Imperial College London

Department of Aeronautics

Swing up Control of Very Flexible Pendulums

Author:
Devansh Ramgopal Agrawal

Supervisor:
Prof. Andrew Wynn

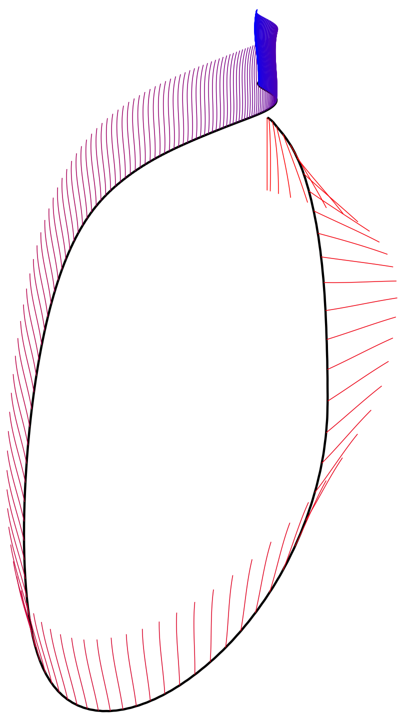
A thesis submitted for the degree of

Aeronautical Engineering with a Year Abroad (MEng)

15 June 2020

Abstract

The swing up control of a 3D cart-pole with a very flexible beam is demonstrated using a geometrically exact beam formulation. A novel control approach exploiting the inherent dynamics of rotating systems is used to achieve the full, stabilised swing up control. The controller's efficacy is first demonstrated on a rigid pendulum, and then applied to the flexible pendulum case.



Acknowledgements

I would like to thank my advisor Prof Wynn and his student Marc Artola for their insight and guidance on this project. Marc also wrote the bulk of the simulator that my work builds on, and thus I thank him for freely sharing this with me and taking the time to explain it. I would also like to thank my family for supporting me from the very beginning. Finally I must thank the midnight cup of coffee that allowed me to finish this paper on time.

Contents

List of Figures

List of Tables

Nomenclature

$\boldsymbol{\omega}$	angular velocity
$\boldsymbol{\omega}(s,t)$	Angular velocity at each section of the beam at any given time
$\boldsymbol{\phi}_{1i}(s)$	i-th mode shape of $\boldsymbol{x}_1$
$\boldsymbol{\phi}_{2i}(s)$	i-th mode shape of $\boldsymbol{x}_2$
$\boldsymbol{\tau}$	Control Torque
$\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$	Local frame of reference
$\boldsymbol{f}(s,t)$	Sectional force resultant at each section of the beam at any given time
$\boldsymbol{f}_g(s)$	External sectional forces and moments due to gravity
$\boldsymbol{g}$	Gravitational force vector in the inertial frame
$\boldsymbol{I}$	Moment of inertia
$\boldsymbol{m}(s,t)$	Sectional moment resultant at each section of the beam at any given time
$\boldsymbol{q}(t)$	Modal coordinates
$\boldsymbol{r}_{cm}(s)$	Offset between the sectional centre of mass and the elastic axis
$\boldsymbol{v}(s,t)$	Linear velocity at each section of the beam at any given time
$\boldsymbol{x}$	State vector
$\boldsymbol{x}_1(s,t)$	Linear and angular velocities at each section of the beam at any given time
$\boldsymbol{x}_2(s,t)$	Resultant sectional forces and moments at each section of the beam at any given time
δq	Difference between current and reference augmented state vector
δu	Difference between current and reference control vector
δ_{ij}	Kronecker delta, $\delta_{ij} = 1$ if $i = k$, $\delta_{ij} = 0$ if $i \neq k$
γ	Strain vetor
κ	Vector of curvatures and twist
κ_0	Initial curvature and pre-twist
${\mathbb{R}_+}^d$	d-dimensional space of positive Reals
$\mathbb{R}^d$	d-dimensional space of Reals
$\mathcal{L}_1$	Linear operator on $\boldsymbol{x}_1$

$\mathcal{L}_2$	Linear operator on $\boldsymbol{x}_2$
$\mathcal{U}$	skew-symmetric operator
ω_i	natural frequency of i-th mode shape
ρI_1	Polar Moment of inertia per unit length
ρI_2	Moment of inertia about 2-axis per unit length
ρI_3	Moment of inertia about 3-axis per unit length
ρ	Density
$\tilde{\boldsymbol{a}}$	skew symmetric matrix of $\boldsymbol{a}$ such that $\boldsymbol{a} \times \boldsymbol{b} = \tilde{\boldsymbol{a}}\boldsymbol{b}$
A	Cross sectional area
C	Compliance Matrix
c	controller strength constant
C_τ	Damping time scales matrix
C_d	$C_d = C_\tau C^{-1}$
D_0, D_1, D_2	Coefficient matrices for linearised damping
E	Matrix containing the initial curvature and pre-twist
E_d	Energy Deficit - the difference between the current energy and the energy at the target state
EA	Direct tension stiffness
EI_2	Bending stiffness about 2-axis
EI_3	Bending stiffness about 3-axis
f_o	Weighting on $Q_{orientation}$ in hybrid cost functions
f_s	Weighting on $Q_{stabilising}$ in hybrid cost functions
GA_2	Shear stiffness in 2-axis
GA_3	Shear stiffness in 3-axis
GJ	Torsional stiffness
I_n	$n \times n$ identity matrix
M	Mass matrix
N_m	Number of mode shapes
P	Cost weight matrix on terminal state
Q	Cost weight matrix on states
$q_{1i}, q_{2,i}$	Modal coordinates of the finite dimensional approximation
R	Cost weight matrix on control input
s	Curvilinear coordinate that defines the elastic axis of the beam

s_∂	Boundary of the beam, s=0 and s=L
$SO(3)$	3D rotation group
t	Time

Chapter 1

Introduction

The design of future, high-endurance, low-emissions aircrafts can greatly benefit from the use of lightweight structures and wings with high aspect ratios. Such vehicles are therefore highly structural flexible. With such structural flexibility and large aerodynamic loads expected from high-performing aerosurfaces, the traditional separation of aero-elastic analysis from the flight dynamics is not suitable. Furthermore, these systems are highly non-linear, and greatly coupled. Such systems present a great challenge to the aircraft designer, as suitable tools for the analysis, design, optimisation and control of these systems do not exist.

The need to develop "more advanced, multidisciplinary ... time-domain analysis methods appropriate to highly flexible, morphing vehicles" was called for in the wake of NASA's Helios mishap [1]. In particular, analysing the interaction and developing suitable control methods for such morphing vehicles provides the motivation for this project.

Flexible structures are also of great relevance to a number of other areas: for instance, the fields of soft robotics, and soft manipulators are seeking similar tools to, for instance, be able to control a soft-elastic tendon for a robotic leg, or to morph the body to wrap around and grasp objects [2, 3]. Biomedical research into the design and actuation of robotic, steerable needles also requires controller design for safe operation [4, 5].

In this project, the swing up control of a highly flexible pendulum is attempted. While rigid pendulums have been very well studied, a flexible beam acting as a pendulum is the simplest structure that can be studied to gain insight into appropriate control methods for flexible structures.

1.1 Objectives

The design of a controller that can efficiently swing up a highly flexible beam pendulum, and stabilise it in an upright position is the main objective of this work. The problem is a 3D, flexible beam extension to the traditional cart-pole problem.

The focus of the research is on understanding the implications of an alternative approach in the design of a controller, and to evaluate its suitability for the control of very flexible structures.

1.2 Contributions

In this paper a non-linear, geometrically exact beam formulation is used to reduce the structure's state into a low-dimensional state. From there a novel cost function that exploits the natural dynamics of a structure is used to define an optimisation problem that is solved using Model Predictive Control (MPC).

I demonstrate that this novel cost function is capable of uprighting a flexible beam, and that it presents a few desirable properties, including ease of implementation, and extendability to more complicated structures.

1.3 Organisation

This thesis is organised as follows: In Chapter 2, some background is provided as to the current literature on the structural modelling and control of flexible structures is provided, and the reasons for choosing the the current approach are discussed. In Chapter 3, quaternion dynamics is introduced, and a new controller for quaternions is identified. The behaviour of this controller is studied using a rigid simple pendulum and provides motivation to apply this controller to flexible beams. Chapter 4 defines the beam model used in this control problem. We bring these together in Chapter 5, where the novel control scheme is applied to the flexible beam and new cost functions are defined. Finally Chapter 6 presents the findings of the paper, through numerical simulations and analysis of the behaviour of the beams. In the conclusion, recommendations are made for future research directions.

Chapter 2

Background

2.1 Energy Shaping and Psuedo-Rigid-Body Paradigm

The simple, rigid pendulum has been extensively studied, and many different control approaches exist to control it, and the related family of problems: the acrobot, cart-pole, quadrotors etc. These include energy shaping [6], partial feedback linearisation [7] and related methods that define a feedback controller based on the difference between the current 'energy' of the system and the desired energy. The word 'energy' is often used loosely to refer to some scalar quantity that can be used in this way to define a controller. While simple and often effective, they are often hard to design for complicated, coupled systems.

The energy shaping methods can be extended to multi-link pendulums easily [8] and thus many such controllers are defined by energy based methods. For instance for the control of a flexible needle as it is to pierce skin, Franco [4] uses an energy based approach - the slender beam is modeled as a two-rigid-link structure with a spring to model the deflection of the beam, the so-called psuedo-rigid-body paradigm [9]. While this offers a method to reduce the need for complicated Finite Element Analysis (FEA) on compliant mechanism, the model does lose accuracy on the beam dynamics.

Only a few flexible inverted pendulum control schemes have been published, but generally apply ideas from the flexible manipulators [10] including Assumed Modes (mode shapes based on Euler-Bernoulli or Timoshenko beam models for simplifed beam properties) or finite element based approximations of the mode shapes. Gandhi [11] used these a modified version of these models to produce an energy shaping based controller with a PID loop to stabilise a tip mass on a flexible pendulum. His methods did not allow for full swing up control as they were linearised about the upright position.

2.2 Modal-Based Models

Recently, there has been interest in developing richer, geometrically exact models of flexible structures and to study the control of such systems. These approaches were first identified by Simo et al [12], allowing for energy preserving description of the beams in a partial differential equation. To allow for control using methods appropriate for vibrations, these equations would need to be written in a modal coordinates. Palacios [13] showed the equations could be written exactly with only quadratic nonlinearity if written using the intrinsic form - using velocities and sectional forces as primary variables rather than the displacements and rotations often used. The control of these beams could then be written in a port-hamiltonian form, making it amenable to rigid-body control methods. The modal formulation has been used to demonstrate low-computational cost MPC control of a flexible structures [14].

