, Output vs. State, $r = 0.15\ [m]$**,** $T_r = 6\ [s]$ **- SISO Sliding Mode**

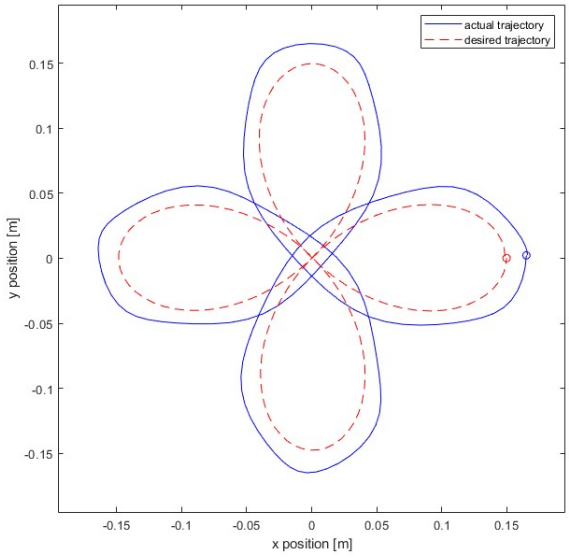


Figure 3.13: Quadrifolium Response, Ball Path, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode

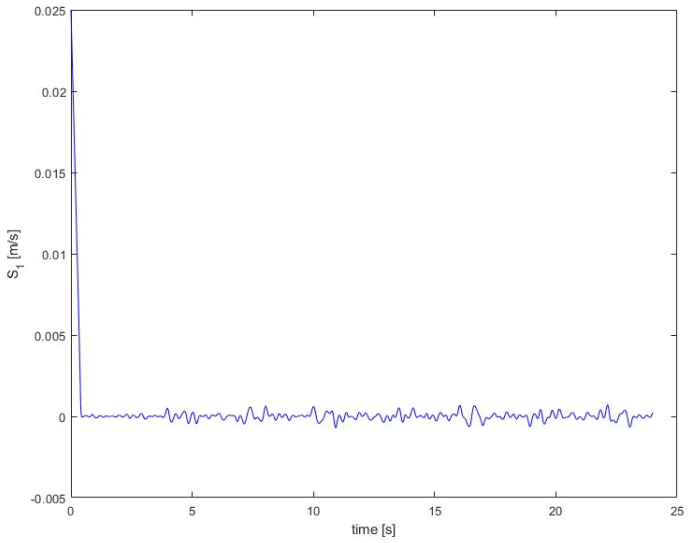


Figure 3.14: Quadrifolium Response, $\mathbf{s_1}$, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - SISO Sliding Mode

There is relatively good performance when one only considers the x_1 response, however the output is allowing an exchange of x_1 and β states to coalesce into y_1, and there

is internal oscillation between these that produces jarring behavior when viewed in animation. Figure 3.15 shows these higher frequencies in both the actuation and the angular response. As the ball tracks the global x and y setpoints, the plate oscillates more often than it would be expected to. This means that this response fails the "visually pleasing" control criteria.

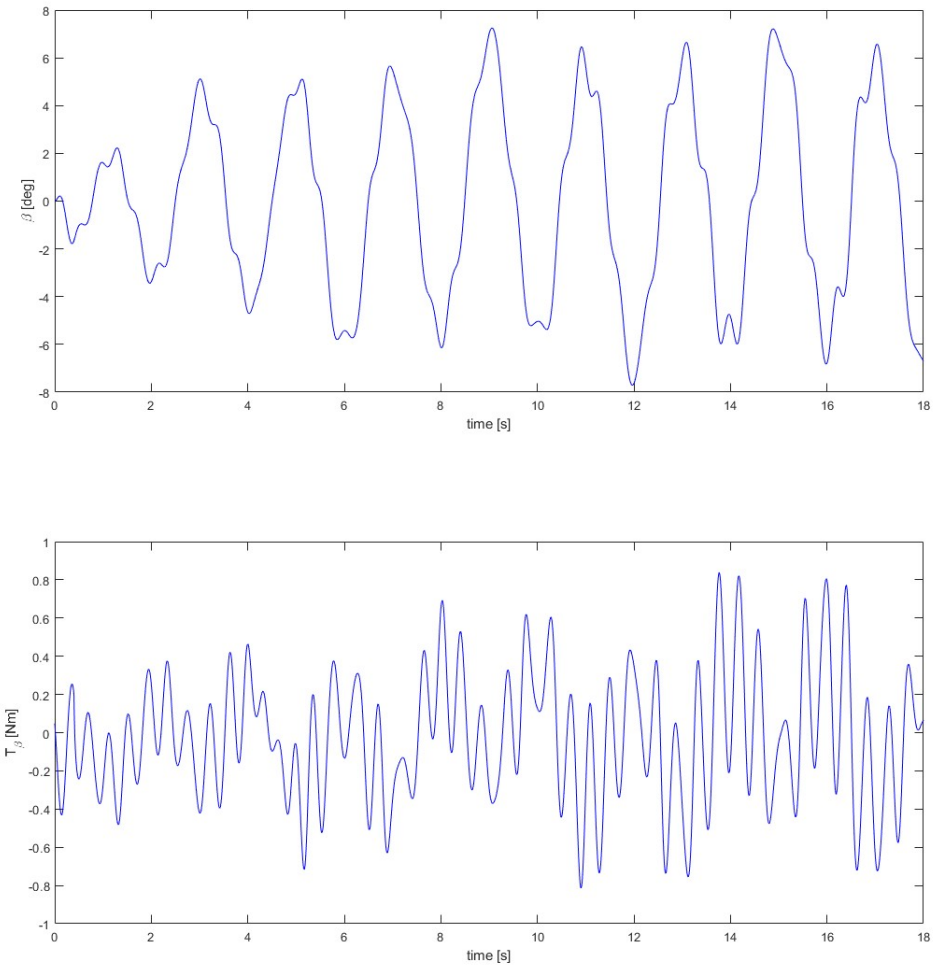


Figure 3.15: Quadrifolium Response, T_β and β, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - SISO Sliding Mode

The control scheme does not always produce a good quadrifolium response, and the resulting instability is different for that of the integral controller. For the integral controller, the instability is caused by the attenuation present in the law. Trajectories are imperfect, and have error that produces exaggerated shapes when compared to the setpoint shape. This happens relatively early in the response. For this controller, internal oscillations present in some of the faster setpoints cause instability long after the trajectory has developed significantly. By making the period 5.15 $[s]$, Fig. 3.16 shows this behavior.

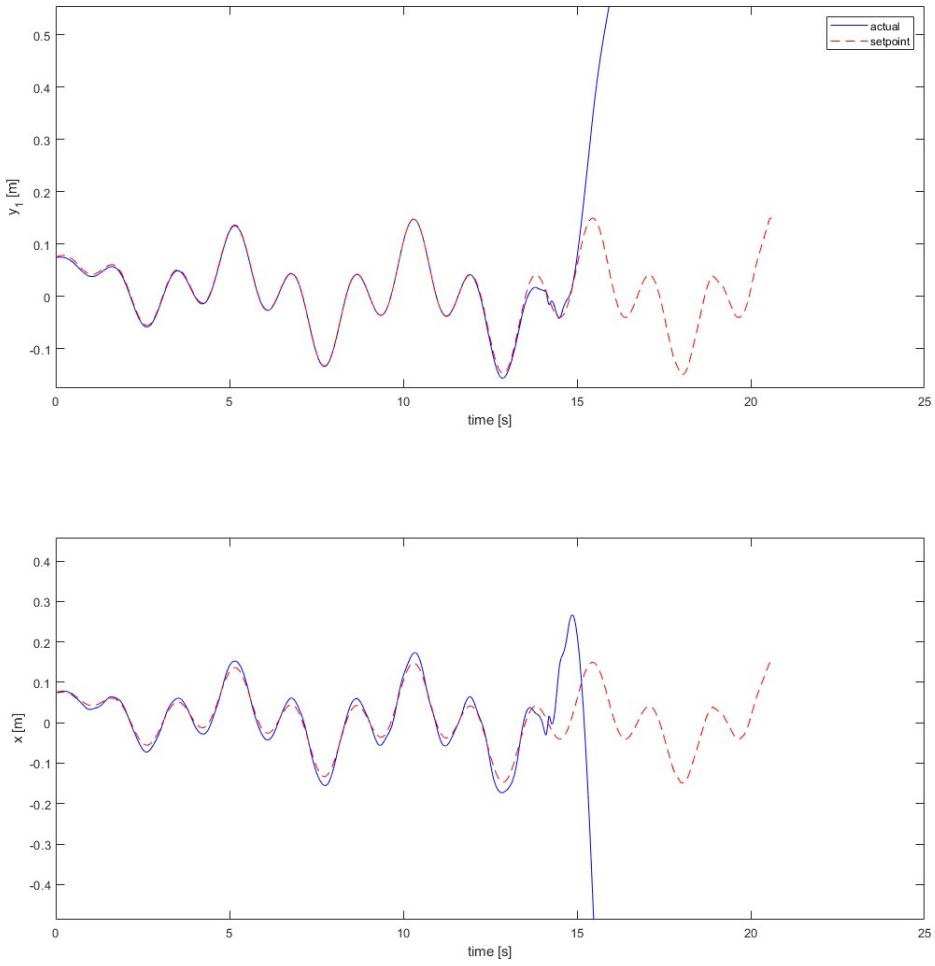


Figure 3.16: Quadrifolium Response, Output vs. State, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ **- SISO Sliding Mode**

Here, the system leaves the sliding surface after it has already entered the boundary layer (Fig. 3.19)[17]. This is because the internal oscillations of the states were dynamic enough to cause $\hat{T}_\beta$ to saturate T_β and the sliding surface could not be kept attractive by $\tilde{T}_\beta$. Figure 3.17 shows β oscillations becoming more violent and T_β saturating.

[17]There is no guarantee that the response in Fig. 3.12 wouldn't also leave the sliding surface eventually. The state response does appear to be continuously developing and may eventually break down. This is a potentiality for all responses given the nature of the law, and is a considerable downside of the architecture.

Figure 3.18 shows that $\hat{T}_\beta$ is responsible for the saturation in T_β and Figure 3.19 shows the system leaving the sliding surface.

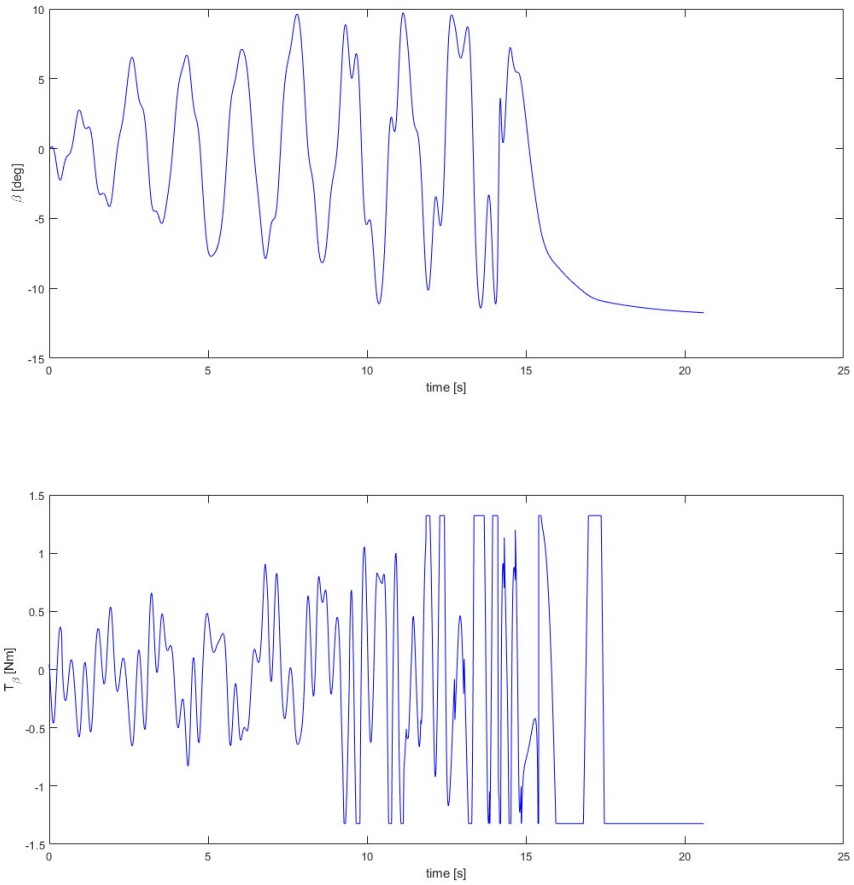


Figure 3.17: Quadrifolium Response, T_β and β, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode

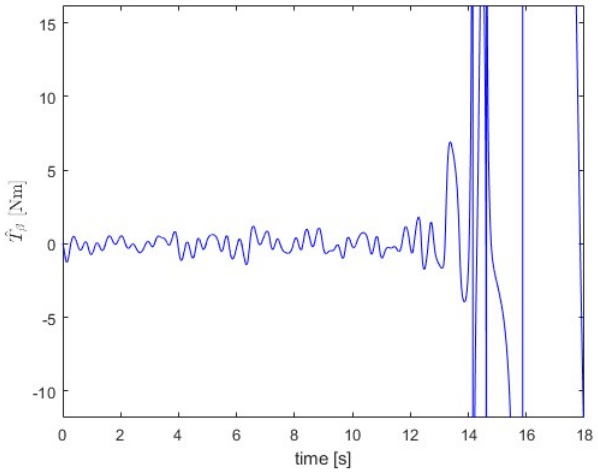


Figure 3.18: Quadrifolium Response, $\hat{T}_\beta$, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode

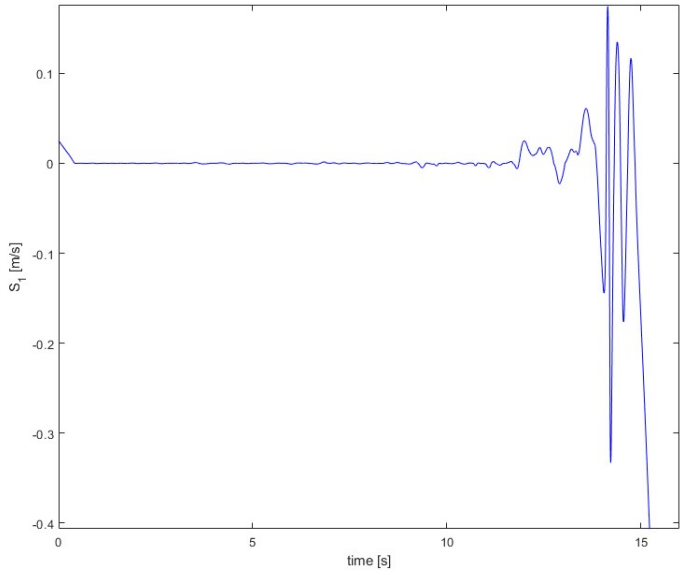


Figure 3.19: Quadrifolium Response, s_1, $r = 0.15\ [m]$, $T_r = 5.15\ [s]$ - SISO Sliding Mode

The issue with internal state oscillation is more visually unappealing for the regulation response. For this, the plate is expected to tilt the ball toward the setpoint, and bring the ball to a standstill at the origin, perhaps with a bit of overshoot. However, the law

does not inherently care about the way in which the output is achieved, and violent oscillations are produced to maintain the ball at a single global x location. Looking down on the plate, one would see a perfectly still ball - yet the plate is wobbling. Figure 3.20 shows this for regulation. The output variable behaves just fine[18], but it is achieved at the cost of an overactive system. The angular and torque responses in Fig. 3.21 show this even better. Here, the plate oscillates by about two degrees, and the motors are constantly active. For regulation, they should theoretically be unnecessary at steady state.

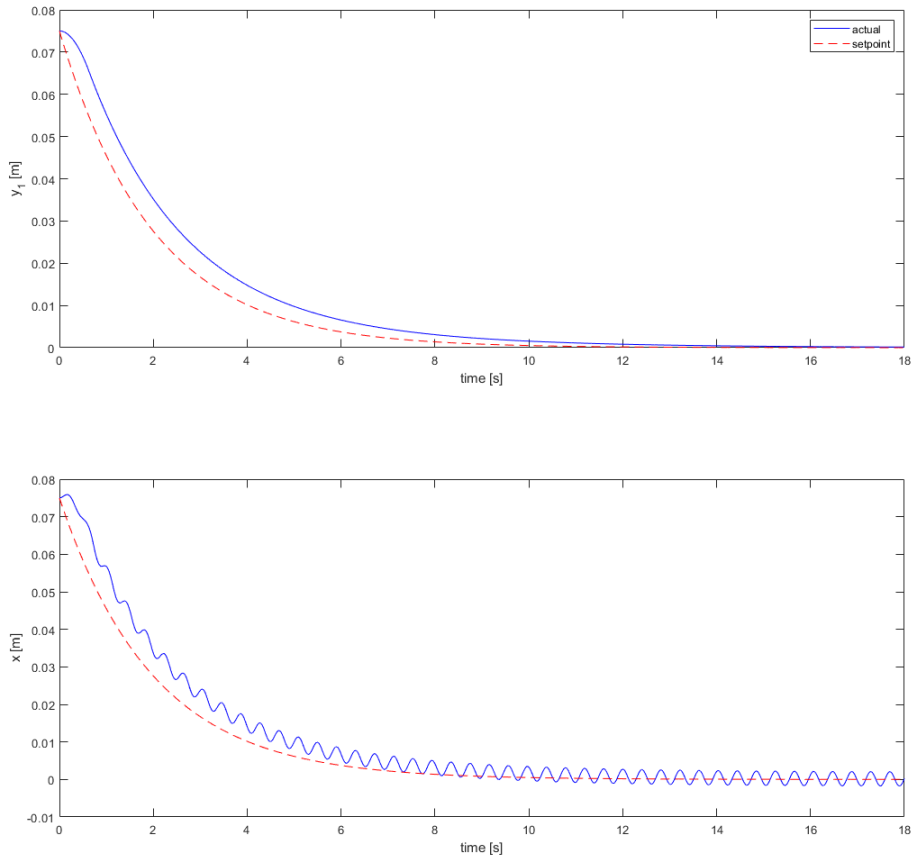


Figure 3.20: Regulation Response - SISO Sliding Mode

[18]If a little slow - with tuning it may be made faster.

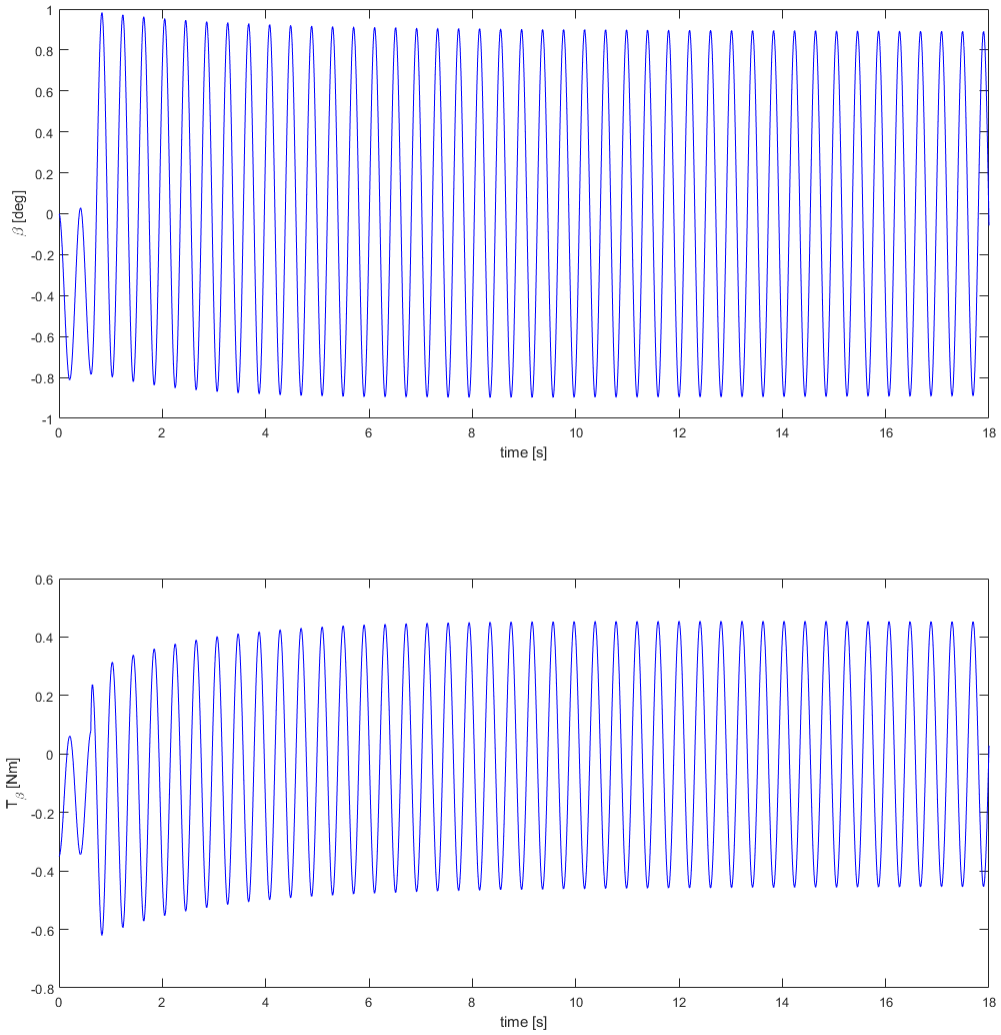


Figure 3.21: Regulation Response, T_β and β - SISO Sliding Mode

The step response has this same issue, and even goes unstable in Fig. 3.22. For this response, the law landed on even more active angular and torque responses, Fig. 3.23, before ultimately breaking down.

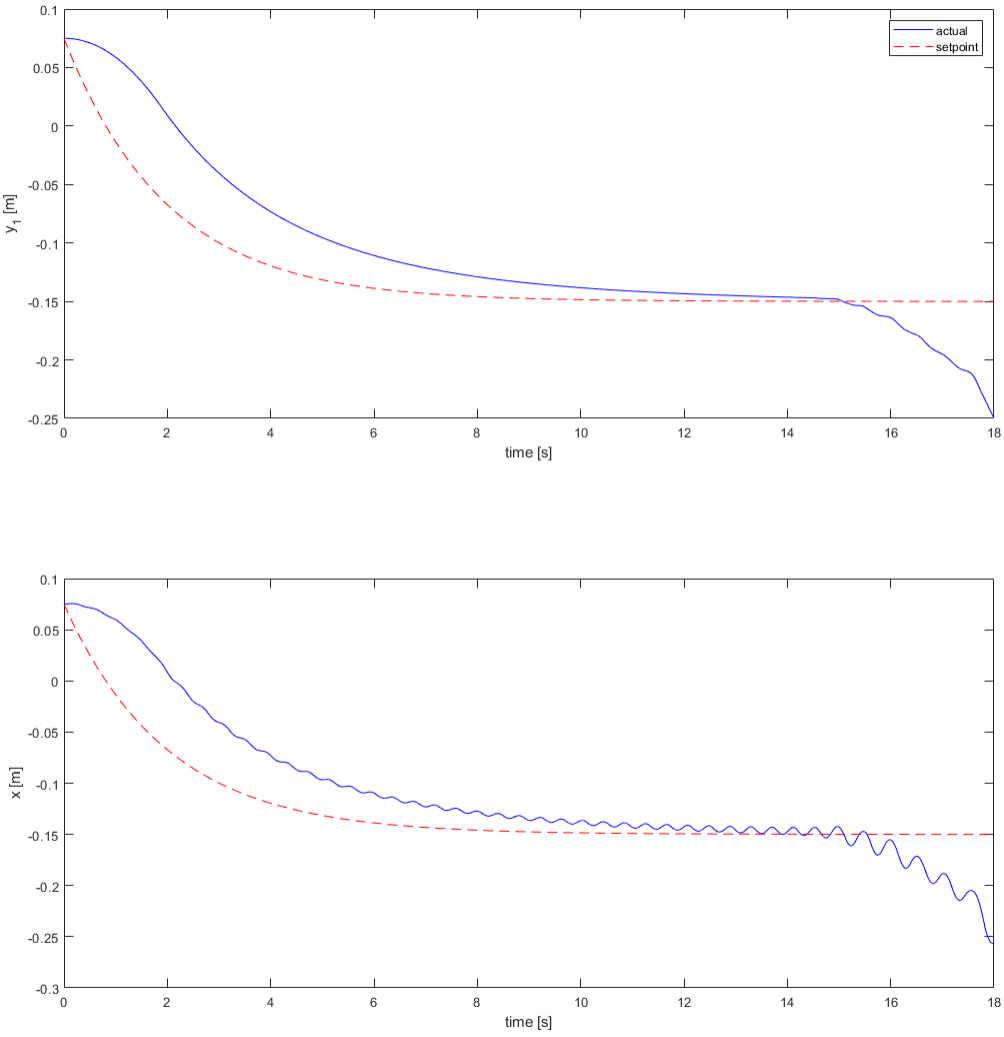


Figure 3.22: Step Response - SISO Sliding Mode

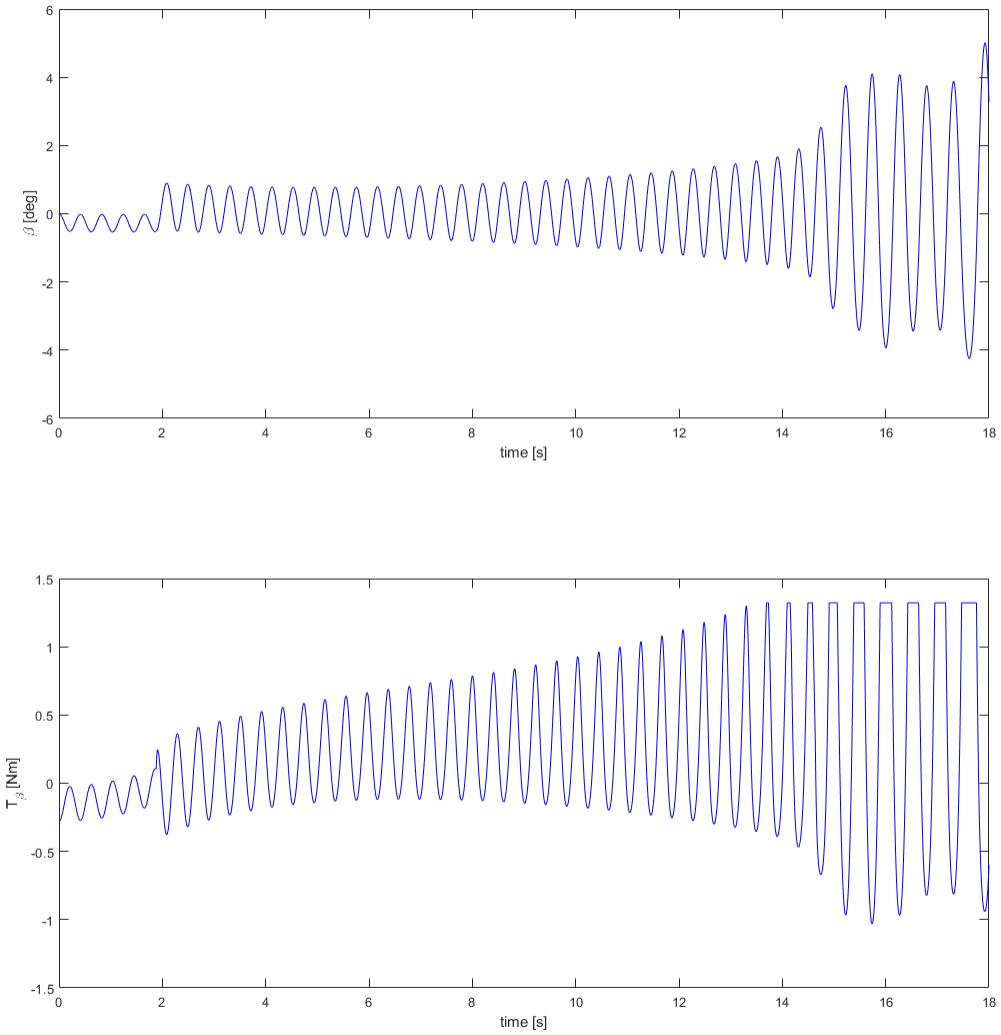


Figure 3.23: Step Response, T_β and β - SISO Sliding Mode

3.3 Full-State Feedback with Feed Forward

In a perfect world, we would have knowledge of ball-and-plate trajectories that produce the desired ball behavior and we could wrap a linear controller around this trajectory. We could select a plate frame trajectory for the ball, determine a non-

linear expression for consistent angular trajectories, and then solve for the torque required to produce that trajectory from any set of initial conditions.