Since this approach is geometrically exact, it allows structures to be accurately modelled over a wide range of operating conditions using only a small number parameters relevant to the control problem. For example, an interpolation scheme was developed to smoothly bridge the structural description over multiple operating regions [15].

Recently, Artola [16] demonstrated the stabilisation of a flexible inverted pendulum described using the modal analysis as above. He employed a MPC scheme to determine the optimal control input, as it could reduce the time horizon that the optimiser had to operate in, and because he could introduce perturbations during a simulation and determine the robustness.

This work extends Artola's model applying a novel controller (defined in Chapter 3) to achieve full swing up of the pendulum.

Chapter 3

Quaternion Dynamics and a Novel Controller

In this chapter, the dynamics of a rigid body with rotation described by quaternions is examined. An observation is made on the evolution of the quaternions, and thus a rigid body controller that has useful properties is defined. The primary purpose of this chapter is to provide motivation for the flexible pendulum controllers defined in Chapter 5.

3.1 Quaternion Dynamics

The quaternion $\boldsymbol{\xi}(t) = [\xi_0, \boldsymbol{\xi}_v^T]^T : \mathbb{R}_+ \to \mathbb{R}^4$ where $\xi_0 \in [-1, 1]$ and $||\boldsymbol{\xi}_v|| \in [0, 1]$ is used to parameterise rotations. These describe a rotation of θ radians about the unit vector $\boldsymbol{l}$,

$$\xi_0 = \cos(\theta/2) \tag{3.1}$$

$$\boldsymbol{\xi}_v = \sin(\theta/2)\boldsymbol{l} \tag{3.2}$$

and therefore we must have

$$||\boldsymbol{\xi}||^2 = \xi_0^2 + ||\boldsymbol{\xi}_v||^2 = 1 \tag{3.3}$$

This rotation can also be described by a rotation matrix, $T \in SO(3)$, [17, Ch.20]

$$T(\boldsymbol{\xi}) = (1 - 2||\boldsymbol{\xi}_v||^2)I_3 + 2\boldsymbol{\xi}_v\boldsymbol{\xi}_v^T + 2\xi_0\tilde{\boldsymbol{\xi}}_v, \tag{3.4}$$

where $\tilde{\boldsymbol{a}}$ denotes the skew-symmetric matrix of $\boldsymbol{a}$ such that $\tilde{\boldsymbol{a}}\boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}$, and I_3 is the 3×3 identity matrix. The evolution of the quaternion is given by

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\tilde{\boldsymbol{\omega}} \end{bmatrix}\begin{bmatrix} \xi_0 \\ \boldsymbol{\xi}_v \end{bmatrix} \tag{3.5}$$

where $\boldsymbol{\omega}(t) : \mathbb{R}_+ \to \mathbb{R}^3$ is the angular velocity of the body.

3.2 Insight

Suppose we are able to construct a controller that satisfies for all time,

$$\boldsymbol{\xi}_v = -\frac{1}{c}\boldsymbol{\omega}, \tag{3.6}$$

where $c \in \mathbb{R}_+$ is a positive scalar constant. Then using Equation (3.5) we have,

$$\frac{\partial \xi_0}{\partial t} = -\frac{1}{2}(-c\boldsymbol{\xi}_v)^T\boldsymbol{\xi}_v = \frac{c}{2}\boldsymbol{\xi}_v^T\boldsymbol{\xi}_v \tag{3.7}$$

and

$$\frac{\partial \boldsymbol{\xi}_v}{\partial t} = -\frac{1}{2}c\xi_0\boldsymbol{\xi}_v + \frac{1}{2}c\boldsymbol{\xi}_v \times \boldsymbol{\xi}_v \tag{3.8}$$

$$= -\frac{c}{2}\xi_0\boldsymbol{\xi}_v \tag{3.9}$$

$$\therefore \frac{\partial ||\boldsymbol{\xi}_v||_2^2}{\partial t} = 2\boldsymbol{\xi}_v^T\frac{\partial \boldsymbol{\xi}_v}{\partial t} \tag{3.10}$$

$$= 2\boldsymbol{\xi}_v^T\left(-\frac{c}{2}\xi_0\boldsymbol{\xi}_v\right) \tag{3.11}$$

$$= -c\xi_0||\boldsymbol{\xi}_v||_2^2 \tag{3.12}$$

Inspecting these equations, we can see that $\frac{\partial \xi_0}{\partial t} > 0$ if $c > 0$. Furthermore, if $c\xi_0 > 0$, we have $\frac{\partial ||\boldsymbol{\xi}_v||^2}{\partial t} < 0$. This suggests that regardless of the starting state, ξ_0 must increase, and therefore will become positive at some point, and tend to its maximum value of 1. Once $\xi_0 > 0$, we must have that $||\boldsymbol{\xi}_v||$ decreases. Since $||\boldsymbol{\xi}_v|| \in [0, 1]$, we can say

$$\lim_{t\to\infty} \boldsymbol{\xi}(t) = [1, 0, 0, 0]^T \tag{3.13}$$

Therefore, if a controller allows a body to satisfy Equation (3.6), the controller will naturally drive the quaternion of the body towards the state where the transformation matrix is the identity matrix,

$$\lim_{t\to\infty} T(\boldsymbol{\xi}(t)) = I_3 \tag{3.14}$$

This is key result motivates the rest of this work. In the next section, the behaviour of this controller is analysed in more detail, later it is applied to the swing-up control of a rigid pendulum.

3.3 Analytic Controller for Freely Rotating Rigid Bodies

To understand the implications of Equation (3.6), imagine a freely rotating rigid body in 3D. Assuming the linear motion is decoupled (and thus omitted), and the only torque acting on the body is due to an externally applied control torque $\boldsymbol{\tau}(t) : \mathbb{R} \to \mathbb{R}^3$. The dynamics of the body (in body reference frame) using the state $\boldsymbol{x} = [\boldsymbol{\xi}^T, \boldsymbol{\omega}^T]^T$ are

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\tilde{\boldsymbol{\omega}} \end{bmatrix}\begin{bmatrix} \xi_0 \\ \boldsymbol{\xi}_v \end{bmatrix} \tag{3.15}$$

$$\boldsymbol{I}\frac{\partial \boldsymbol{\omega}}{\partial t} + \boldsymbol{\omega} \times (\boldsymbol{I}\boldsymbol{\omega}) = \boldsymbol{\tau} \tag{3.16}$$

where $\boldsymbol{I} \in S_{++}^3$ is a symmetric matrix describing the moments of inertia in the body reference frame. Expanding Equation (3.7),

$$\frac{\partial \xi_0}{\partial t} = \frac{c}{2}\boldsymbol{\xi}_v^T\boldsymbol{\xi}_v = \frac{c}{2}(1 - \xi_0^2) \tag{3.17}$$

we arrive at a decoupled ordinary differential equation in ξ_0, that can be solved to give

$$\xi_0(t) = \tanh\left(\frac{ct}{2} - c_1\right) \tag{3.18}$$

Substituting this into Equation (3.9), we have

$$\frac{\partial \boldsymbol{\xi}_v}{\partial t} = -\frac{c}{2}\left[\tanh\left(\frac{ct}{2} - c_1\right)\right]\boldsymbol{\xi}_v \tag{3.19}$$

which is a decoupled non-linear ODE in each dimension of ξ_v. We can solve for this analytically too, and using Equation (3.6) and Equation (3.16) solve for the angular velocity and required torque as functions of time,

$$\boldsymbol{\xi}_v(t) = \operatorname{sech}\left(\frac{ct}{2} - c_1\right)\boldsymbol{c}_2 \tag{3.20}$$

$$\boldsymbol{\omega}(t) = -c\operatorname{sech}\left(\frac{ct}{2} - c_1\right)\boldsymbol{c}_2 \tag{3.21}$$

$$\boldsymbol{\tau}(t) = \frac{c^2}{1+\cosh(ct-2c_1)}\left(2\boldsymbol{c}_2 \times (\boldsymbol{I}\boldsymbol{c}_2) + \sinh\left(\frac{ct}{2} - c_1\right)\boldsymbol{I}\boldsymbol{c}_2\right) \tag{3.22}$$

where

$$c_1 = -\operatorname{arctanh}\xi_0(0) \tag{3.23}$$

$$\boldsymbol{c}_2 = \frac{\boldsymbol{\xi}_v(0)}{||\boldsymbol{\xi}_v(0)||} \tag{3.24}$$

are determined from the initial values of $\boldsymbol{\xi}$. Note, $\boldsymbol{\omega}(0) = -c\boldsymbol{\xi}_v(0)$ is a fixed parameter based on the choice of c.

These solutions provide interesting insights:

1. Notice that the controller strength parameter c only appears in the evolution of ξ as ct, and therefore c can be interpreted as the inverse of a time scale for the quaternions.
2. We can bound the maximum torque needed for such a controller to be

$$||\boldsymbol{\tau}||_{\max} \leq c^2\left|\left|\boldsymbol{c}_2 \times (\boldsymbol{I}\boldsymbol{c}_2) + \frac{1}{4}\boldsymbol{I}\boldsymbol{c}_2\right|\right| \leq \left(\frac{2+\sqrt{5}}{4}\right)c^2||\boldsymbol{I}|| \approx 1.06c^2||\boldsymbol{I}|| \tag{3.25}$$

   The maximum torque required is proportional to c^2, allowing us to pick c based on torque limits.[1] Note, the latter bound is much looser than the former for most $\boldsymbol{c}_2, \boldsymbol{I}$

3. Without any external forces, the controller maintains the angular velocity and $\boldsymbol{\xi}_v$ about their respective initial axis. While this makes the rotation effectively 2D, the direction and magnitude of the torques (in the body frame) will change over time.
4. While all states should bring the body to the desired final state, the controller struggles near $\xi_0 = -1$ and $\boldsymbol{\xi}_v = [0,0,0]$. This is because $T(\boldsymbol{\xi}) = T(-\boldsymbol{\xi})$, that is, a quaternion and its negative represent the same rotation. [17]. However the controller is unable to see this, and could cause a full rotation rather than stabilising the point (as in the first two rows of Figure 3.1). This issue is easily solved if we account for this while specifying the initial conditions, taking care to express them in the form closer to the $[1,0,0,0]$ target point.

   At $\boldsymbol{\xi} = [-1,0,0,0]$, the controller would take an infinite amount of time to reach the target state. For a small perturbation (for example $\xi_0(0) = -0.99$), the controller reaches the target in $\sim 10/c$ seconds (Figure 3.1).

5. The time taken to for the quaternion's scalar component to be greater than some threshold ($\xi_0(t) \geq \hat{\xi}_0$) can be found analytically,

$$t = \frac{2\left(\operatorname{arctanh}(\hat{\xi}_0) + c_1\right)}{c} \tag{3.26}$$

   and for $\hat{\xi}_0 = 0.9$ we have $t \geq (2.944 + 2c_1)/c$.

The response due to this controller is shown in Figure 3.1 from 5 different starting states. $\boldsymbol{\xi}$, for $c = 1$. We can see that despite no information on the target state being provided to the controller, it can navigate to the expected state.

3.4 Swing up control of rigid pendulums

While the previous section provides analytical results on the behaviour of this controller, it (a) neglects any external forces (b) does not allow for $\boldsymbol{\omega}$ to deviate from the prescribed value based on c. In this section, a toy problem of a 2D pendulum will be used to study this controller better.

[1]This simplification is possible since $||\boldsymbol{c}_2|| = 1$

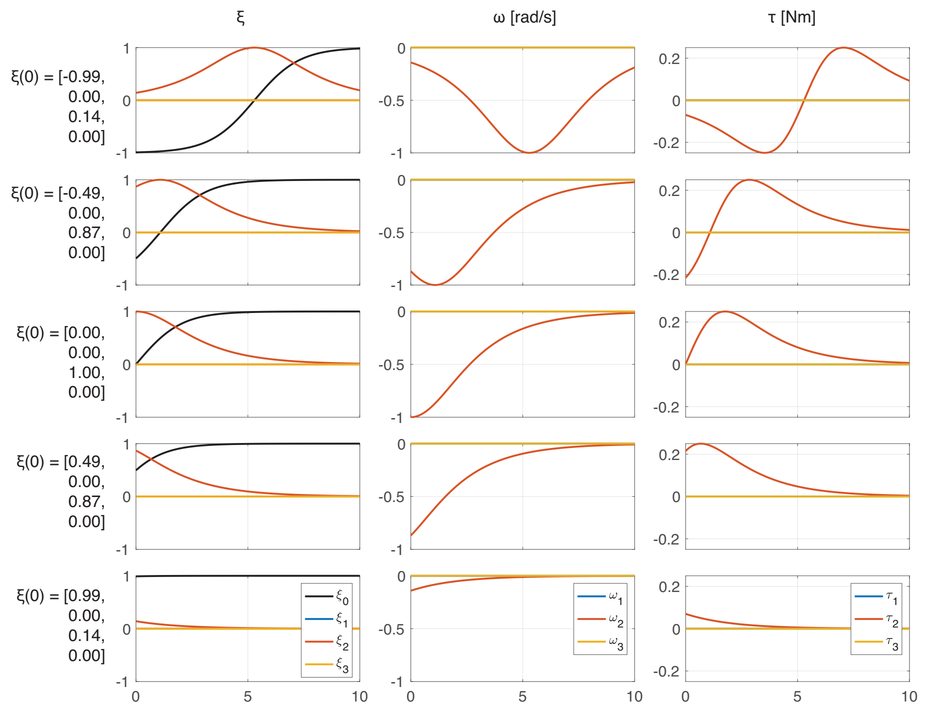


Figure 3.1: Analytic Controller. Each row represents the solution from a different starting quaternion, indicated on the left. The columns show the quaternion components, the angular velocity and the torque applied.

3.4.1 Problem definition

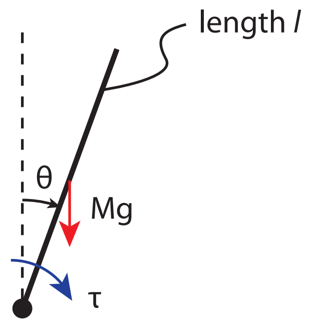


Figure 3.2: Definition of parameters for rigid pendulum.

Consider a rigid beam pendulum, with length $L = 1$ m, total mass $M = 1$ kg which is uniformly distributed across the beam. It starts in the downward stable position, $(\theta = \pi)$, and the target is to reach $\theta = 0$ using a torque at the pivot point τ. Gravitational acceleration $g_0 = 9.81$ m/s.

The dynamics are therefore given by,

$$\frac{\partial \theta}{\partial t} = \omega \tag{3.27}$$

$$\frac{\partial \omega}{\partial t} = \frac{\tau + (M g_0 L/2)\sin\theta}{\frac{ML^2}{3}} = \frac{3\tau}{ML^2} + \frac{3g_0}{2L}\sin\theta = 3\tau + \frac{3g_0}{2}\sin\theta \tag{3.28}$$

3.4.2 Energy Shaping

The energy of the pendulum is given by

$$E = \frac{1}{6}\omega^2 + \frac{g_0}{2}\cos\theta \tag{3.29}$$

and defining the difference between the current and target energy as $E_d = E - g_0/2$, the rate of change of the energy deficit is

$$\dot{E}_d = \frac{1}{3}\omega\dot{\omega} - \frac{g_0}{2}\sin\theta\dot{\theta} = \omega\tau \tag{3.30}$$

therefore, if

$$\tau(t) = -k\omega E_d = -k\omega\left(\frac{1}{6}\omega^2 + \frac{g_0}{2}(\cos\theta - 1)\right) \tag{3.31}$$

$$\dot{E}_d = -k\omega^2 E_d \tag{3.32}$$

the controller will bring the energy deficit to asymptotically to 0. The closed loop dynamics under this controller is shown in Figure 3.3 for $k = 0.1$. However there a few issues with this controller:

1. The controller is not stabilising. The fixed point in the upright condition is attractive, but not stable. This is because the Lyapunov function (E_d) as stated above is not positive for all (θ, ω) not at the fixed point. The fixed point is attractive in that the controller will always add energy when the system energy is lower than the target set point, and remove energy in when system energy is higher than it should be. Since there is only one fixed point where the energy deficit is 0, the system with the energy shaping controller is attracted to this point.
2. An example of this instability is seen if we imagine a state near the equilibrium but with positive ω and negative θ. This situation would (under open loop dynamics) move towards to fixed point. However the torque can be negative! Combined with the fact that the controller is unstable, there are situations in which the pendulum will go all the way around instead of being stabilised near the upright condition. As such, a second controller (often a Linear Quadratic Regulator) is used near the equilibrium point [6].
3. The initial condition $(\theta, \omega) = (\pi, 0)$ is stable, and under this controller, no torque will be applied. As such, for it to start swinging, a small perturbation must be provided.

3.4.3 Orientation Based Controller

To define a controller that tracks Equation (3.6), we define a optimal control problem,

$$J = \underset{\tau(t)}{\text{minimize}} \quad \int_0^{t_f} \left|\left|\boldsymbol{\xi}_v + \frac{\boldsymbol{\omega}}{c}\right|\right|^2 dt \tag{3.33a}$$

This controller was chosen as it would minimise the time averaged error in $\boldsymbol{\xi}_v = -\boldsymbol{\omega}/c$. However this can also cause it to act against the natural dynamics of the problem, unlike the energy shaping methods which seek to slightly modify the dynamics of the system to allow them to reach the target state. There may be other ways to encode the desired behaviour, for instance

$$J = \int_0^{t_f} \Psi(t)\left|\left|\boldsymbol{\xi}_v + \frac{\boldsymbol{\omega}}{c}\right|\right|^2 + \boldsymbol{\tau}^T R\boldsymbol{\tau} dt \tag{3.34}$$

where $\Psi(t)$ is weighting function that places greater weight at a future time (for example, $\Psi(t) = \tanh(t/t_f)$ or $\Psi(t) = t^2$). In this way, the controller might use the initial time to place the state on a trajectory that will be close to satisfying $\boldsymbol{\xi}_v = -\boldsymbol{\omega}/c$ while not trying to fight the dynamics. In this work however, the behaviour of Equation (3.33) was the focus, and thus the choice and impact of Ψ is left for future research.

For a 2D problem, we can simplify Equation (3.33) to

$$J = \underset{\tau(t)}{\text{minimize}} \quad \int_0^{t_f} \left(\sin\left(\frac{\theta}{2}\right) + \frac{\omega}{c}\right)^2 dt \tag{3.35a}$$

$$\text{subject to} \quad \text{dynamics,} \tag{3.35b}$$

$$\theta(0) = \pi, \omega(0) = 0 \tag{3.35c}$$

This non-trivial optimisation problem must be solved numerically (using a psuedo-spectral method [18]), and results for various c are shown in Figure 3.3. Full implementation details are listed in the appendix.

3.4.4 An appropriate value for the controller strength, c

There is one particular value of c that yields interesting results. Suppose $c = \sqrt{6g_0}$. In this case, if our controller was perfect, we could write

$$\begin{aligned}
\omega &= -\sqrt{6g_0}\sin\frac{\theta}{2} && (3.36)\\
\therefore \dot{\omega} &= -\sqrt{6g_0}\cos\frac{\theta}{2}\frac{\dot{\theta}}{2} && (3.37)\\
&= -\frac{\sqrt{6g_0}}{2}\cos\frac{\theta}{2}\left(-\sqrt{6g_0}\sin\frac{\theta}{2}\right) && (3.38)\\
&= \frac{3g_0}{2}\sin\theta && (3.39)
\end{aligned}$$

which is exactly the dynamics of the pendulum! Using $c = \sqrt{6g_0} \approx 7.67$, we have defined the path as that which has the same energy as the target state, and thus the new controller can suggest the same control law as that from energy shaping.

We can also show that near the upright position, the LQR results based on the new controller (referred to as the orientation based controller) match those from a torque minimising controller using the weights,

$$Q_{torque} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R_{torque} = 10 \tag{3.40}$$

$$Q_{orientation}(c) = \begin{bmatrix} 1/4 & 1/(2c) \\ 1/(2c) & 1/c^2 \end{bmatrix} \qquad R_{orientation} = 10^{-10} \tag{3.41}$$

where $Q_{orientation}$ is the linearised version of Equation (3.35) and the small cost on R was necessary since LQR algorithms are ill-defined for singular R. The LQR problem was solved using `Mathematica`, for various c, and the weights of each controller ($\tau = -\boldsymbol{K}\left[\theta, \omega\right]^T$)is listed below.

Table 3.1: Feedback controller weights for various types of controllers

Type	$K_1/\|\|\boldsymbol{K}\|\|$	$K_2/\|\|\boldsymbol{K}\|\|$
Torque	0.967223	0.253939
Orientation, $c = \sqrt{6g_0}$	0.967660	0.252257
Orientation, $c = 1$	0.447248	0.894410
Orientation, $c = 10$	0.980578	0.196129
Orientation, $c = 100$	0.997779	0.066118

These results show that the orientation based controller for $c = \sqrt{6g_0}$ has exactly almost exactly the same behaviour as the LQR applied at the upright position.