There are problems with this approach. Finding a consistent set of angular trajectories involves analytically solving the highly nonlinear dynamics with a cumbersome amount of terms. Even if we could analytically solve the dynamics, finding a set of initial conditions that produce a stable angular trajectory would be highly non-trivial because the system is naturally unstable. Out of an infinite number of trajectories that produce the desired ball behavior, it may turn out that there is only one that produces stable angular behavior. The inherent instability in the system also makes a numerical approach to solving the nonlinear dynamics difficult. One would need to iterate through initial conditions until a stable trajectory is produced in the simulation output. If it so happens that there are only a few good trajectories, these initial conditions may be very difficult to find.

Despite the setbacks, the idea of planning out a dynamically consistent angular trajectory can be put to good use. If we linearize the system about the operating point, as was done for the FSFB controller of Section 3.1, the task of finding these trajectories becomes much easier. Whether or not the linearizing assumptions hold is determined in simulations using the full nonlinear model. In an ordinary differential equation (ODE) relating the x dynamics to the β dynamics, (3.49), there is an unstable eigenvalue for the open loop system - this is a reflection of the natural instability in the real system. This is shown explicitly in Section 3.3.1. For the nonlinear relationship between x and β it is not so simple as the existence of an unstable eigenvalue.

It is shown that faster trajectories produce more nonlinearity-related performance issues, but great performance can still be achieved. The full-state feedback controller with feed forward discussed in this section relates the angular states to the desired ball trajectory in the linear model. It then finds a dynamically consistent trajectory for

the entire state vector that depends on the choice of setpoint signal[19]. A feed forward torque can be ascertained from the resulting trajectory. To help address offset initial conditions and model uncertainty, a FSFB regulation controller is implemented on the error states.

3.3.1 Control Law

To ensure that the error controller provides integral action, the FSFB with feed forward scheme utilizes a similar augmented state vector to that of Section 3.1. For system 1, this vector is shown in (3.33).

$$\mathbf{e}_1 = \begin{bmatrix} \int_0^t e_x \, \mathrm{d}t \\ e_x \\ \dot{e}_x \\ e_\beta \\ \dot{e}_\beta \end{bmatrix} \tag{3.33}$$

To reason out the control scheme, two additional vectors must be introduced and related to $\mathbf{e}_1$. The augmented state vector for the absolute states, $\mathbf{x}_{1a}$, is shown in (3.34). The setpoint vector for this scheme, $\mathbf{x}_{1s}$, is shown in (3.35) [20].

[19]The setpoint signal, x_s, must be chosen such that a particular solution to the ODE can be found analytically. The particular solution cannot be found easily with numerical methods.

[20]While vectors $\mathbf{x}_1$ and $\mathbf{x}_{1s}$ both contain integrated x and x_s terms, the Simulink® implementation, Section 3.3.2, first constructs e_x and then directly integrates it.

$$\mathbf{x}_{1a} = \begin{bmatrix} \int_0^t x \, \mathrm{d}t \\ x \\ \dot{x} \\ \beta \\ \dot{\beta} \end{bmatrix} \tag{3.34}$$

$$\mathbf{x}_{1s} = \begin{bmatrix} \int_0^t x_s \, \mathrm{d}t \\ x_s \\ \dot{x}_s \\ \beta_s \\ \dot{\beta}_s \end{bmatrix} \tag{3.35}$$

These are related to $\mathbf{e}_1$ in the following way:

$$\mathbf{e}_1 = \mathbf{x}_{1s} - \mathbf{x}_1 \tag{3.36}$$

The dynamics of all three of these vectors are shown below. By manipulating these equations, and applying a FSFB control law on the error states, it can be shown that asymptotic tracking is theoretically possible for the linear system. The dynamics of $\mathbf{x}_{1a}$ and $\mathbf{x}_{1s}$ are given in (3.37) and (3.38). Here, $\hat{T}_\beta$ is the feed forward torque. It is an actuation plan consistent with the setpoint dynamics. If the system is at zero error, this torque will keep the system on the desired trajectory[21].

[21]Matrices A_{1a} and B_{1a} are the same between these two equations because the setpoint obeys the original dynamics of the system

$$\dot{\mathbf{x}}_{1s} = A_{1a}\mathbf{x}_{1s} + B_{1a}\hat{T}_{\beta} \tag{3.37}$$

$$\dot{\mathbf{x}}_{1a} = A_{1a}\mathbf{x}_{1a} + B_{1a}T_{\beta} \tag{3.38}$$

The matrices A_{1a} and B_{1a} in these equations are very similar to (3.7) and (3.8) for the FSFB controller with integral action. The augmented state vectors for the two FSFB controllers differ only by a few minus signs that arise from the error definitions. For this controller, they are:

$$A_{1a} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -1.8786 & 0 & 4.1026 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 20.2622 & 0 & 31.3281 & 0 \end{bmatrix} \tag{3.39}$$

$$B_{1a} = \begin{bmatrix} 0 \\ 0 \\ -0.8511 \\ 0 \\ 9.1798 \end{bmatrix} \tag{3.40}$$

The dynamics of $\mathbf{e}_1$ are derived as follows:

$$\dot{\mathbf{e}}_1 = \dot{\mathbf{x}}_{1s} - \dot{\mathbf{x}}_{1a} \tag{3.41}$$

$$\dot{\mathbf{e}}_1 = A_{1a}\mathbf{x}_{1s} - A_{1a}\mathbf{x}_{1a} + B_{1a}\hat{T}_\beta - B_{1a}T_\beta \tag{3.42}$$

$$\dot{\mathbf{e}}_1 = A_{1a}\mathbf{e}_1 + B_{1a}\tilde{T}_\beta \tag{3.43}$$

Here the input to the error dynamics, $\tilde{T}_\beta$, is introduced:

$$\tilde{T}_\beta = \hat{T}_\beta - T_\beta \tag{3.44}$$

This torque has a direct influence on the error states if $\hat{T}_\beta$ is correctly determined. Using $\tilde{T}_\beta$, a full-state feedback law is introduced, as was done in Section 3.1.1, to place the poles of the error dynamics.

$$\tilde{T}_\beta = -K_1\mathbf{e}_1 \tag{3.45}$$

Continuing from (3.43), the closed-loop error dynamics become (3.46).

$$\dot{\mathbf{e}}_1 = [A_{1a} - B_{1a}K_1]\mathbf{e}_1 \tag{3.46}$$

The result is a stable, homogeneous system of ODEs in the error states that is independent of the chosen setpoint. What's left is to determine the feed forward torque

$\hat{T}_\beta$. This is done by eliminating $\hat{T}_\beta$ from (1.23)[22] and solving the resulting ODE. Rows two and four of (1.23), equations for $\ddot{x}_s$ and $\ddot{\beta}_s$, are each solved for $\hat{T}_\beta$:

$$\hat{T}_\beta = 1.175\ddot{x}_s - 2.2072x_s + 4.82\beta_s \tag{3.47}$$

$$\hat{T}_\beta = 0.1089\ddot{\beta}_s - 2.2072x_s - 3.4127\beta_s \tag{3.48}$$

Equating (3.47) and (3.48), and solving for $\ddot{x}_s$, produces a forced, linear ODE with forcing function determined by the desired setpoint.

$$\ddot{x}_s = -0.0927\ddot{\beta}_s + 7.0071\beta_s \tag{3.49}$$

The general solution to this equation is comprised of a homogeneous part, β_{sH} , and particular part, β_{sP}. The homogeneous part has a single unstable eigenvalue and if this were an initial-value problem (which it is not), this part would depend on the initial conditions. The particular part generally takes on the form of the forcing function $\ddot{x}_s$ and is solely determined by the forcing function.

$$\beta_s = \beta_{sH} + \beta_{sP}$$

$$\beta_s = C_1 e^{-8.6936t} + C_2 e^{8.6936t} + \beta_{sP} \tag{3.50}$$

For a stable open loop system, the ODE could be solved numerically, as an initial value problem, but in this case the unstable eigenvalue poses a problem. Unless the

[22]Equation (1.23) with absolute variables replaced by setpoint variables.

initial conditions used in the numerical solution were just right[23], this part of the solution would show itself and the angular trajectory we ask the controller to track would itself be unstable. The key to benefiting from (3.50) is to realize that this is not actually an initial-value problem, we're simply looking for an angular setpoint that is consistent with the dynamics of the system. The solution does not need to satisfy any initial or boundary conditions, and so long as it is feasible for the system to eventually reach the specified trajectory in β, it is a valid setpoint signal. In other words, the only requirements we have for β_s is that it be consistent with x_s and the equations of motion, and that it is bounded (for stability purposes). A FSFB error control law can handle the offset initial conditions and lead the angular states onto β_s as time progresses.

The term in (3.50) that satisfies the requirements of consistency with x_s and the dynamics, and boundedness is β_{sP}, the particular solution[24]. This part of the solution will be bounded if the forcing function is bounded, and it will have the same frequency content as the forcing function. The range of possible β_{sP} is the frequency response for (3.49), i.e. it contains the same information as its Bode plot. This means that for a periodic setpoint signal, one can extract the amplitude ratio and phase of each frequency component from the Bode plot and construct β_{sP} for all β_s. For this thesis, the MATLAB® symbolic toolbox was used to solve for the particular solution.

For a certain desired setpoint, an expression for β_s is found in this manner, and then substituted, for β_s, into either (3.47) or (3.48) to solve for the feed forward torque

[23]The solver would also have to be very good. Solver errors will cause the unstable eigenvalue to seep back into the solution.

[24]Including the stable term of the homogeneous solution would also produce a β_s that satisfies the requirements, but it is unnecessary to do this.

$\hat{T}_\beta$. Because β_s is analytical, computing $\dot{\beta}_s$ is straightforward. Using (3.44), (3.45) and (3.47), and replacing β_s with β_{sP} , the final control law (3.51) can be derived.

$$\begin{aligned} T_\beta &= \hat{T}_\beta - \tilde{T}_\beta \\ T_\beta &= 1.175\ddot{x}_s - 2.2072x_s + 4.82\beta_{sP} + K_1\mathbf{e}_1 \end{aligned} \tag{3.51}$$

Once again, this control law cannot account for the nonlinearities present in the true dynamics. However, Section 3.3.2, as well as LSED studies in Chapter 4, show that even for highly dynamic trajectories (ones that push motors to the limit) the influence of these nonlinearities is relatively small.

3.3.2 Simulation Results

The Simulink® implementation is very similar to that of the FSFB controller with integral action. The outer and inner loops are given in figures 3.24 and 3.25.

The FSFB with feed forward outer loop shows the addition of a feed forward input,

$$\mathbf{u}_{FF} = \begin{bmatrix} \hat{T}_\beta \\ \hat{T}_\gamma \end{bmatrix} \tag{3.52}$$

and a summing junction to produce the absolute input, per (3.44). The feed forward input is a time dependent actuation plan generated in the symbolic toolbox using the methods of Section 3.3.1. In Fig. 3.25, a setpoint vector in all the states is used to compute the error vector[25].

[25]The integral state is not included in the setpoint vector, as e_x is integrated directly.