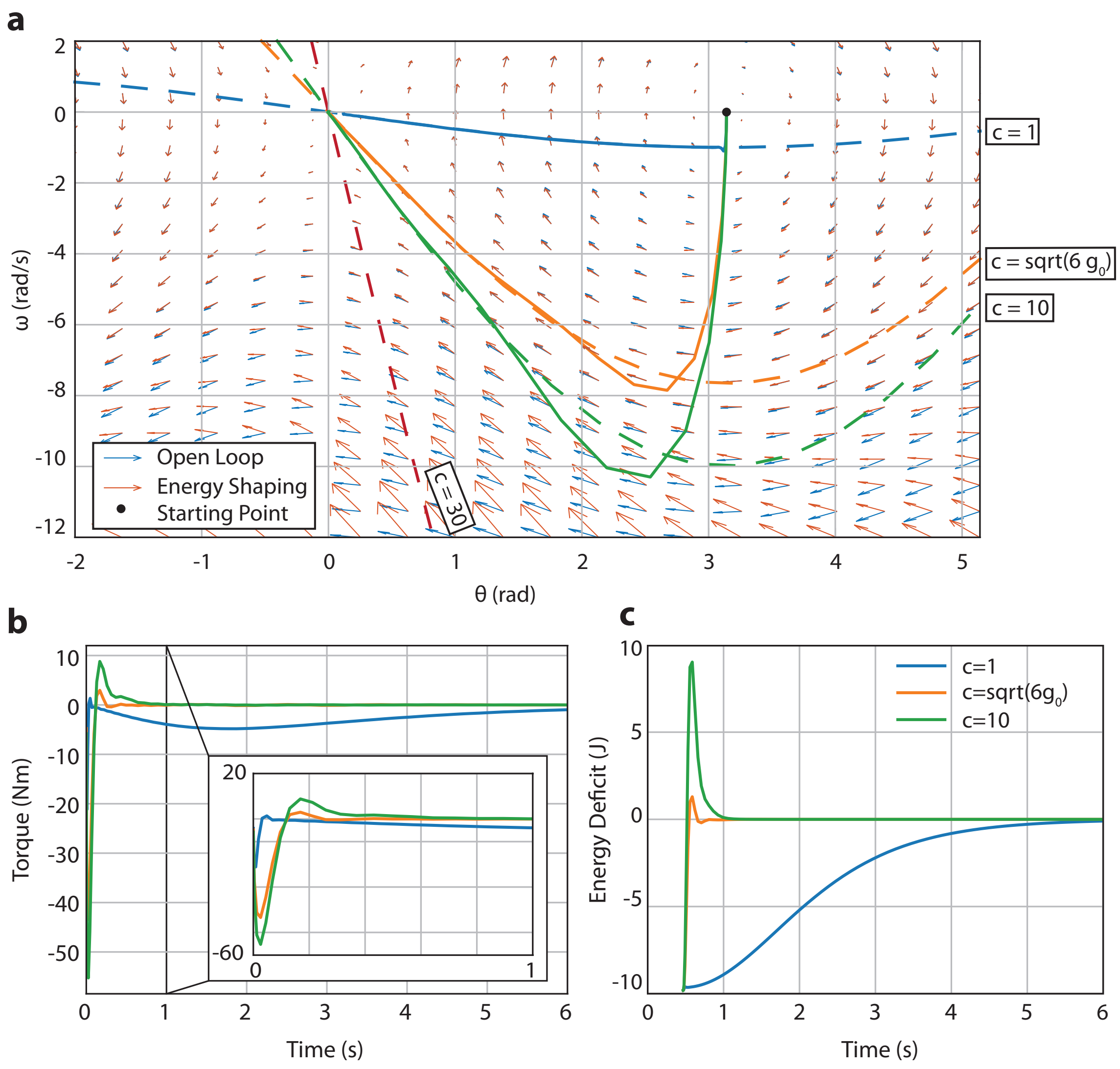


Figure 3.3: Rigid Pendulum Dynamics. (a) State space diagram for a simple pendulum showing the open-loop controller (blue) and the closed loop control for energy shaping (red). Note, $\theta = 0$ for the upright situation. Dashed lines show solutions to $\sin\theta/2 + \omega/c = 0$ for various c. Solid lines indicate solutions of the optimal control problem Equation (3.35) for corresponding c. (b) Control history for these solutions. Inset shows the initial area at higher magnification. (c) History of energy deficit for these solutions.

Chapter 4

Flexible Beams

The control of highly flexible structures is challenging, not only because the physical models and dynamics are far more complicated, but because the size of the problem to describe these flexible structures is necessarily much greater. A simple rigid pendulum can be described by two states: θ, ω and only a few parameters are needed to fully describe its behaviour: M, L, I, g_0. An equivalent flexible beam, as will be described in much greater detail in the next section, requires knowledge of these states at each position along the beam's length.

Furthermore, classical beam theories like the Euler-Bernoulli beam theory or the Timoshenko flexible beam theory are not geometrically exact, and fail to accurately describe the beams behaviour for large deformations. Here I use geometrically exact non-linear beam theory described by [16] to capture the deformations and dynamics precisely. The theory allows us to account for a few key effects that arise from the interaction of rigid body motion, axial deformations, beam bending, and twisting: (1) the geometrically stiffening effect, where the deformed structure is stiffer due to the reorientation of stress fields to the in-plane directions (2) changes of global inertia due to the changing shape of the sections, and (3) the follower force effect, which allows the force acting on a section to be affected by the local orientation. Accounting for these effects is crucial to describing the 3D behaviour the beams and plates. In this work the deformations and rotations are primarily present in 2 of the 3 space dimensions, although the full 3D structure is used in analysis and simulation.

To solve for the optimal control of these highly flexible beams, a Non-linear Model Predictive Control (NMPC) method is employed. NMPC has high computational cost when computing the behaviour of high dimensional systems. Indeed, a continuous highly flexible beam has infinite dimensions. A direct discretization of such a beam, using methods like Finite Elements, may have thousands of state variables and would be computationally intractable. A modal description using the beam's primary states (displacements and rotations) may have models with $\mathcal{O}(10^2)$ states to be describe the beam accurately. In this work, a reduced order modal system is used with an intrinsic formulation. In this analysis, the velocities and angular velocities of the beam are tracked, and thus require $\mathcal{O}(10)$ states. This allows the problem to be far more computationally tractable. However the primary variables are now derived from these intrinsic states by integration, introducing some complexity.

In the following sections, the structural model for highly flexible beams is described, including a method to determine the primary states. Next, the finite dimensional approximations to produce the reduced order model is described, and the particular beam structure for this work is presented. The extension to include gravitational forces is described.

4.1 Structural Model

4.1.1 Intrinsic Beam Formulation

In this section, the geometrically exact, fully intrinsic non-linear beam theory model of Hodges [19] is described. The dynamics of the structure are described by two intrinsic states: the linear

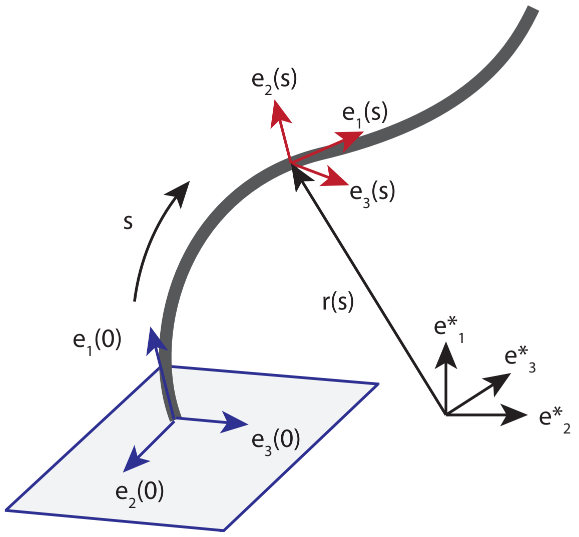


Figure 4.1: Frames of reference on a flexible beam (thick, grey). Global frame (black), base frame (blue) and frame at some arbitrary location along the elastic axis, s (red)

and angular velocities at each of location along the beam, $\boldsymbol{x}_1(s,t) := [\boldsymbol{v}^T, \boldsymbol{\omega}^T]^T : [0,L] \times \mathbb{R}_+ \to \mathbb{R}^6$ and the resultant sectional forces and moments, $\boldsymbol{x}_2(s,t) := [\boldsymbol{f}^T, \boldsymbol{m}^T]^T : [0,L] \times \mathbb{R}_+ \to \mathbb{R}^6$ where s is the curvilinear coordinate that defines the elastic axis and t denotes time.

The undamped equations of motions are written here in a local, body attached frame of reference ($\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$ centred on the origin) such that $\boldsymbol{e}_1$ is tangent to the elastic axis (as in Artola [16]),

$$M\frac{\partial \boldsymbol{x}_1}{\partial t} - \frac{\partial \boldsymbol{x}_2}{\partial s} - E\boldsymbol{x}_2 + \mathcal{L}_1(\boldsymbol{x}_1)M\boldsymbol{x}_1 + \mathcal{L}_2(\boldsymbol{x}_2)C\boldsymbol{x}_2 = \boldsymbol{f}_1 \tag{4.1}$$

$$C\frac{\partial \boldsymbol{x}_2}{\partial t} - \frac{\partial \boldsymbol{x}_1}{\partial s} + E^T\boldsymbol{x}_1 - \mathcal{L}_1^T(\boldsymbol{x}_1)Cx_2 = 0 \tag{4.2}$$

where $M(s) : [0,L] \to \mathbb{S}^6_{++}$ is the symmetric positive definite mass matrix, $C(s) : [0,L] \to \mathbb{S}^6_+$ is the symmetric positive semi-definite compliance matrix (terms defined in the Nomenclature section),

$$M = \mathrm{diag}(\rho A, \rho A, \rho A, \rho I_1, \rho I_2, \rho I_3) \tag{4.3}$$

$$C = \mathrm{diag}(1/EA, 1/GA_2, 1/GA_3, 1/GJ, 1/EI_2, 1/EI_3) \tag{4.4}$$

and $E(s) : [0,L] \to \mathbb{R}^{6\times 6}$ contains the initial curvature and pre-twist (for instance if the structure is an arch),

$$E = \begin{bmatrix} \tilde{\boldsymbol{\kappa}}_0 & 0 \\ \tilde{\boldsymbol{e}}_1 & \tilde{\boldsymbol{\kappa}}_0 \end{bmatrix} \tag{4.5}$$

and $\mathcal{L}_1 : \mathbb{R}^6 \to \mathbb{R}^{6\times 6}$ and $\mathcal{L}_2 : \mathbb{R}^6 \to \mathbb{R}^{6\times 6}$ are defined by

$$\mathcal{L}_1(\boldsymbol{x}_1) = \begin{bmatrix} \tilde{\boldsymbol{\omega}} & 0 \\ \tilde{\boldsymbol{v}} & \tilde{\boldsymbol{\omega}} \end{bmatrix} \tag{4.6}$$

$$\mathcal{L}_2(\boldsymbol{x}_2) = \begin{bmatrix} 0 & \tilde{\boldsymbol{f}} \\ \tilde{\boldsymbol{f}} & \tilde{\boldsymbol{m}} \end{bmatrix} \tag{4.7}$$

where, as before, the $\tilde{(\cdot)}$ is the cross-product operator.