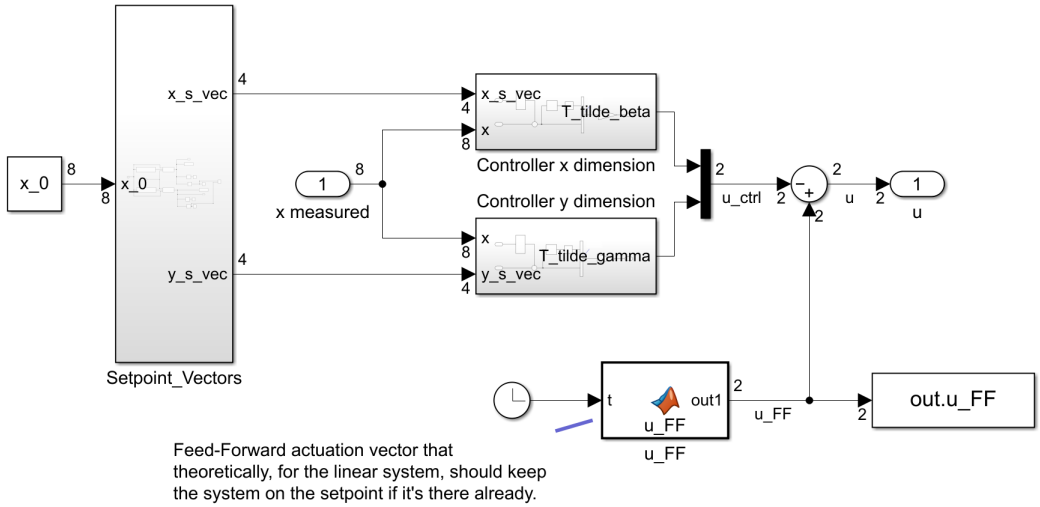


Figure 3.24: Outer Control Architecture - FSFB With Feed Forward

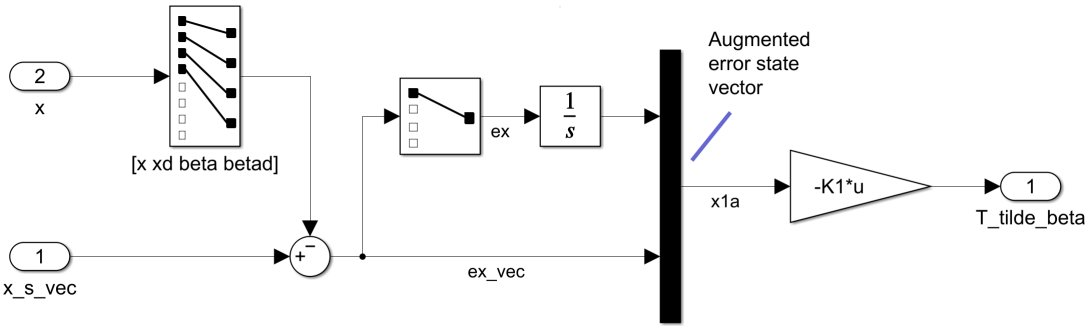


Figure 3.25: x Axis Control Architecture - FSFB With Feed Forward

Gain matrices for this controller, (3.53) and (3.54), were chosen to be identical to those of Section 3.1.2, adjusting of course for sign differences. The time constant τ of (3.3), was set to 1 once again. These choices allow for a more direct comparison of the two architectures.

$$K_1 = [\ 516.2 \quad 1209.2 \quad 378.7 \quad 409.1 \quad 47.6\ ] \tag{3.53}$$

$$K_2 = [\ -516.2 \quad -1209.2 \quad -378.7 \quad 409.1 \quad 47.6\ ] \tag{3.54}$$

Simulation results for the feed forward controller differ from those of the integral controller in that they show, in a red dashed line, the angular setpoint, β_s, and feed forward torque, $\hat{T}_\beta$, that the controller introduces.

With (effectively) the same gain matrices, the regulation response in Fig. 3.26 is identical to that of the Fig. 3.5. For the feed forward controller, β_s and $\dot{\beta}_s$ are zero - which are the same angular setpoints as the implied angular setpoints in the integral action controller. Also, both the feed forward torque of this architecture, and the setpoint vector of the integral controller are zero. Consequently the closed-loop dynamics of the two architectures, (3.11) and (3.46), are identical for regulation[26].

[26]If $\hat{T}_\beta$ is zero, $\tilde{T}_\beta = -T_\beta$ and one need only work out the sign nuances.

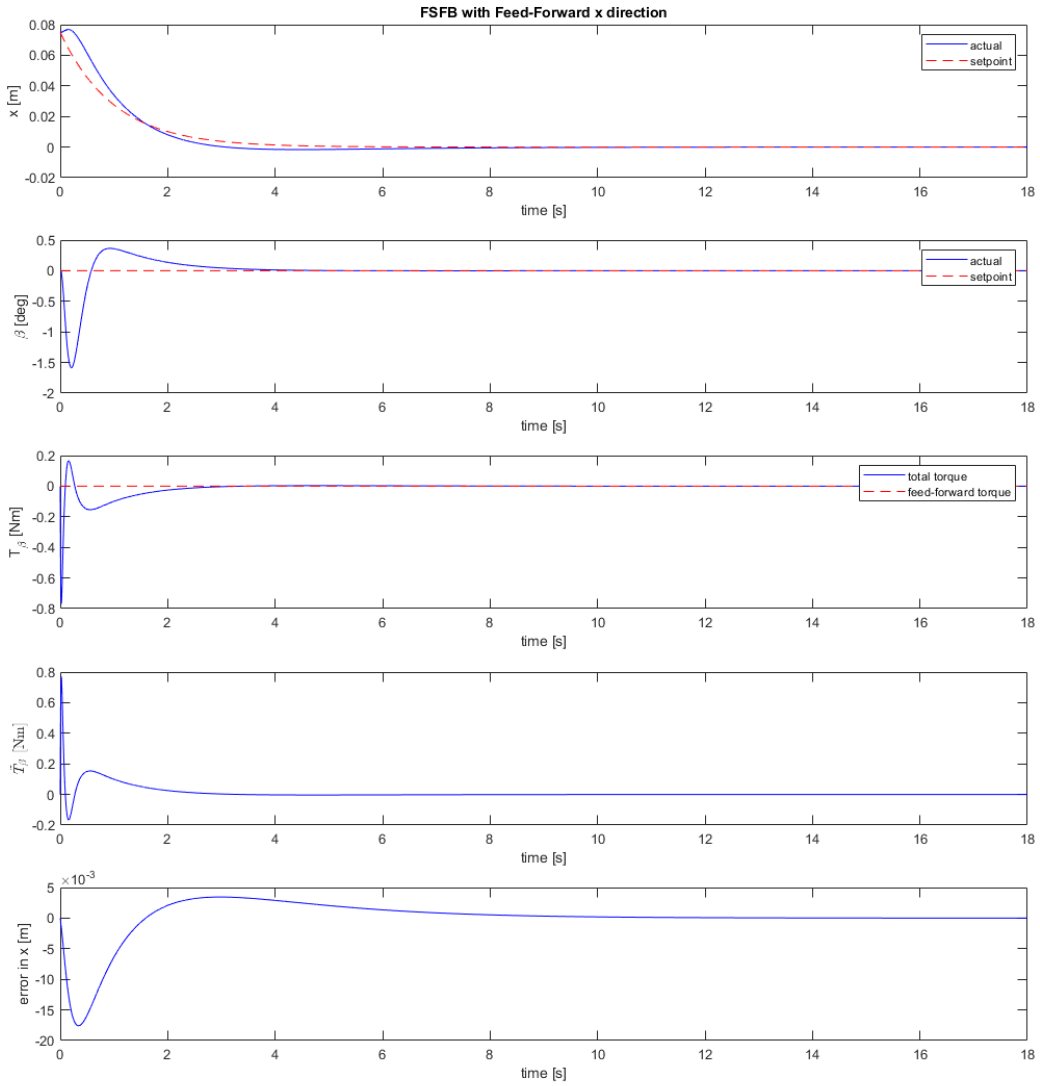


Figure 3.26: Regulation Response - FSFB With Feed Forward

In Fig. 3.26, the angular setpoint of zero is shown explicitly, and the error-affecting torque, $\tilde{T}_\beta$ is shown to approach zero as predicted by (3.46). The error, as with the integral controller, can be seen to asymptotically approach zero. The absolute torque

and the error-affecting torque differ only by a minus sign, as shown in (3.55). As predicted by the law, it approaches $\hat{T}_\beta = 0$.

$$T_\beta = -\tilde{T}_\beta \tag{3.55}$$

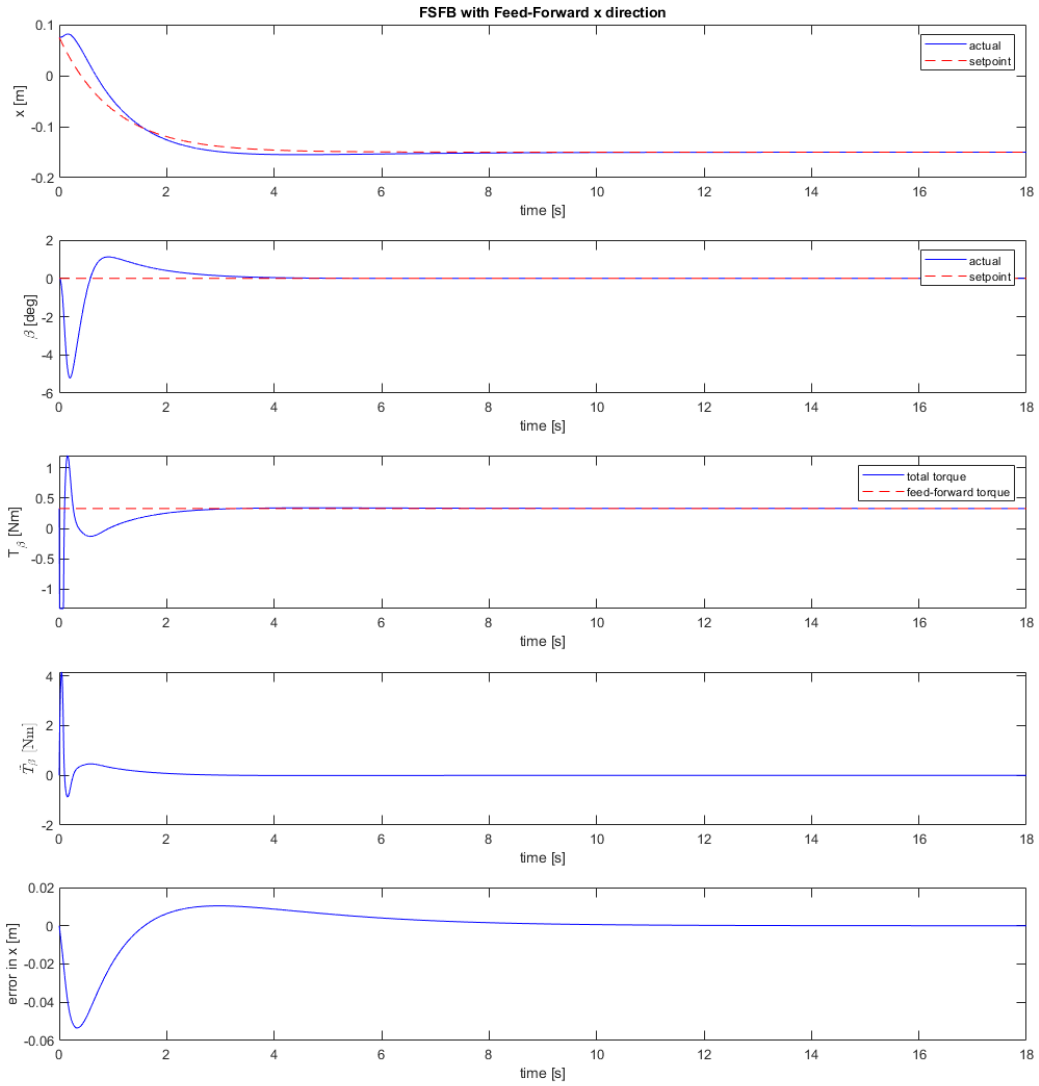


Figure 3.27: Step Response, $x_s = -0.15\ [m]$ - FSFB With Feed Forward

Just as with Fig. 3.26, the angular setpoint for the step response is zero, and T_β approaches $\hat{T}_\beta$ - this time to a nonzero value. As this occurs, the error-affecting torque, $\tilde{T}_\beta$, and the error once again approach zero. The regulation and step responses for both controllers perform well, but the quadrifolium responses in figures 3.28 and 3.29 show the advantage of the feed forward[27].

[27]Section 3.3.3 discusses the architecture with and without the feed forward element. There are great advantages to having dynamically consistent setpoints in the entire state vector whether or not one uses a feed forward torque.

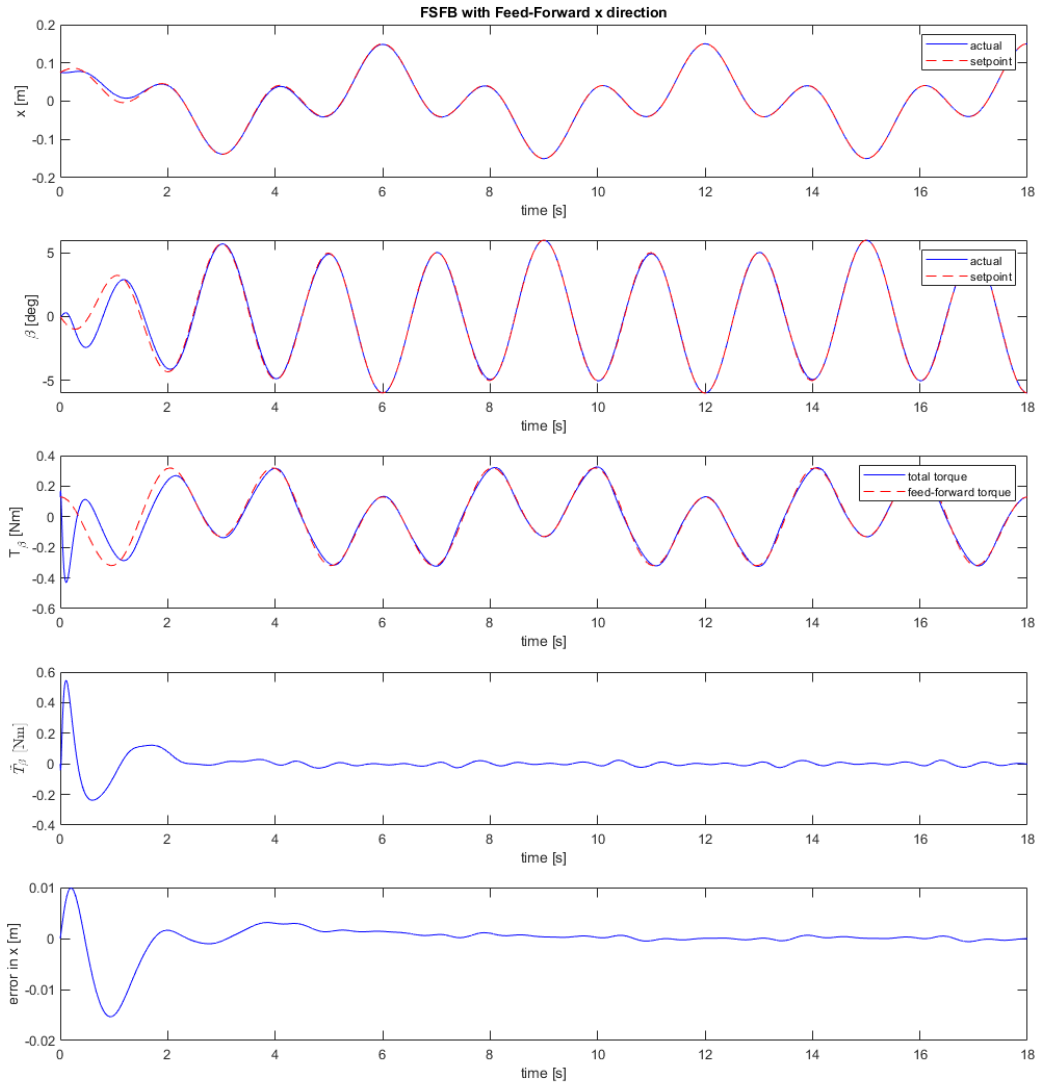


Figure 3.28: Quadrifolium Response, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - FSFB With Feed Forward

For the quadrifolium response with a period of 6 seconds, Fig. 3.28, the performance is far better than that of the integral controller, Fig. 3.7. The initial conditions are almost immediately handled by the controller, with the help of the trajectory smoothing (3.3), and x becomes visually indistinguishable from x_s. For this response,

the β setpoint is non-trivial and is an attenuation/amplification of the two frequency components in x_s, the rose curve frequencies of (3.1). The controller tracks this as well - a necessary condition for x_s to be tracked. Here, even for a more complicated trajectory, the absolute torque approaches $\hat{T}_\beta$. However, since the trajectory is more dynamic than in figures 3.26 and 3.27, there are more nonlinearity errors and neither $\tilde{T}_\beta$ nor e_x truly asymptotically approach zero as predicted by the closed-loop error dynamics for a linear plant model, (3.46)[28].

[28] A simulation using the linear model of the system is presented in 3.3.3 and shows the theoretical behavior for a quadrifolium setpoint.

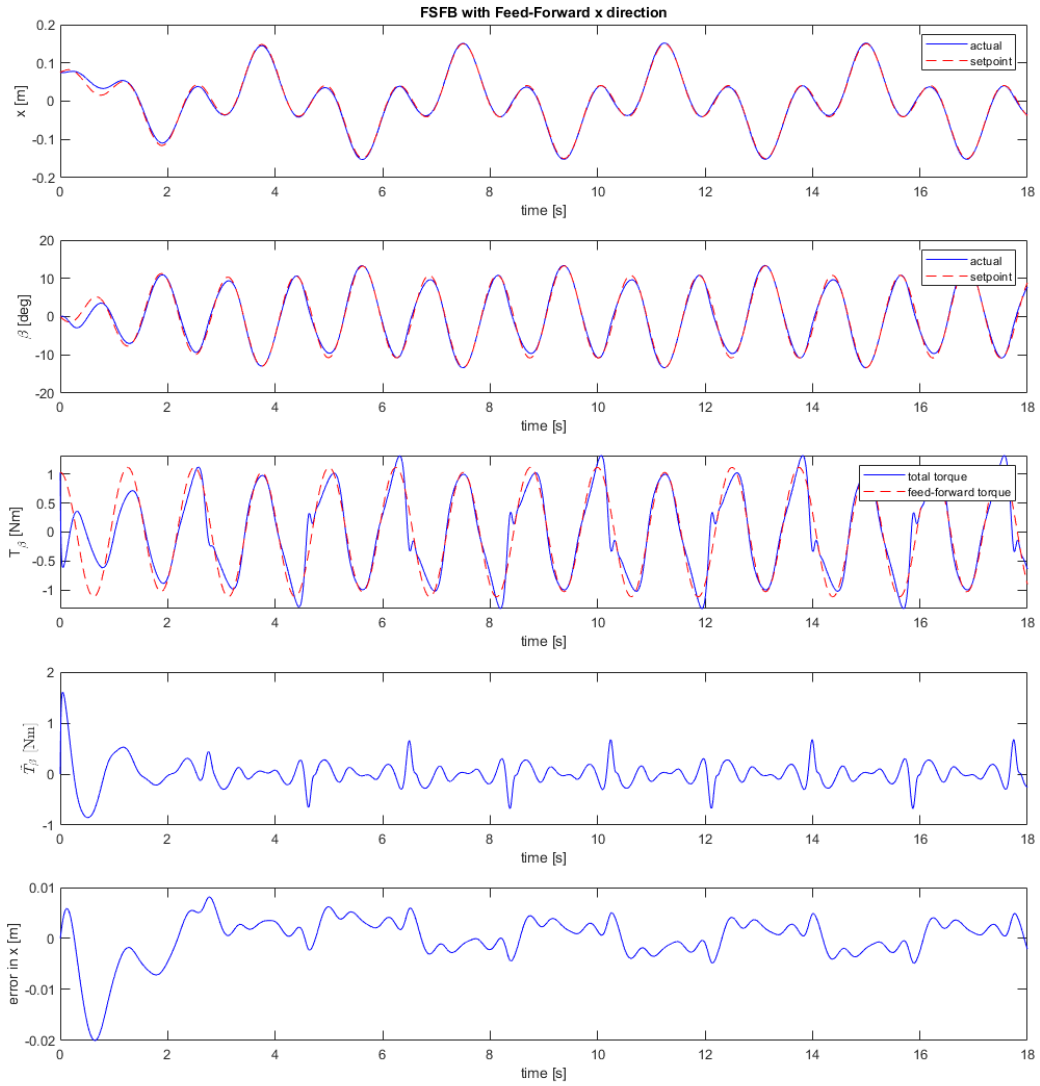


Figure 3.29: Quadrifolium Response, $r = 0.15\ [m]$, $T_r = 3.75\ [s]$ - FSFB With Feed Forward

At a period of 3.75 seconds, good performance is also achieved in Fig. 3.29, though nonlinearity error is prominent due to the aggressiveness of the setpoint. This can be seen most readily in the $\tilde{T}_\beta$ and e_x plots. Still, the x, β, and feed forward paths are tracked relatively well.

The shape carved out by the ball for each of these responses is shown in figures 3.30 and 3.31. The 6 second period shows a far better adherence to the desired shape than that of the integral controller, Fig. 3.8, and with the relatively large ball, it is difficult to notice the small imperfections in the response[29]. The 3.75 second period response is also far better than that of Fig. 3.8, but still shows distortion of the intended shape. In animation, however, it is usually difficult to notice these imperfections.

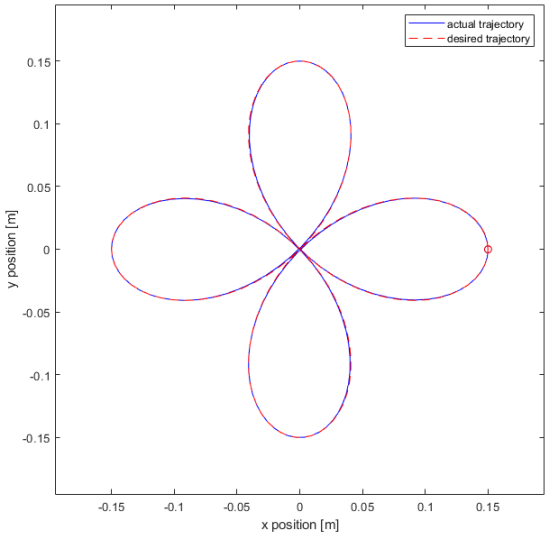


Figure 3.30: Quadrifolium Response, Ball Path, $r = 0.15\ [m]$, $T_r = 6\ [s]$ - FSFB With Feed Forward

[29]3D animations were produced to give a visual experience of the ball-and-plate behavior - these claims are based on experiencing the responses this way. A snapshot of a 3D animation is shown in Fig. 2.1.

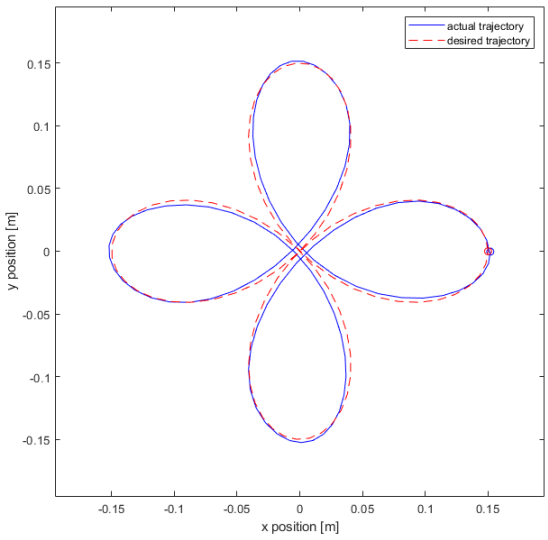


Figure 3.31: Quadrifolium Response, Ball Path, $r = 0.15\ [m]$, $T_r = 3.75\ [s]$ - FSFB With Feed Forward

3.3.3 Turning off the Feed Forward Torque

Turning off the feed forward torque for this architecture, while retaining the feedback element, produced almost indistinguishable responses. Using a dynamically consistent setpoint produces much better performance than setting β_s to zero. However, there is a distinct advantage to using the feed forward torque. An exploration of a control scheme similar to the feed forward controller, but without $\hat{T}_\beta$ is investigated in this section.

To show a theoretical comparison between the two controllers, simulation results using a linear plant model are used. Figures 3.32 and 3.33 show the architecture with and without $\hat{T}_\beta$ implemented. The quadrifolium response for the feedback-only controller show that the law is very effective at tracking the setpoint in both x and β. However, a key difference between the responses is that, for Fig. 3.32, $\tilde{T}_\beta$ does not decay to zero - it actually approaches a signal very close to the computed $\hat{T}_\beta$. Here, e_x behaves

very similarly to that of Fig. 3.33. However, the error can't possibly be zero if it is producing the actuation necessary to keep it there.

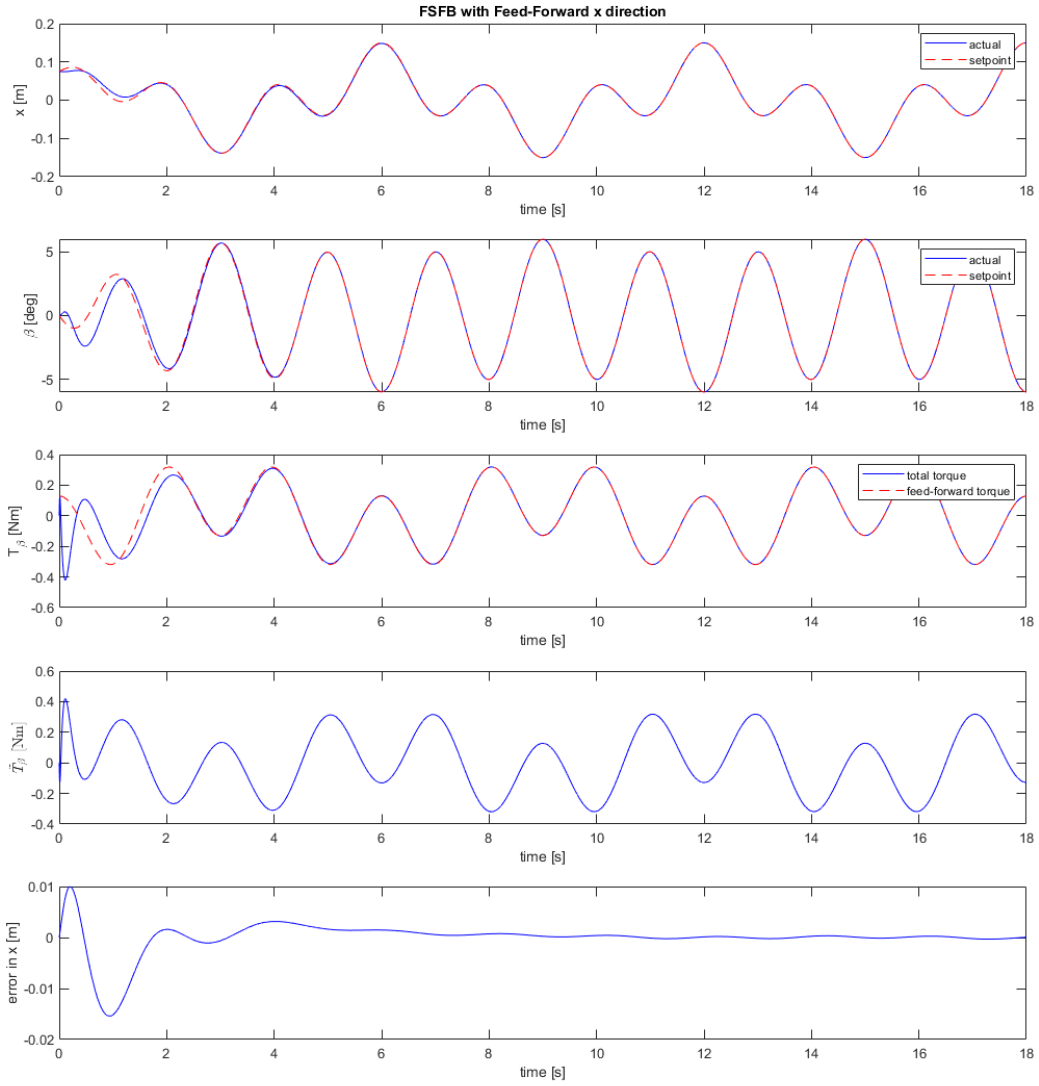


Figure 3.32: Quadrifolium Response, Linear Plant Model, $r = 0.15$ $[m]$, $T_r = 6$ $[s]$ - Feed Forward Turned Off