The vector of external forces is $\boldsymbol{f}_1 \in \mathbb{R}^6$ contains both the forces and moments per unit length at each section, and may depend on the state (ie intrinsic variables), the primary variables and/or time.

Boundary conditions are described by either free conditions,

$$\boldsymbol{x}_1(s_\partial, t)^T \boldsymbol{x}_2(s_\partial, t) = 0 \tag{4.8}$$

or by prescribed a state,

$$\boldsymbol{x}_1(s_\partial, t) = \boldsymbol{x}_{1\partial}(t), \quad \boldsymbol{x}_2(s_\partial, t) = \boldsymbol{x}_{2\partial}(t) \tag{4.9}$$

For example, for a cantilevered beam, we must have $\boldsymbol{x}_1(0,t) = 0, \boldsymbol{x}_2(L,t) = 0$.

Therefore, this intrinsic description of the beam is independent of the primary variables, displacements and rotations.

4.1.2 Recovering displacements and rotations

Since displacements and rotations do not appear in the formulation above, they must be derived as dependent variables. There are two methods to determine theses: either by integrating the linear and angular velocities in time from the initial starting state, or by integrating the local rotations and extensions over the length of the beam.

In either case, we need the local orientation relative to an inertial frame of reference, $\boldsymbol{e}_1^*, \boldsymbol{e}_2^*, \boldsymbol{e}_e^*$. A transformation matrix $T(s,t) : [0, L] \times \mathbb{R}_+ \to \mathbb{R}^{3\times 3} := T(\boldsymbol{\xi}(s,t))$ is defined in terms of Equation (3.4) and transforms bewteen the local and inertial frames of reference. The quaternions must satisfy both

$$\frac{\partial \boldsymbol{\xi}}{\partial s} = \mathcal{U}(\boldsymbol{\kappa} + \kappa_0)\boldsymbol{\xi} \tag{4.10}$$

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = \mathcal{U}(\boldsymbol{\omega})\boldsymbol{\xi} \tag{4.11}$$

where $\kappa = [0, I_3]C\boldsymbol{x}_2$ is the vector of curvatures and twist and $\mathcal{U}$ is the skew symmetric operator acting on $\boldsymbol{a} \in \mathbb{R}^3$,

$$\mathcal{U}(\boldsymbol{a}) = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{a}^T \\ \boldsymbol{a} & -\tilde{\boldsymbol{a}} \end{bmatrix} \tag{4.12}$$

Similarly, the displacement field $r(s,t)$ can be solved by either

$$\frac{\partial \boldsymbol{r}}{\partial s} = T(s,t)(\boldsymbol{e}_1 + \gamma) \tag{4.13}$$

$$\frac{\partial \boldsymbol{r}}{\partial t} = T(s,t)\boldsymbol{v} \tag{4.14}$$

where $\gamma = [I_3, 0]C\boldsymbol{x}_2$ is the strain vector.

4.1.3 Damping

Damping is incorporated into the model using the Kelvin-Voigt model [16], by replacing

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{m} \end{bmatrix} \to C^{-1}\begin{bmatrix} \gamma \\ \boldsymbol{\kappa} \end{bmatrix} + C_\tau C^{-1}\frac{\partial}{\partial t}\begin{bmatrix} \gamma \\ \boldsymbol{\kappa} \end{bmatrix} = \boldsymbol{x}_2 + C_\tau \frac{\partial \boldsymbol{x}_2}{\partial t} \tag{4.15}$$

where $\dot{(\cdot)} = \frac{\partial (\cdot)}{\partial t}$ and $(\cdot)' = \frac{\partial (\cdot)}{\partial s}$ and $C_\tau \in \mathbb{S}_+^6$ is a matrix containing the time scales characterising damping in the material. This allows us to write the nonlinear intrinsic beam equations with linear damping as

$$M\dot{\boldsymbol{x}}_1 - \boldsymbol{x}_2' - E\boldsymbol{x}_2 + \mathcal{L}_1(\boldsymbol{x}_1)M\boldsymbol{x}_1 + \mathcal{L}_2(\boldsymbol{x}_2)C\boldsymbol{x}_2 = D_0\boldsymbol{x}_1 + D_1\boldsymbol{x}_1' + D_2\boldsymbol{x}_1'' + \boldsymbol{f}_1 \tag{4.16}$$

$$C\dot{\boldsymbol{x}}_2 - \boldsymbol{x}_1' + E^T\boldsymbol{x}_1 - \mathcal{L}_1^T(\boldsymbol{x}_1)C\boldsymbol{x}_2 = 0 \tag{4.17}$$

with the new boundary condition

$$\boldsymbol{x}_1(s_\partial, t)^T(\boldsymbol{x}_2(s_\partial, t) + C_d(\boldsymbol{x}_1'(s_\partial, t) - E^T\boldsymbol{x}_1(s_\partial, t))) = 0 \tag{4.18}$$

where $C_d = C_\tau C^{-1}$ and

$$D_0 = -(C_d E^T)' - E C_d E^T \tag{4.19}$$
$$D_1 = C_d' - C_d E^T + E C_d \tag{4.20}$$
$$D_2 = C_d \tag{4.21}$$

In this work, Equations (4.16) and (4.17) are used, and the specific parameters used match those from Artola [16].

4.2 Finite Dimensional Model Reduction

4.2.1 Using only the intrinsic states

A finite dimensional approximation of the system dynamics is created using the natural modes of the system [13]. The system is linearised about the unloaded and undeformed condition, $\boldsymbol{f}_1 = \boldsymbol{x}_1 = \boldsymbol{x}_2 = 0$ and without damping,

$$M\frac{\partial \boldsymbol{x}_1}{\partial t} - \frac{\partial \boldsymbol{x}_2}{\partial s} - E\boldsymbol{x}_2 = 0 \tag{4.22}$$
$$C\frac{\partial \boldsymbol{x}_2}{\partial t} - \frac{\partial \boldsymbol{x}_1}{\partial s} + E^T\boldsymbol{x}_1 = 0 \tag{4.23}$$

and is solved for non-trivial solutions of the form

$$\boldsymbol{x}_1 = \boldsymbol{\phi}_{1i}(s)\sin(\omega_i t) \tag{4.24}$$
$$\boldsymbol{x}_2 = \boldsymbol{\phi}_{2i}(s)\cos(\omega_i t) \tag{4.25}$$

where $\phi_{1i}(s), \phi_{2i}(s) : [0, L] \to \mathbb{R}^6$ are the eigenfunctions that describe the mode shapes, and ω_i is the associated natural frequency. Einstein summation notation is used, over indices $i = 1, ...N_m$, where N_m is the number of mode shapes to consider. The eigenfunctions form an orthogonal basis and are normalised such that

$$\int_0^L \boldsymbol{\phi}_{1i}^T M \boldsymbol{\phi}_{1j} ds = \delta_{ij} \tag{4.26}$$
$$\int_0^L \boldsymbol{\phi}_{2i}^T C \boldsymbol{\phi}_{2j} ds = \delta_{ij} \tag{4.27}$$

where δ_{ij} is the Kronecker delta. The mode shapes for the beam studied in this work is depicted in Figure 4.2.

The modal expansion of the state is written as

$$\boldsymbol{x}_1(s,t) = \boldsymbol{\phi}_{1i}(s) q_{1i}(t) \tag{4.28}$$
$$\boldsymbol{x}_2(s,t) = \boldsymbol{\phi}_{2i}(s) q_{2i}(t) \tag{4.29}$$

where $[q_1(t), q_2(t)] : \mathbb{R}_+ \to \mathbb{R}^{2N_m}$ form the modal coordinates.

Equations (4.16) and (4.17) can then be expressed using the modal coordinates using the Galerkin projection. Using the modal expansion (Equations (4.28) and (4.29)) in the dynamics, premultiplying by each eigenfunctions and integrating over the length of the beam, we obtain (full details in [16])

$$\dot{\boldsymbol{q}} = W\boldsymbol{q} + N(\boldsymbol{q})\boldsymbol{q} + \begin{bmatrix}\boldsymbol{\eta}\\ 0\end{bmatrix} \tag{4.30}$$

where $\boldsymbol{q}(t) = [\boldsymbol{q}_1(t)^T, \boldsymbol{q}_2(t)^T]^T$ are the stacked modal coordinate vector of size $2N_m$.[1]

[1]For numerical ease, the rigid body modes are removed from $\boldsymbol{q}_2$, as will be discussed later