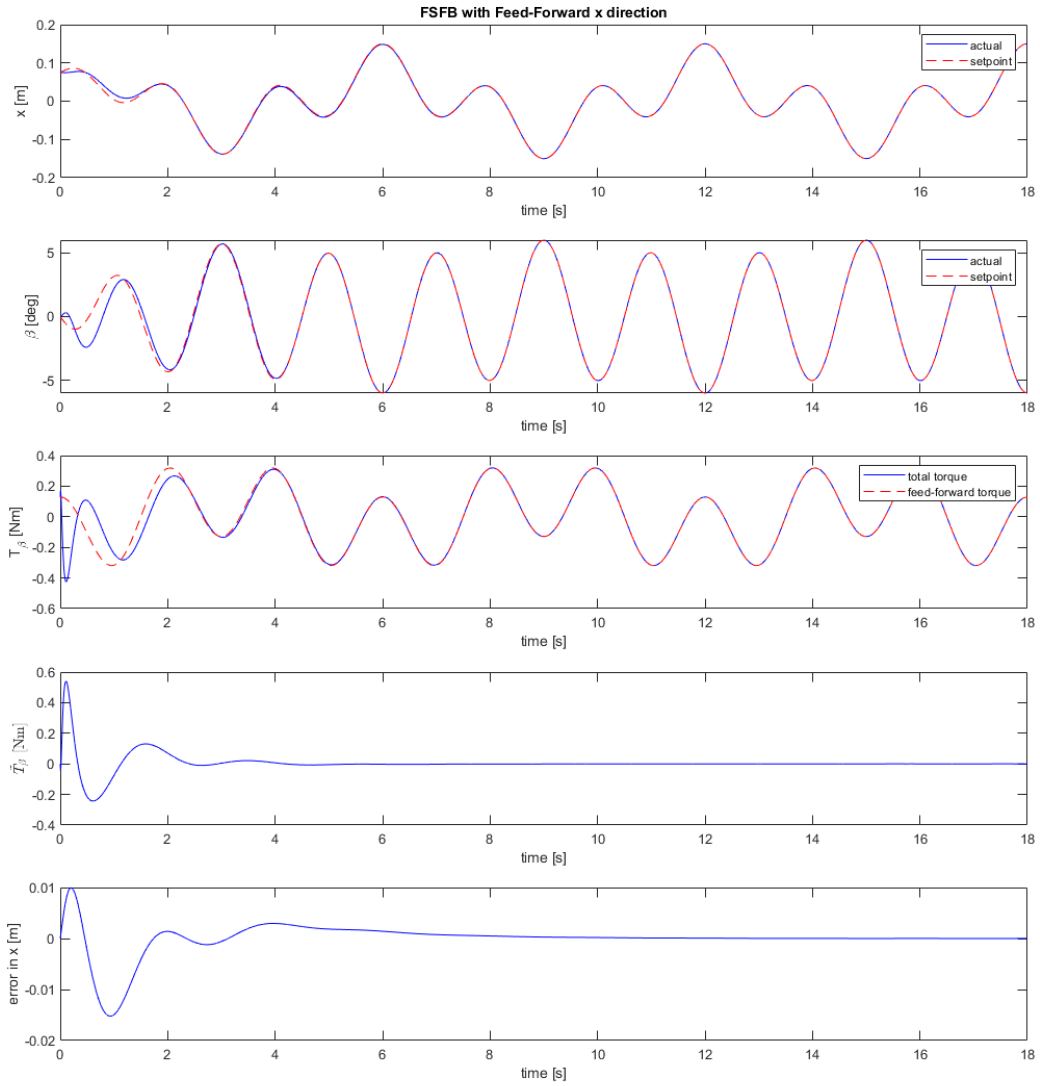


Figure 3.33: Quadrifolium Response, Linear Plant Model, $r = 0.15\ [m]$**,** $T_r = 6\ [s]$ **- Feed Forward Kept On**

It is insightful to look more closely at some of the signals. Figures 3.34 and 3.35 show zoomed in error responses for each controller. The closer scrutiny reveals that only the feed forward controller asymptotically tracks the setpoint. This is because, for

the first controller, e_x needs to contribute to the actuation where in the second, it does not.

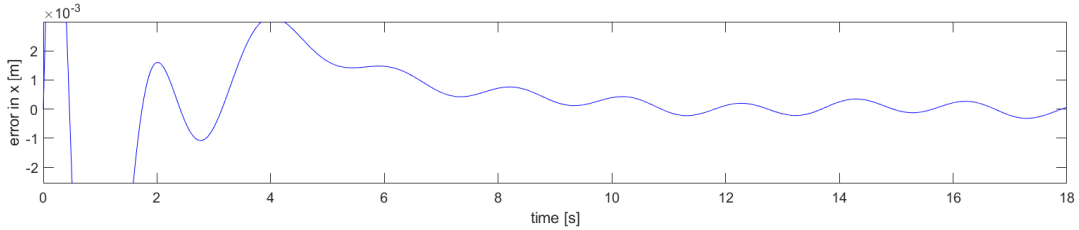


Figure 3.34: e_x zoomed in - Feed Forward Turned Off

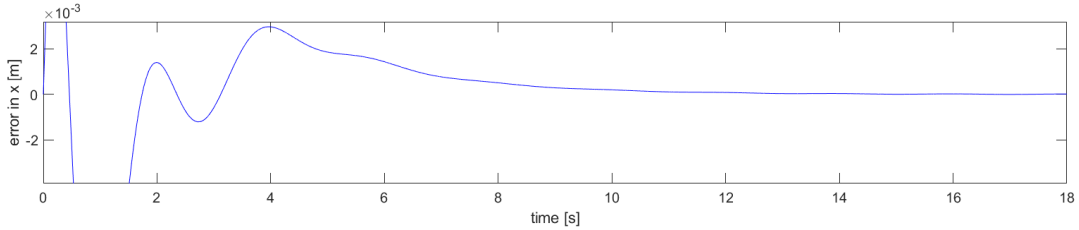


Figure 3.35: e_x zoomed in - Feed Forward Turned On

The T_β response in Fig. 3.32 appears at first to approach $\hat{T}_\beta$. This is illustrated better in Fig. 3.36. However, a zoomed in plot, Fig. 3.37, shows that the difference oscillates around zero with a very small amplitude.

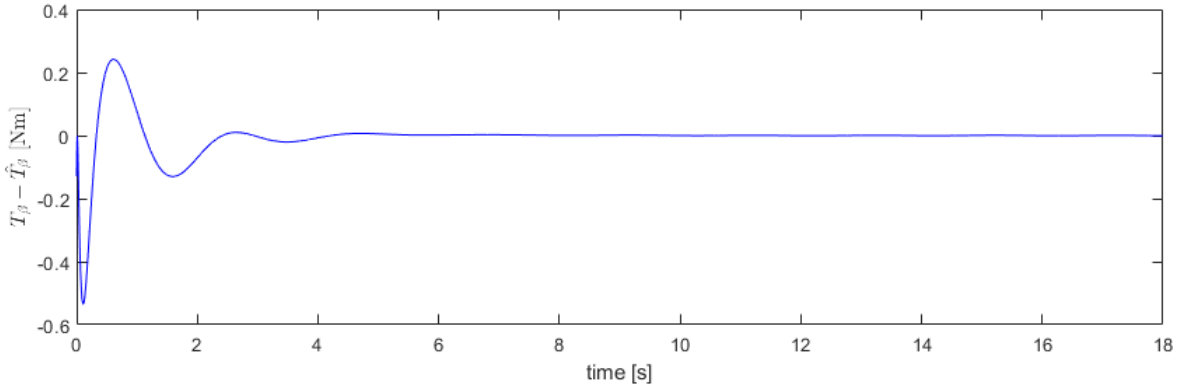


Figure 3.36: $T_\beta - \hat{T}_\beta$ - Feed Forward Turned Off

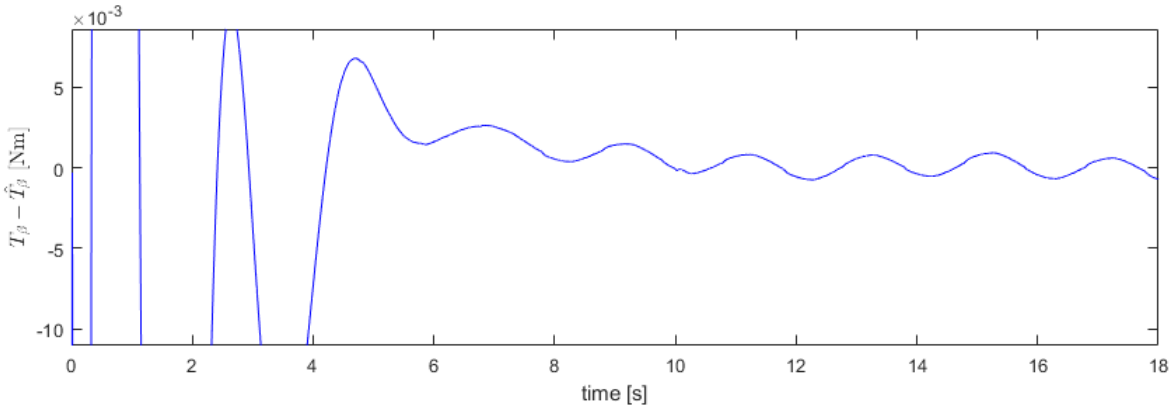


Figure 3.37: $T_\beta - \hat{T}_\beta$ **Zoomed In - Feed Forward Turned Off**

To understand what is going on, we need to take a look at $\dot{\mathbf{e}}_1$ without the feed forward term present. The open loop dynamics of the error vector, $\mathbf{e}_1$, can be expressed in a similar manner to the open loop dynamics of $\dot{\mathbf{x}}_{1a}$ for the integral controller, once again given by (3.6):

$$\dot{\mathbf{x}}_{1a} = A_{1a}\mathbf{x}_{1a} + B_{1a}T_\beta + S_1\mathbf{x}_{1s}. \tag{3.6}$$

Differentiating $\mathbf{e}_1$ produces (3.56), where A_{1a} and B_{1a} are still given by (3.39) and (3.40) respectively, and $\bar{\mathbf{x}}_{1s}$ is given by (3.57).

$$\dot{\mathbf{e}}_1 = A_{1a}\mathbf{e}_1 - B_{1a}T_\beta + S_1\bar{\mathbf{x}}_{1s} \tag{3.56}$$

$$\bar{\mathbf{x}}_{1s} = \begin{bmatrix} x_s \\ \ddot{x}_s \\ \beta_s \\ \ddot{\beta}_s \end{bmatrix} \tag{3.57}$$

The bar notation is introduced to differentiate the setpoint vector used in deriving the feed forward law,

$$\mathbf{x}_{1s} = \begin{bmatrix} \int_0^t x_s \, \mathrm{d}t \\ x_s \\ \dot{x}_s \\ \beta_s \\ \dot{\beta}_s \end{bmatrix}, \tag{3.35}$$

from that used here. The S_1 matrix differs from that of the FSFB with integral action law and is given by (3.58).

$$S_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.8786 & 1 & -4.1026 & 0 \\ 0 & 0 & 0 & 0 \\ -20.2622 & 0 & -31.3281 & 1 \end{bmatrix} \tag{3.58}$$

In the $\tilde{T}_\beta$ relation given in Section 3.3.1,

$$\tilde{T}_\beta = \hat{T}_\beta - T_\beta, \tag{3.44}$$

if no feed forward torque ($\hat{T}_\beta$) is implemented, and only the feedback portion of the law,

$$\tilde{T}_\beta = -K_1 \mathbf{e}_1, \tag{3.45}$$

is used, then the absolute torque, T_β, becomes (3.59).

$$T_\beta = K_1 \mathbf{e}_1 \tag{3.59}$$

With this, the closed-loop dynamics become (3.60).

$$\dot{\mathbf{e}}_1 \;=\; [A_{1a} - B_{1a}K_1]\mathbf{e}_1 + S_1\bar{\mathbf{x}}_{1s} \tag{3.60}$$

This differs from the feed forward closed-loop dynamics of Section 3.3.1,

$$\dot{\mathbf{e}}_1 \;=\; [A_{1a} - B_{1a}K_1]\mathbf{e}_1, \tag{3.46}$$

in that the $S_1\mathbf{x}_{1s}$ term is present and the error states no longer homogeneously decay to the zero vector. The form of (3.60) is exactly like that of (3.11) for the integral controller.

$$\dot{\mathbf{x}}_{1a} \;=\; [A_{1a} - B_{1a}K_1]\mathbf{x}_{1a} + S_1\mathbf{x}_{1s} \tag{3.11}$$

For the FSFB controller with feed-forward turned off, the objective of a controls engineer would be to attenuate the frequency content of $S_1\mathbf{x}_{1s}$ through choice of K_1. This is precisely what the K_1 vector is doing in Fig. 3.32. Rather than this being a coincidence, it may be that decently tuned gains in the integral controller of Section 3.1 are bound to work even better here. Or perhaps the frequency response for this scheme attenuates most reasonable trajectory vectors for this system. These speculations are left for future consideration.