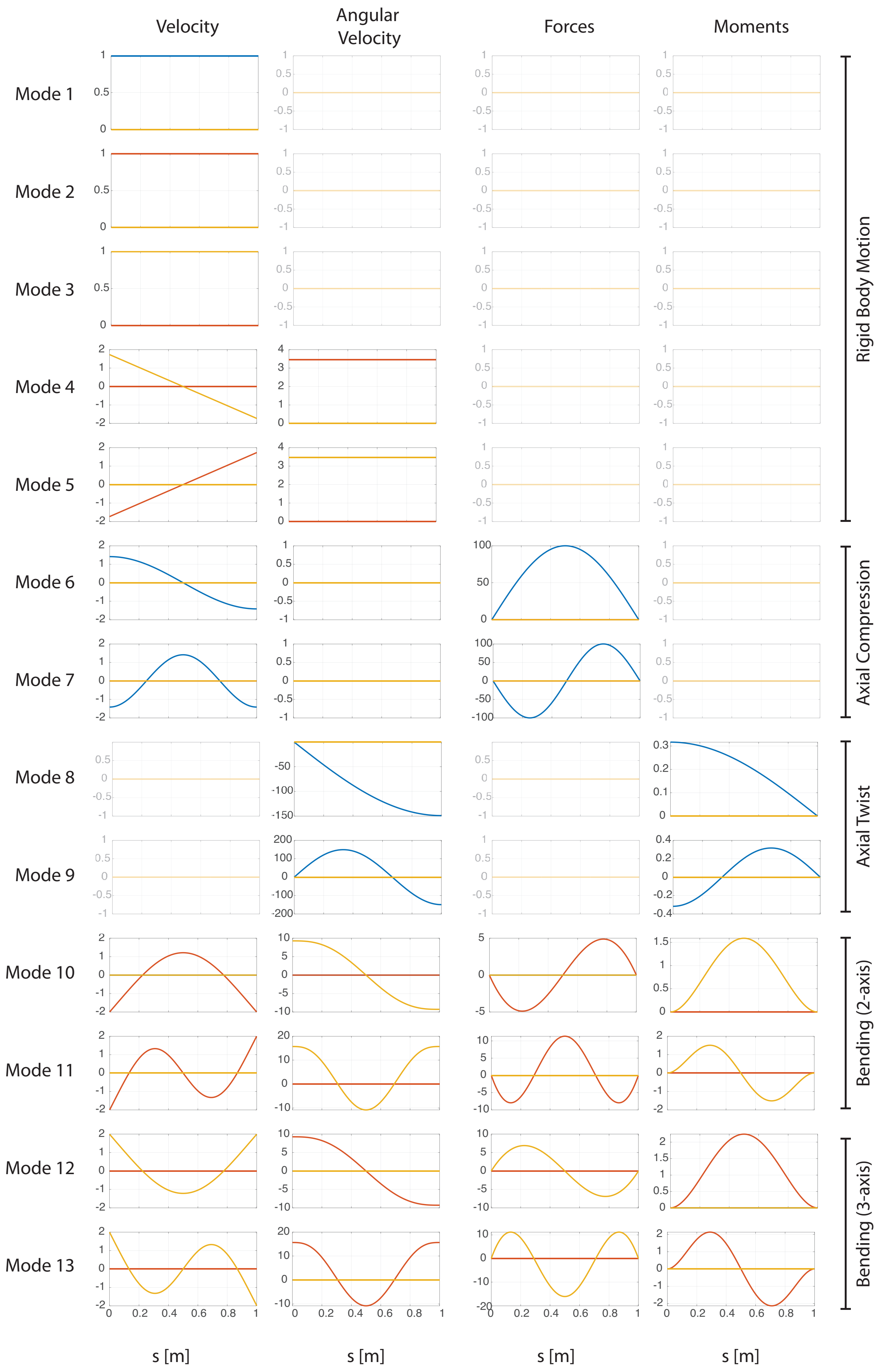


Figure 4.2: Mode Shapes for the beam studied in this work, after normalisation. Blue, red and yellow refer to $[e_1, e_2, e_3]$-axis respectively.

The matrix W is a skew symmetric matrix

$$W = \begin{bmatrix} \Sigma & \Omega \\ -\Omega & 0 \end{bmatrix} \tag{4.31}$$

where Ω is a diagonal matrix of eigenvalues and Σ is the modal damping matrix,

$$[\Sigma]_{i,j} = \int_0^L \boldsymbol{\phi}_{1i}^T (D_0 \boldsymbol{\phi}_{ij} + D_1 \boldsymbol{\phi}_{1j}' + D_2 \boldsymbol{\phi}_{1j}'') ds \tag{4.32}$$

and $N(\boldsymbol{q})$ is non-linear in $\boldsymbol{q}$,

$$N(\boldsymbol{q}) = \begin{bmatrix} -q_{1l}\Gamma_1^l & -q_{2l}\Gamma_2^l \\ q_{2l}(\Gamma_2^l)^T & 0 \end{bmatrix} \tag{4.33}$$

where

$$[\Gamma_1^l]_{i,j} = \int_0^L \boldsymbol{\phi}_{1i}^T \mathcal{L}_1(\boldsymbol{\phi}_{1j}) M \boldsymbol{\phi}_{1l} ds \tag{4.34}$$

$$[\Gamma_2^l]_{i,j} = \int_0^L \boldsymbol{\phi}_{1i}^T \mathcal{L}_2(\boldsymbol{\phi}_{2j}) C \boldsymbol{\phi}_{2l} ds \tag{4.35}$$

$\boldsymbol{\eta}$ contains all the forcing terms (gravity and control inputs in this work),

$$[\eta]_i = \int_0^L \boldsymbol{\phi}_{1i}^T \boldsymbol{f}_1 ds \tag{4.36}$$

Note that $W, \Gamma_1^l, \Gamma_2^l$ are constant matrices and thus are precomputed offline.

The instantaneous energy of the system can be written as

$$\epsilon(t) = \frac{1}{2}\boldsymbol{q}^T \boldsymbol{q} \tag{4.37}$$

and implicitly depends on the mass and compliance matrices through the normalisation (Equations (4.26) and (4.27)). This quantity contains both the energy associated with the rigid body motion and the energy associated with the strain energy from all deformations. Gravitational effects will be considered next.

4.2.2 Including a point control force

In general, the control force must be include as in Equation (4.36). In this work, the controller will exert a linear force at the base of the pendulum, and thus η_u can be expressed as

$$[\boldsymbol{\eta}_u]_i = \int_0^L \boldsymbol{\phi}_{1i}^T \begin{bmatrix} u_{e1} \\ u_{e2} \\ u_{e3} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta(s) ds = \boldsymbol{\phi}_{1i}(0)^T \boldsymbol{u} \tag{4.38}$$

where $\delta(s)$ is the delta function, and $[u_{e1}, u_{e2}, u_{e3}]^T$ is the force acting on the base in the local reference frame of the beam.

4.2.3 Including the motion of the base

To allow the base location to be tracked, the location of the base in the inertial frame can be exposed in the state. We introduce $\boldsymbol{x}_{base}$ into the state vector with dynamics,

$$\frac{\partial \boldsymbol{x}_{base}}{\partial t} = T(\xi(0,t)) \left[I_3, 0\right] \boldsymbol{\phi}_{1i}(0) q_{1i} \tag{4.39}$$

4.2.4 Including gravitational forces

Gravitational forces act on the beam in a direction defined in the inertial frame of reference, in particular, we have

$$\boldsymbol{f}_g(s,t) = \mu(s)\begin{bmatrix} I_3 \\ \tilde{\boldsymbol{r}}_{cm}(s)\end{bmatrix} T(s,t)^T\boldsymbol{g} \tag{4.40}$$

where $\boldsymbol{f}_g(s) : [0,L] \to \mathbb{R}^6$ is the vector of sectional forces and moments caused by gravity, $\mu(s) : [0,L] \to \mathbb{R}_+$ is the sectional mass per unit length of the beam, $\boldsymbol{r}_{cm}(s)$ is the offset between the sectional centre of mass and the elastic axis (0 for symmetric beam cross-sections), T describes the transformation from the inertial to the local frame of reference and $\boldsymbol{g}$ is the gravitational acceleration vector in the inertial frame. Therefore, the finite dimensional approximation of the beam must include the orientation at each section, and encode the gravitational forces.

First, a finite dimensional approximation for the local orientation is developed. The local orientation at all locations on the beam is described by a cubic Catmull-Rom interpolation scheme between a few sampled points $\boldsymbol{\xi}(t)$,

$$\boldsymbol{\xi}(s,t) = S(s)\boldsymbol{\xi}(t) \tag{4.41}$$

where $\boldsymbol{\xi}(t) = [\xi_1^T(t), \dots \xi_{N_\xi}^T(t)]^T : \mathbb{R}_+ \to \mathbb{R}^{4N_\xi}$ stacks the quaternions at N_ξ specified locations along the beam. Only a small number of points are needed to reasonably track the quaternions along the beam, and in this work we chose $N_\xi = 3$. Detailed investigation of the impact of N_ξ is beyond the scope of this work. The matrix $S(s)$ contains the piecewise cubic interpolation ensuring smooth matching at all tracked quaternions, and detailed derivation of this matrix is provided in the appendix.

The Galerkin projection of the gravitational force therefore gives

$$\boldsymbol{\eta}_g(\boldsymbol{\xi}) = \boldsymbol{\eta}_{g0} + \xi_l \Gamma_g^l \boldsymbol{\xi} \tag{4.42}$$

where

$$[\eta_{g0}]_i = \int_0^L \boldsymbol{\phi}_{1i}^T \mu \begin{bmatrix} I_3 \\ \tilde{\boldsymbol{r}}_{cm}\end{bmatrix} \boldsymbol{g} ds \tag{4.43}$$

$$[\Gamma_g^l]_{i,j} = \int_0^L \boldsymbol{\phi}_{1i}^T \mu \begin{bmatrix} I_3 \\ \tilde{\boldsymbol{r}}_{cm}\end{bmatrix} T_{jl}(S_j, S_l)^T \boldsymbol{g} ds \tag{4.44}$$

where S_j is the j-th column of $S(s)$ and

$$T_{jl}(\boldsymbol{\xi}_j, \boldsymbol{\xi}_l) = -2\boldsymbol{\xi}_{j,v}^T \boldsymbol{\xi}_{l,v} I_3 + 2\boldsymbol{\xi}_{j,v}\boldsymbol{\xi}_{l,v}^T + 2\xi_{j,0}\tilde{\boldsymbol{\xi}}_{l,v} \tag{4.45}$$

where both η_{g0} and Γ_g are constant matrices precomputed offline.

This allows the dynamics to be summarised as

$$\dot{\boldsymbol{q}} = W\boldsymbol{q} + N(\boldsymbol{q})\boldsymbol{q} + \begin{bmatrix} \boldsymbol{\eta}_g(\boldsymbol{\xi}) \\ 0 \end{bmatrix} \tag{4.46}$$

$$\dot{\boldsymbol{\xi}}_k = \mathcal{U}(\Phi_\omega \phi_1(s_k)\boldsymbol{q}_1)\boldsymbol{\xi}_k, \quad k = 1, \dots N_\xi \tag{4.47}$$

where $\Phi_w = [0, I_3]$ selects the angular velocity components from the state at each sample point s_k and $\mathcal{U}$ is the operator defined earlier.