With the integral controller, the design goal is not as clear as it is for (3.60). In the case of the integral controller, attenuating the setpoint for all states is not necessarily a good thing. We want the error states to be small, but the angular states are absolute and shouldn't be small. In fact we want them to be close to the signals consistent with the desired x trajectory. This is a huge pitfall for the integral controller if one is to choose gains using frequency domain analysis.

The feedback law for this section does not implement a feed forward torque, but it can be written in terms of the feed forward torque one would need to get the behavior discussed in Section 3.3. This feed forward term takes the place of the setpoint term $S_1\bar{\mathbf{x}}_{1s}$, and the substitution tells us that failing to supply the feed forward torque is the same as actuating the closed-loop error dynamics with $B_{1a}\hat{T}_\beta$. I.e., a FSFB law utilizing a dynamically consistent setpoint vector, $\mathbf{x}_{1s}$ (3.35), is the same as lumping the feed forward torque of Section 3.3 in with the error dynamics.

$$\dot{\mathbf{e}}_1 \;=\; A_{1a}\mathbf{e}_1 - B_{1a}(\hat{T}_\beta - \tilde{T}_\beta) + S_1\bar{\mathbf{x}}_{1s} \tag{3.61}$$

Substituting $\tilde{T}_\beta$ for the FSFB law, and leaving $\hat{T}_\beta$ in the equation, we get (3.62).

$$\dot{\mathbf{e}}_1 \;=\; [A_{1a} - B_{1a}K_1]\mathbf{e}_1 - B_{1a}\hat{T}_\beta + S_1\bar{\mathbf{x}}_{1s} \tag{3.62}$$

If we were to make (3.62) look exactly like the closed-loop feed forward controller dynamics,

$$\dot{\mathbf{e}}_1 \;=\; [A_{1a} - B_{1a}K_1]\mathbf{e}_1, \tag{3.46}$$

then $B_{1a}\hat{T}_\beta$ would need to be precisely $S_1\mathbf{x}_{1s}$. This establishes an equivalence of the two terms.

$$S_1\bar{\mathbf{x}}_{1s} \;=\; B_{1a}\hat{T}_\beta \tag{3.63}$$

This means we can rewrite (3.60) as (3.64).

$$\dot{\mathbf{e}}_1 \;=\; [A_{1a} - B_{1a}K_1]\mathbf{e}_1 + B_{1a}\hat{T}_\beta \tag{3.64}$$

Equation (3.64) shows that the closed-error dynamics, without the feed forward torque of Section 3.3 implemented, still contains information about that feed forward torque. These error dynamics are now actuated in the same way as the system would be actuated if the error were zero and the ball were on the trajectory. This may or may not have deeper significance, but it is an interesting fact nonetheless.

The task of attenuating $B_{1a}\hat{T}_\beta$ would likely be much easier here than for the integral controller. Since error in every state drives the actuation, the likelihood that the resulting torque is similar to $\hat{T}_\beta$ is higher. So a there would likely be a wide range of K_1 gains that produce favorable e_x Bode plots. Speaking loosely, there are no contradictory objectives in the feedback law. Regardless, the procedure for tuning each controller would be the same. One would place the poles of the closed-loop transfer function to create a favorable Bode plot for e_x, given the forcing function's frequency content.

This task can be avoided though if one simply implements $\hat{T}_\beta$. If the dynamically consistent setpoint vector has already been formulated, $\hat{T}_\beta$ immediately follows. Also, when the scheme lacks feed forward, the error controller is coupled to the overall

performance. With feed forward, one can tune the error controller separately. If one needs to tune to avoid some mechanical resonance, or prominent noise frequencies, it does not have to be balanced with attenuating $B_{1a}\hat{T}_{\beta}$.

Chapter 4

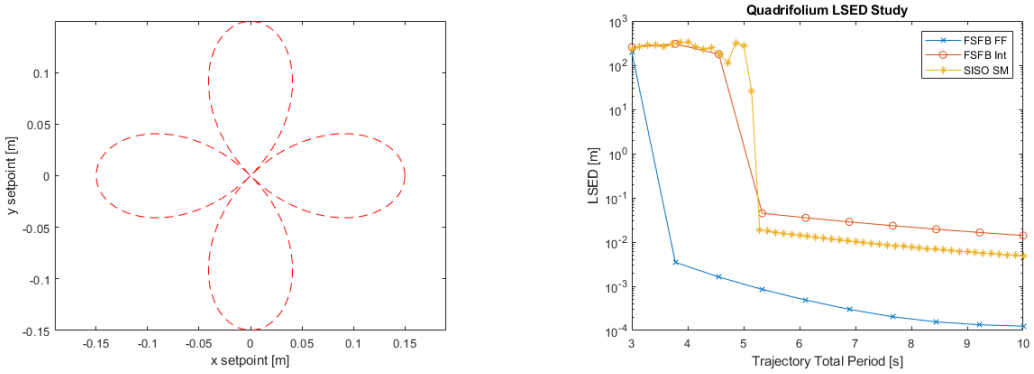
LOCK-STEP EUCLIDEAN DISTANCE STUDY

This chapter presents normalized Lock-Step Euclidean distance (2.4) studies for the controller architectures presented in Chapter 3. The same controller gains are used for these studies as were used for simulation results in that chapter. Normalized LSED measures are plotted for several rose curves across a range of periods. These allow for a more quantitative comparison between the architectures, and help solidify controller advantages/limitations discussed in the control architectures chapter. There, suitable sets of controller gains were found that, produced stable responses for some regulation, step, and quadrifolium cases. However, the solution space for the controllers is quite large. For each, there are many tuning parameters, setpoints, and initial conditions that can be tested. The studies do not present simulation data from an optimized set of gains for each controller. One could find gains that minimize the LSED over a range of setpoints and then present the LSED studies for the range, however this would be a large design challenge on its own. Rather, gains were found through manual iteration and were used as a demonstration of a typical controller response for that architecture. The LSED studies here take a similar approach, and alongside studies in Appendix A, serve as persuasive evidence for the recommendation in Chapter 5. The FSFB with feed forward controller performs better than the other two designs over a wide range of trajectories and is stable for more aggressive dynamics.

Studies were conducted for rose curves with a $r = 0.15$ $[m]$ over a T_r range of three to ten seconds. The system generally went unstable for T_r values below around three seconds - any faster and the torques would typically saturate causing failure. It was decided that the transient parts of the rose curve responses should be neglected.

Simulations were conducted for four periods of each setpoint. To get the steady state behavior, the first two periods were not included in the LSED computations. To compute the LSED, each steady state response was re-sampled at a frequency of thirty Hertz and measured relative to the setpoint evaluated at those same times.

In Chapter 3, the SISO sliding mode controller was shown to perform better than the FSFB with integral action controller, and the FSFB with feed forward was shown to perform better than both. Figure 4.1 confirms this over a wider range of quadrifolium setpoints. At about $T_r = 5.25$ $[s]$, the sliding mode and integral controllers go unstable. Those measures are orders of magnitude greater than those where stable behavior was verified earlier. The feed forward controller though, goes unstable at a significantly faster setpoint, around $T_r = 3.75$ $[s]$. As discussed in Chapter 3, the primary source of error in the feed forward controller is nonlinearity error. As expected, this increases with the frequency of the setpoint for all studies conducted.

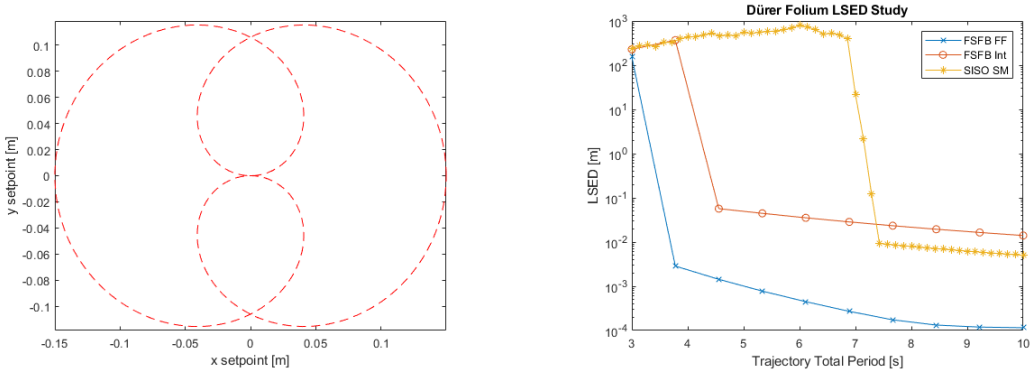


(a) Setpoint Curve (b) LSED Results

Figure 4.1: LSED Study, Quadrifolium Rose Curve, n = 2, d = 1

The Dürer folium study of Fig. 4.2 shows that the integral controller is actually able to produce more aggressive trajectories than that of the sliding mode controller. The sliding mode controller shows significantly degraded performance, only being able to

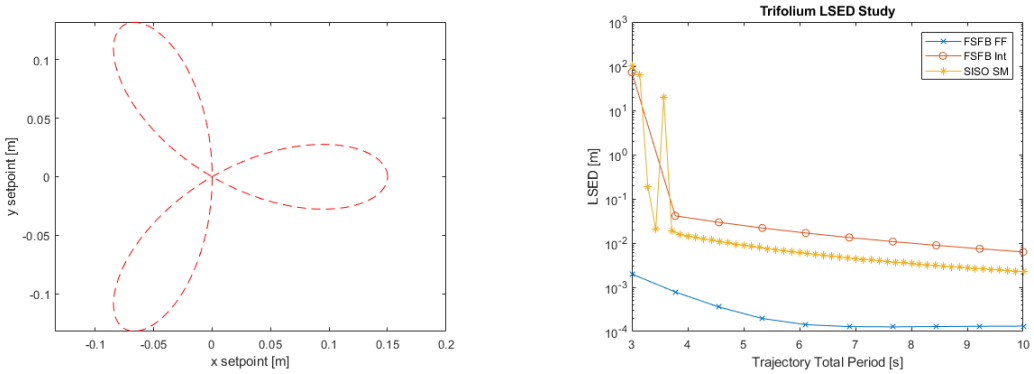
achieve relatively slow trajectories. For results where they are both stable, however, the sliding mode controller still performs better.



(a) Setpoint Curve (b) LSED Results

Figure 4.2: LSED Study, Dürer Folium Rose Curve, n = 1, d = 2

The trifolium study, Fig. 4.3, shows interesting behavior for the sliding mode controller. The controller shows some unpredictability in that it produces a stable response at one trajectory and an unstable response at a slower trajectory. This behavior may be present elsewhere, but a higher density of periods may be necessary to notice it for the other trajectories.



(a) Setpoint Curve (b) LSED Results

Figure 4.3: LSED Study, Trifolium Rose Curve, n = 3, d = 1

Chapter 5

DISCUSSION, CONCLUSION, AND FUTURE WORK

This discusses the architecture detailed in Chapters 3 and 4 in the context of concepts from the literature review, Section 1.2, and with regard to control criteria laid out in Chapter 2. The work here reiterates and collects statements about the unique behavior of each control law presented, and gives a final recommendation based on the totality of information in this thesis. This chapter also summarizes the content of this thesis, restates the final controller recommendation in this broader context, and identifies potential avenues for future work and implementation on hardware.

5.1 FSFB with Integral Action Discussion

The FSFB with Integral action controller utilizes a standard, classical, approach to trajectory-tracking control. Linear feedback on an augmented state vector (3.4) containing only error states in x and y is used to both stabilize the system and tend the system toward the desired trajectory. With regard to Tedrake's work on underactuated robotics, the control law neither seeks to cancel out the nonlinear dynamics, nor does it directly attempt to take advantage of the natural dynamics of the system. The law can be summarized as follows: add some stabilizing feedback from the angular states, i.e. introduce springiness and damping to the plate, and actuate the ball toward the desired trajectory in its own states.

Inherent in this law is phase and amplitude error, as the integrator is inadequate to supply the necessary torque to keep the system on the trajectory. In other words, this controller does not have a proof of asymptotic tracking in any of the models

presented. Additionally, modifying the phase and amplitude of the setpoint's frequency components such that the response lands on the correct trajectory is argued to be intractable and fundamentally flawed. Acceptable performance may be achieved with this approach, but tuning the controller is a difficult task. Not only may the controller need to be tuned differently for different setpoint functions, but the error dynamics and the overall performance are coupled together. If one needs to tune for noise, mechanical resonance, disturbance rejection, initial conditions, etc., this must be balanced with the steady state tracking performance.

Despite these flaws, the controller does perform well in the regulation and step cases. It also produces a qualitatively pleasing response as the angular behavior is in-sync with the ball behavior (they contain the same frequency components). With the tuning presented, the controller is able to achieve more aggressive trajectories than that of the SISO sliding mode control scheme, though it performs worse where both controllers stabilize the system. For faster trajectories, the performance is not very good, even if it produces a stable response, but for slower trajectories, it performs well enough that someone viewing the hardware in action might have a hard time telling that the setpoint is not being perfectly tracked. A final remark is that this controller is relatively easy to implement, and may prove to be the easiest to discretize and deploy on real hardware.

5.2 SISO Sliding Mode Discussion

The SISO sliding mode controller contains an internal model of the decoupled nonlinear dynamics. The law was explored in an attempt to account for modelled dynamics that are neglected in the linear controllers, but this was done at the cost of altering the objective for the controller. Because the system is underactuated, the nonlinear

input matrix, $g_1(\mathbf{x}_1)$, cannot be inverted and the state dynamics cannot be cancelled out (Tedrake [10]). However, it is found that choosing an output variable, y_1 (3.19), comprised of both x and β states allows us to formulate a control law that cancels the output dynamics. Therefore this law is in opposition to Tedrake's mantra that one should take advantage of the natural dynamics of the system. The law is shown to place the x trajectory close enough to the setpoint, in many cases, despite the fact that it is not actually the control objective.