This allows the dynamics to be written concisely as

$$\frac{\partial}{\partial t}\begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{\xi} \end{bmatrix} = \begin{bmatrix} W & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{\xi} \end{bmatrix} + \begin{bmatrix} N(\boldsymbol{q}) & \boldsymbol{\xi}_l \Gamma_g^l \\ 0 & N_\xi(\boldsymbol{q}_1) \end{bmatrix}\begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{\xi} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\eta}_{g0} \\ 0 \\ 0 \end{bmatrix} \tag{4.48}$$

The entire system state is now captured in an augmented state vector, $\boldsymbol{q}_a = [\boldsymbol{q}_1^T, \boldsymbol{q}_2^T, \boldsymbol{\xi}^T, \boldsymbol{x}_{base}^T]^T$, and the dynamics in a standard form,

$$\frac{\partial \boldsymbol{q}_a}{\partial t} = \hat{A}(\boldsymbol{q}_a)\boldsymbol{q}_a + \hat{B}\boldsymbol{u} + \hat{C} \tag{4.49}$$

Chapter 5

Controller Design

5.1 Model Predictive Control

A conventional discrete time MPC formulation is used to solve Equation (4.49) for the control history $\boldsymbol{u}(t)$. We assume the state if fully and exactly observable, and there are no external disturbances. The solution is derived over a prediction horizon τ_p that is split into N intervals.

$$
\begin{aligned}
\underset{\boldsymbol{q}_a^k,\,\boldsymbol{u}^k}{\text{minimize}} \quad & \frac{1}{2}\delta\boldsymbol{q}_a^{N^T} P \delta\boldsymbol{q}_a^N + \sum_{k=0}^{k=N-1} \frac{1}{2}\delta\boldsymbol{q}_a^{k^T} Q \delta\boldsymbol{q}_a^k + \frac{1}{2}\delta\boldsymbol{u}^{k^T} R \delta\boldsymbol{u}^k && (5.1a)\\
\text{subject to} \quad & \delta\boldsymbol{q}_a^0 = \delta\boldsymbol{q}_{a0} \qquad \text{(initial condition)}, && (5.1b)\\
& \boldsymbol{q}_a^{k+1} = \boldsymbol{f}(\boldsymbol{q}_a^k, \boldsymbol{u}^k) \quad \forall k = 1, ...N-2, \quad \text{(dynamics)}, && (5.1c)\\
& \delta\boldsymbol{u}_k \in [\boldsymbol{u}_l, \boldsymbol{u}_u] \quad \forall k = 1, ...N-1, \quad \text{(control limits)} && (5.1d)
\end{aligned}
$$

where $Q > 0$ and $R > 0$ represent the cost function weights on the state and control inputs, $P > 0$ is the cost on the terminal state within the prediction horizon and $\delta\boldsymbol{q}_a = \boldsymbol{q}_a - \boldsymbol{q}_{a,ref}$ and $\delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_{ref}$ represent the deviations of the state and control input from a reference state and control input. The reference states are chosen to be a fixed point of the system, ie $\dot{\boldsymbol{q}}_a = \boldsymbol{f}(\boldsymbol{q}_{a,ref}, \boldsymbol{u}_{a,ref}) = 0$.

While the discretisation time step for determining the control inputs is of length $t_s = \frac{\tau_p}{N+1}$, the dynamics (Equation (5.1c)) are solved for using a more accurate Runge Kutta-4 integration of the system dynamics Equation (4.49). This optimisation problem is solved using the parallelised multiple shooting method which has desirable computational speed [16]. The code is implemented using MATLAB, and a listing is available on Github.[1]

Algorithm 1: Overview of method used to solve for optimal control

```
Result: Optimal control history $\boldsymbol{u}(t)$
Define problem parameters;
Load Mode shapes (precomputed offline);
Normalise mode shapes;
Compute coefficient matrices of Equation (4.49);
Solve for $\boldsymbol{q}_{a,ref}, \boldsymbol{u}_{ref}$;
Specify initial state $\boldsymbol{q} \leftarrow \boldsymbol{q}_{a0}$;
Specify initial guess for states over prediction horizon;
Solve for initial displacement and rotations;
Define $Q, R, P$;
for ($t = 0;\ t < t_f;\ t = t + t_s;$) do
    (For each time step in simulation duration);
    Reset BFGS Hessian estimate;
    while $\epsilon > tolerance$ do
        for ($i = 0;\ i < N;\ i++;$) do
            (For each shot);
            Determine final state of shot;
            Determine sensitivity of final state of each shot to start state and control inputs;
            Determine sensitivity of total cost to start state and control inputs;
        end
        Define equality and inequality constraints;
        Update Hessian estimate (using BFGS);
        Solve quadratic program for new estimate of start states and control inputs;
        $\epsilon \leftarrow$ Estimate of constraint violation;
        Update start states and control inputs;
    end
    Select $\boldsymbol{u}^0$ and integrate forward one time-step (using RK4);
end
```

5.2 Design of Cost Functions

The cost matrices Q, R, P fundamentally control the behaviour of the beam and are the focus of this work. First the design of cost matrices with desirable convergence properties is presented. Next the design of a controller that exploits the natural dynamics of quaternions is presented. In the following chapter they are implemented and compared.

5.2.1 Stabilising cost function

By including the quaternion in the state we have introduced linearly unstabilisable modes about the target reference state with $\boldsymbol{q}_1 = 0$. Intuitively this means that if the beam is at rest in an equilibrium condition, a small perturbation of the orientation will either induce oscillatory behaviour (for instance if the pendulum was in its downright position) or exponentially diverge from the equilibrium condition (for instance from the upright position). Therefore, the matrices Q, R, P must be chosen such that the non-linear controller can stabilise these unstable modes.

Artola [16] defines the required Linear Matrix Inequality (LMI) conditions to prove the controller can bring the flexible pendulum to an upright state from a region around this unstable equilibrium position. In particular he decomposes the state into stabilisable and unstabilisable modes

[1] `https://github.com/dev10110/Final_Year_Project`

$\boldsymbol{\delta} q = [\boldsymbol{w}^T, \boldsymbol{d}^T]^T$ and defines the MPC cost function

$$V(\delta \boldsymbol{q}_a, \boldsymbol{u}) = V_f(\boldsymbol{q}_a^N) + \sum_{k=0}^{N-1} l(\delta \boldsymbol{q}_a^k, \boldsymbol{\delta u}^k) \tag{5.2}$$

$$= \boldsymbol{w}^T P_w \boldsymbol{w} + \sum_{k=0}^{N-1} \left(\frac{\beta}{4} \boldsymbol{w}^T P_w \boldsymbol{w} + \boldsymbol{u}^T R \boldsymbol{u} \right) \tag{5.3}$$

thus only penalising the stabilisable states, and can be written in the standard cost function formulation of Equation (5.1). For this cost function (and appropriately chosen β, P_w, P_d, R, it was shown that there exists a controller that can drive the final state to where both $\boldsymbol{w}^T P_w \boldsymbol{w}$ and $\boldsymbol{d}^T P_d \boldsymbol{d}$ are bounded. Specifically we can chose the cost functions such that the former is (potentially) large but the later is small. This means that the controller can bring the uncontrollable modes to a small bound, and then asymptotically make the stabilisable modes arbitrarily small.

To avoid confusion, this cost function shall henceforth be referred to as the stabilising cost function.

The analysis however only proves this controller is stable for some region of initial states about the reference state. In the next section, I propose a different cost function that makes no guarantees of convergence, but produces desirable behaviour.

5.2.2 Orientation cost function

As demonstrated in the previous chapter, if a controller can impose a relationship between the angular velocity and the vector component of the quaternion as

$$\boldsymbol{\xi}_v = -\frac{\boldsymbol{\omega}}{c} \tag{5.4}$$

the quaternion will evolve towards the $\boldsymbol{\xi} = [1, 0, 0, 0]^T$ state. Since the angular velocity cannot be directly controlled, a suitable controller might be expected to minimise the error,

$$\underset{\boldsymbol{u}(t)}{\text{minimize}} \quad \int_0^{t_f} \left|\left| \boldsymbol{\xi}_v + \frac{\boldsymbol{\omega}}{c} \right|\right|_2^2 dt \tag{5.5}$$

To define an similar controller over a flexible beam, I propose the averaged error over the length of the beam,

$$\underset{\boldsymbol{u}(t)}{\text{minimize}} \quad \int_0^{t_f} \int_0^L \left|\left| \boldsymbol{\xi}_v(s,t) + \frac{\boldsymbol{\omega}(s,t)}{c} \right|\right|_2^2 ds dt \tag{5.6}$$

The spatial integrand can be expressed in terms of the modal coordinates,

$$\int_0^L \left[\underbrace{\boldsymbol{\xi}_v^T \boldsymbol{\xi}_v}_{(i)} + \underbrace{\frac{1}{c} \boldsymbol{\xi}_v^T \boldsymbol{\omega}}_{(ii)} + \underbrace{\frac{1}{c} \boldsymbol{\omega}^T \boldsymbol{\xi}_v}_{(iii)} + \underbrace{\frac{1}{c^2} \boldsymbol{\omega}^T \boldsymbol{\omega}}_{(iv)} \right] ds \tag{5.7}$$

where each of the terms are

$$(i): \quad \int_0^L \boldsymbol{\xi}_v(s,t)^T \boldsymbol{\xi}_v(s,t) ds = \boldsymbol{\xi}^T(t) \left[\int_0^L S(s)^T \Phi_{\xi_v}^T \Phi_{\xi_v} S(s) ds \right] \boldsymbol{\xi}(t) \tag{5.8}$$

$$= \boldsymbol{\xi}^T Q_{\xi\xi} \boldsymbol{\xi} \tag{5.9}$$

$$(ii): \quad \int_0^L \frac{1}{c} \boldsymbol{\xi}_v^T(s,t) \boldsymbol{\omega}(s,t) ds = \frac{1}{c} \boldsymbol{\xi}^T(t) \left[\int_0^L S(s)^T \Phi_{\xi_v}^T \Phi_\omega \boldsymbol{\phi}_{1i} ds \right] q_{1i} \tag{5.10}$$

$$= \frac{1}{c} \boldsymbol{\xi}^T [Q_{\xi\omega}]_i q_{1i} \tag{5.11}$$

$$(iii): \quad \int_0^L \frac{1}{c} \boldsymbol{\omega}(s,t)^T \boldsymbol{\xi}_v(s,t) ds = \left(\frac{1}{c} \boldsymbol{\xi}^T [Q_{\xi\omega}]_i q_{1i} \right)^T \tag{5.12}$$

$$(iv): \quad \int_0^L \frac{1}{c^2} \boldsymbol{\omega}(s,t)^T \boldsymbol{\omega}(s,t) ds = \frac{1}{c^2} q_{1i} \left[\int_0^L \boldsymbol{\phi}_{1i}^T \Phi_\omega^T \Phi_\omega \boldsymbol{\phi}_{1j} ds \right] q_{1j} \tag{5.13}$$

$$= \frac{1}{c^2} q_{1i} [Q_{\omega\omega}]_{ij} q_{1j} \tag{5.14}$$

where $\Phi_{\xi_v} = [0, I_3] \in \mathbb{R}^{3\times4}$ selects the vector component of the quaternions, and $\Phi_\omega = [0, I_3] \in \mathbb{R}^{3\times6}$ selects the angular velocity components of the $\boldsymbol{q}_1$ states. Note $[Q_{\xi\omega}]_i$ refers to the i-th column of $Q_{\xi\omega}$. For simplicity I have assumed c is a scalar constant, although the extension for c being state, space and/or time dependent is trivial.