Though there is no proof of stability in the states, simulating the SISO sliding mode control law is shown to produce relatively good performance for rose curve trajectories. Simulated responses show close to asymptotic tracking in the output variable (limited by model uncertainty), where stability is maintained, yet internal states oscillate at higher frequencies than the setpoint in order to achieve it. The result is a qualitatively unpleasant experience for the viewer. Additionally, for step and regulation cases, performance is poor. Stability with the SISO sliding mode control law in regulation and step cases is unpredictable and internal state oscillations are prominent for these types of setpoint signals.

The SISO sliding mode controller LSED results were between those of the integral controller and the feed forward controller, for stable responses. The law was least able to produce highly dynamic trajectories as internal state activity would cause the torque to saturate and the sliding surface could no longer be made attractive. Additionally, the law would be most difficult to implement as the nullifying feedback contains many functions to evaluate and discretizing the law would be non-trivial.

5.3 Recommendation: Full-State Feedback with Feed Forward

The FSFB with feed forward controller is recommended over the FSFB with integral action and SISO sliding mode controllers. It is the only control law containing a proof of asymptotic tracking in any model of the system, it shows excellent performance in simulation, outperforming the competition for a wide range of setpoint signals and setpoint frequencies, and it produces the most qualitatively pleasing visual experience when viewed in animation. Additionally, it has the advantage that the error dynamics are not directly tied to the tracking performance - as it is for the integral controller.

The control law contains a proof of asymptotic tracking in the linear model, and utilizes the natural dynamics of the system, in theme with Tedrake, to track the desired ball trajectory. To produce the necessary elements of the law, the input torque is algebraically eliminated from the equations of motion. The resulting ODE relating the ball setpoint to the angular states is solved, and the particular solution alone is used to produce a dynamically consistent setpoint vector. This solution is then substituted into the torque equation to produce the feed forward torque. With the combination of these elements, and when tuned well enough, the control law satisfies all of the control criteria from Chapter 2, and it produces excellent performance.

Performance predicted in the linear system is shown to hold even in simulations using the coupled nonlinear model. For much more dynamic trajectories than the competing laws, the responses are stable and the LSED measure is orders of magnitude lower. When viewed in animation, the responses are often visually indistinguishable from perfect tracking and produce a qualitatively pleasing visual experience. Just as with the integral controller, the angular trajectory has the same frequency components as the ball trajectory and the plate is in-sync with the ball. With this combination of

tracking performance and synchronization, the visual experience of the feed forward controller is above and beyond the integral and sliding mode controllers.

A final, major advantage of the control law, is the that the error dynamics are uncoupled from the tracking performance in the linear model[1]. The error dynamics, given by (3.46), need only be stable and decaying for asymptotic tracking to be achieved - i.e. any tuning will produce good tracking at steady state. The tuning can then be done to combat nonlinearities (like saturation or unmodelled vibration), attenuate noise, or buff out uncertain model parameters. In Section 3.3.3, the dynamically consistent setpoint vector alone is used to produce good tracking. This controller, however, does not produce uncoupled error dynamics that the feed forward torque produces.

5.4 Conclusion

The ball-and-plate system, whose dynamics are complicated, nonlinear, and underactuated was introduced in this thesis. Coupled nonlinear, decoupled nonlinear, and decoupled linear models, derived by Richter, with parameters specific to the latest iteration of the hardware, were briefly presented and used to facilitate discussion around various controller designs. The asymptotic tracking problem was defined and a trajectory similarity measure was selected to evaluate responses that do not achieve asymptotic tracking. This Lock-Step Euclidean distance measure was evaluated for many simulations in the LSED studies presented in this thesis. Though not all-encompassing, these studies supported the controller discussion and the final design recommendation. Underactuation was shown to be an incredibly important property that limited the control techniques that could be utilized.

[1]There will always be nonlinearity issues present, but simulations show that for the most part, these are inconsequential, with the exception of input saturation.

Within the limitations of the underactuation, three different controllers were explored, two derived in the linear model, and one derived in the decoupled nonlinear model. Each control architecture was motivated mathematically, and simulations were presented that highlighted important behaviors. The Full-State Feedback with Integral action and Single-Input-Single-Output Sliding Mode controllers were found to have merit, but were ultimately fundamentally limited and inferior to the Full-State Feedback with Feed Forward controller. This final controller was recommended for its proof of asymptotic tracking in the linear model, its excellent performance in simulations with the full nonlinear plant model, and its pleasing visual experience when viewed in animation. The Full-State Feedback with Feed Forward design was shown to be an excellent theoretical foundation for trajectory-tracking control on ball-and-plate hardware.

5.5 Future Work

The simulations presented in this thesis lack some important real-world considerations. The model did not include noise present in the system - a including a model of the noise would allow for tuning in simulation that more accurately represents what implementation on real hardware will entail. For the implementation, it may be necessary to introduce a filter, perhaps a Kalman Filter, to reduce noise and produce similar performance to that shown in simulation.

The control laws did not take into account disturbances. If it is desired that someone should be able to poke the ball and have the system recover from the action, more work is needed. The tuning that was conducted for this thesis did not hold well for large values of the error, and an exponential smoothing function (3.3) was found necessary. This function only smooths the system from the initial conditions to

the trajectory, and a disturbance would essentially re-introduce a set of large initial conditions. Future work would may need to include a less aggressive error controller that does not saturate or a smoothing law that reacts to disturbances.

The initial conditions used in simulation were not realistic. For the ball to start at an offset plate-frame location with the plate perfectly horizontal will not occur if the motors are turned off. With the motors turned off, the ball will slide to the edge of the plate as it tilts and hits the floor. It would then be stopped by barriers present along the edge. This barrier would need to be included in the system model if the associated initial conditions were to be properly accounted for in simulation. The hardware design by Mangin includes a manual balancing via USB interface that allows the system to start roughly at the unstable equilibrium point. Implementation of the trajectory-tracking controller would likely benefit from this initial condition and future designs should build it in.

Any controller implemented on a microcontroller will necessarily have a discrete actuation. The laws presented in this thesis do not account for this. Future work may require that either the hardware support a fast enough loop closure rate that this can be neglected, or that a discretization be brought in mathematically and evaluated further.

BIBLIOGRAPHY

[1] Jung-Hyeon Jeon and Chang-Ho Hyun. Adaptive sliding mode control of ball and plate systems for its practical application. *2nd International Conference on Control and Robotics Engineering*, 2017. doi: 10.1109/ICCRE.2017.7935054.

[2] Hongwei Liu and Yanyang Liang. Trajectory tracking sliding mode control of ball and plate system. *2nd International Asia Conference on Informatics in Control, Automation and Robotics*, 2010. doi: 10.1109/CAR15979.2010.

[3] Scott Mangin. Development of a prototype ball-and-plate balancing platform. Master's thesis, California Polytechnic State University, San Luis Obispo, 2022.

[4] Amin Mohammadi and Ji-Chul Ryu. Neural network-based pid compensation for nonlinear systems: ball-on-plate example. *International Journal of Dynamics and Control*, 2020. doi: 10.1007/s40435-018-0480-5.

[5] Mario Ramírez-Neria, Hebertt Sira-Ramírez, Rubén Garrido-Moctezuma, and Alberto Luviano-Juárez. On the linear control of underactuated nonlinear systems via tangent flatness and active disturbance rejection control: The case of the ball and beam system. *Journal of Dynamic Systems, Measurement, and Control*, 2016. doi: 10.1115/1.4033313.

[6] Zachary Richter. Nonlinear model development and validation for ball and plate control system. Master's thesis, California Polytechnic State University, San Luis Obispo, 2021.

[7] Dan Simon. System stability. https://academic.csuohio.edu/simond/linearsystems/stability/, 08 2002. Accessed: 2010-09-30.

[8] Jean-Jacques E. Slotline and Weiping Li. *Applied Nonlinear Control*. Prentice Hall, 1991.

[9] Yaguang Tao, Alan Both, Rodrigo Silveira, Kevin Buchin, Stef Sijben, Ross Purves, Patrick Laube, Dongliang Peng, Kevin Toohey, and Matt Duckham. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58:1–27, 06 2021. doi: 10.1080/15481603.2021.1908927.

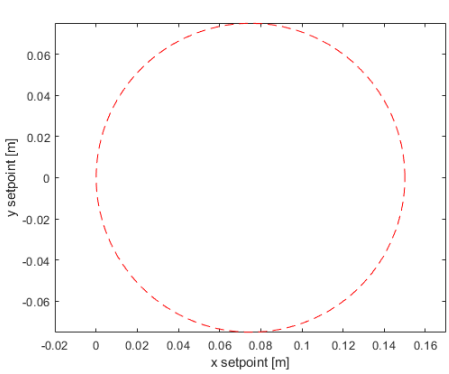
[10] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation*, chapter 1. 2009. URL https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-832-underactuated-robotics-spring-2009/readings/MIT6_832s09_read_ch01.pdf. Course Notes for MIT 6.832.

APPENDICES

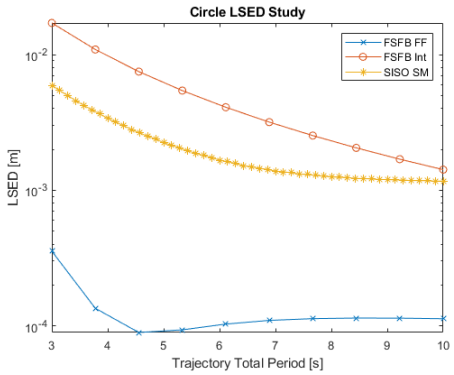
Appendix A

ADDITIONAL LSED STUDIES

This appendix gives additional Lock-Step Euclidean Distance study data for rose curves not presented in the main body of this thesis.

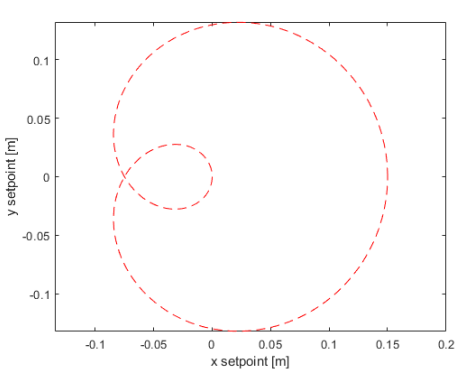


(a) Setpoint Curve

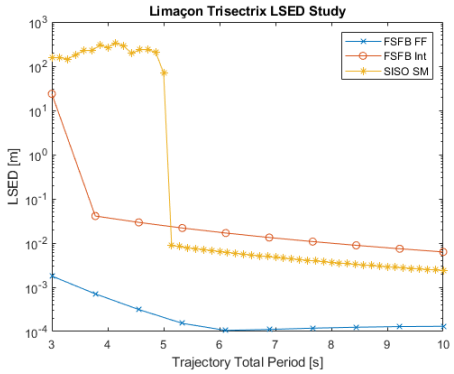


(b) LSED Results

Figure A.1: LSED Study, Circle Rose Curve, n = 1, d = 1

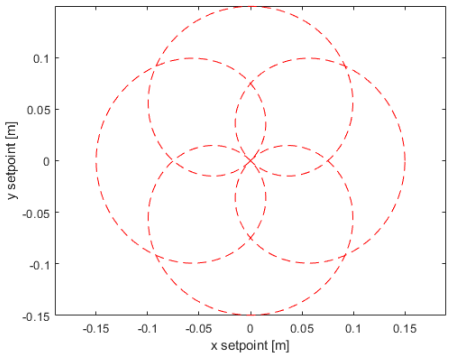


(a) Setpoint Curve

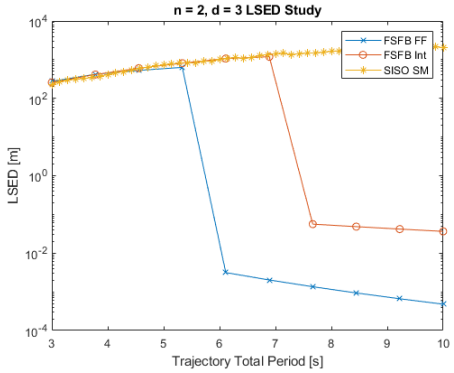


(b) LSED Results

Figure A.2: LSED Study, Limaçon Trisectrix Rose Curve, n = 1, d = 3

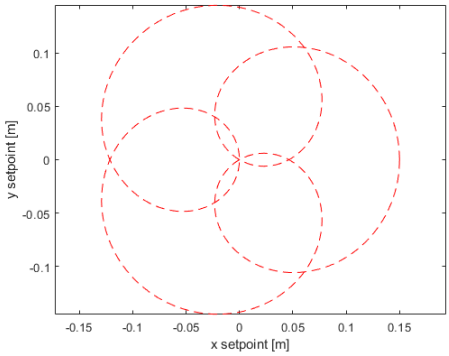


(a) Setpoint Curve

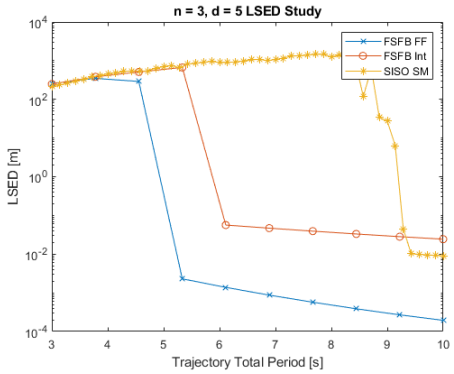


(b) LSED Results

Figure A.3: LSED Study, Rose Curve, n = 2, d = 3



(a) Setpoint Curve



(b) LSED Results

Figure A.4: LSED Study, Rose Curve, n = 3, d = 5