Therefore, the cost weights Q can be written as

$$\boldsymbol{q}_a^T Q \boldsymbol{q}_a = \begin{bmatrix} \boldsymbol{q}_1^T & \boldsymbol{q}_2^T & \boldsymbol{\xi}^T & \boldsymbol{x}_{base}^T \end{bmatrix} \begin{bmatrix} \frac{1}{c^2} Q_{\omega\omega} & 0 & \frac{1}{c} Q_{\xi\omega}^T & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{c} Q_{\xi\omega} & 0 & Q_{\xi\xi} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1 \\ \boldsymbol{q}_2 \\ \boldsymbol{\xi} \\ \boldsymbol{x}_{base} \end{bmatrix} \tag{5.15}$$

Based on the structure of Equation (5.15) we can make a few observations:

1. There is no weighting on the q_2 states - this controller will make no effort to bring the sectional forces and moments to their target values. As we will see later, this means that the strain energy is only dissipated through damping, not due to the controller.
2. There is no weighting on the x_{base} states - we expect that this controller will not have any tendency to being the base back to the origin.
3. The value of c will have an important effect in the response of the controller. For large c, the cost on the angular velocity states is removed. Effectively this means that we loose information on the beneficial coupling between ξ_v and ω. The controller will try to minimise the vector component of all the quaternions without information on how they are related. Furthermore, if c is not of an appropriate order of magnitude, the controller (as we saw in the rigid body case) would spend effort in slowing down or speeding up the angular velocity. It is reasonable to assume that the appropriate value of c therefore depends on model parameters, initial conditions and material properties of the beam.

Before proceeding to numerical simulations of this new cost function, inspecting the numerical values of Q provides further insight. Two modifications are made to the cost function to increase numerical stability, and a hybrid cost function including both the stabilising controller and the orientation based controller is introduced.

Modification 1: Removing Modes 8 & 9

To understand an appropriate order of magnitude for the controller cost c, Figure 5.1 shows graphically the weights of the controller for different values of c. Initially, it seems that the choice of c has little impact on the weights, as two particular weights $Q_{8,8}$ and $Q_{9,9}$ dominate the weighting matrix completely. This makes sense as the angular velocity of the normalised 8th and 9th mode shapes have extrema of order 150 rad/s (Figure 4.2), while the quaternions states have order 1.

However the angular velocity of modes 8 and 9 represent the rotations about the $\boldsymbol{e}_1$ axis, which are not directly controllable. As such, removing them from the simulation will aid in the numerically stability of the controller. This produces the 'modified' cost function Q_{mod}, which sets $Q_{8,8}, Q_{9,9}$ to 0, and is visualised in the second row of Figure 5.1. The performance of the controller for different values of c will be discussed with the results.

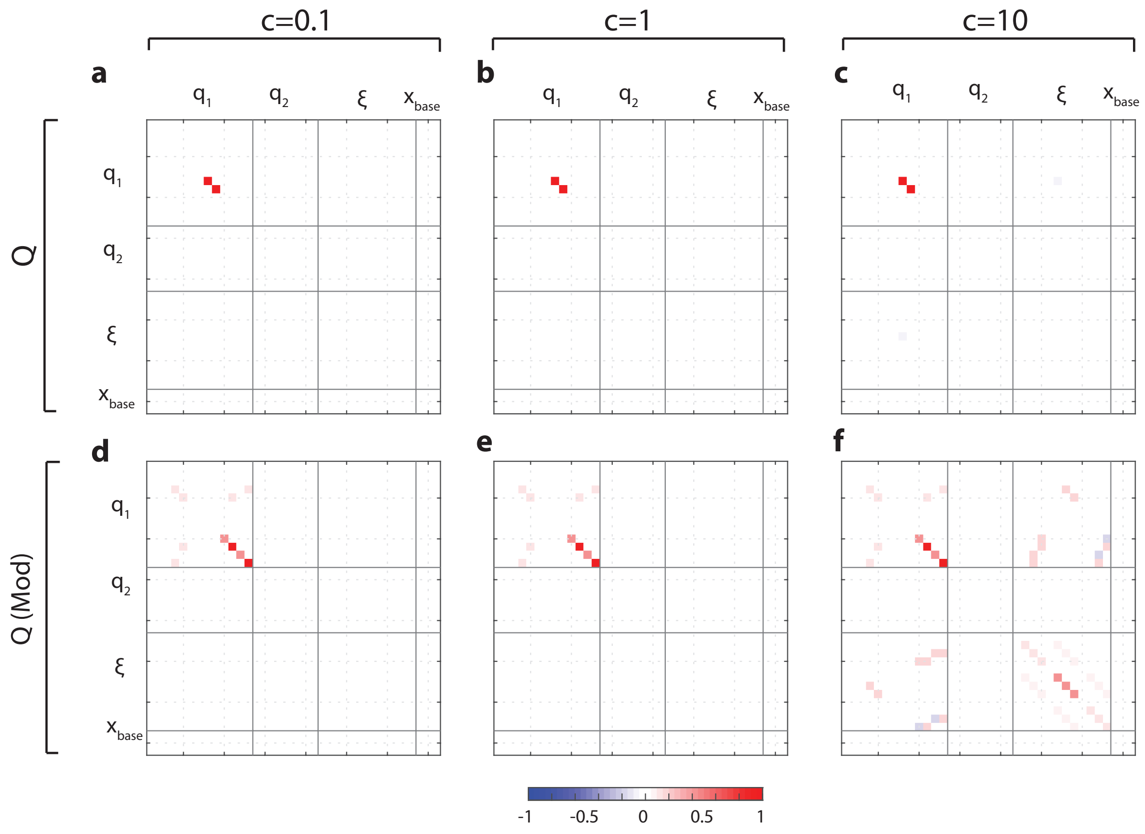


Figure 5.1: Visualisation of the cost matrices. (a,b,c) show the orientation based cost matrix, and (d,e,f) show the modified cost matrix. Each matrix is normalised by its 2-norm. The dark lines across the images show the block-matrix divisions of Equation (5.15).

Modification 2: Removing Mode 23

When deriving the stabilising cost function, we established that it only places a cost on the stabilisable modes of the beam. If the orientation based controller makes no such assumption, does it try to stabilise the unstabilisable modes?

Figure 5.2 shows the sparsity pattern of both matrices. We can see that the stabilising controller has many more non-zero weights than the orientation based controller, in particular it also responds to $\boldsymbol{q}_2$ and $\boldsymbol{x}_{base}$. Furthermore, most of the weights for the orientation based cost function lie on the stabilisable modes identified by the stabilising cost function.

Only mode 23 has weights in the orientation based controller where the stabilising controller does not. This mode refers to the $\boldsymbol{e}_1$ component of the quaternion describing the orientation of the base - the axial twist of the beam at the base. However as this is fixed in the problem (as a boundary condition), it should not have an impact on the convergence of the controller. Indeed, looking over the simulations (next chapter) we can see that mode 23 has small activations of order 10^{-4}. For numerical reasons, the weights dependent on mode 23 can be removed before running the optimisation problem.

In the next section, I discuss hybrid cost functions that include both the stabilising Q and the orientation based Q and how they compare.

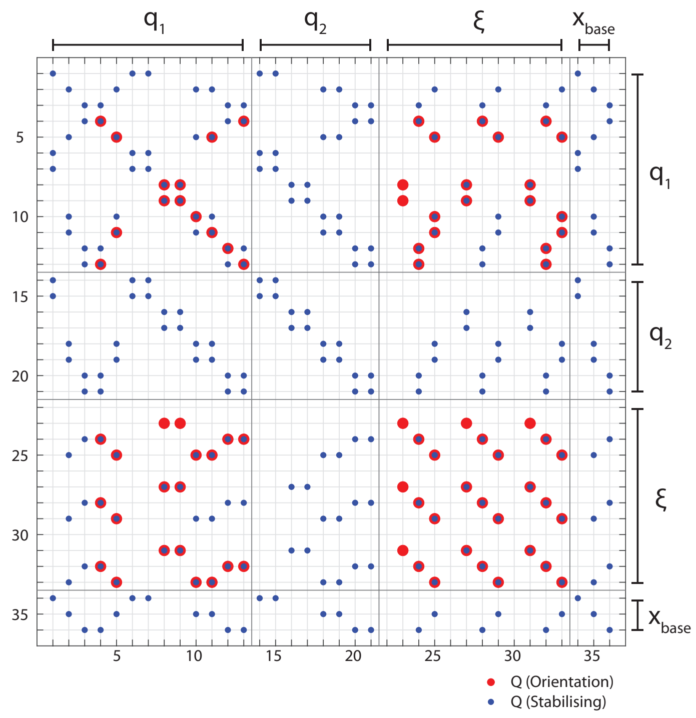


Figure 5.2: Sparsity of cost matrices. Each blue dot indicates the entries of the stabilising cost matrix that are non-zero, and each red dot corresponds to those of the orientation based controller. Only weights associated with mode 23 are non-zero in the orientation based controller but not in the stabilising cost matrix. 'Non-zero' is defined as $|(Q_{ij})| > 10^{-12}$.

5.2.3 Hybrid Cost functions

As noted earlier, the orientation based controller makes no effort to bring the base to the origin or to stabilise the q_2 states or the rigid body motion. As will be demonstrated in the next chapter, we observe such behaviour numerically. Thus, a hybrid controller is desirable - away from the upright condition, the MPC should use the orientation based controller to drive the flexible pendulum towards an upright condition. Once close to the upright condition, the stabilising controller can be used effectively to bring the pendulum to the desired final state.

This can be implemented directly in code using conditional statements. We could define a small region around the upright condition and switch cost functions midway during the simulation from the orientation based controller to stabilising controller. There are a few limitations though: (1) this terminal region would need to be defined rigorously, (2) a sudden switch in the controller cost function will not produce a smooth transition, and it is likely that a sudden change in the applied force will destabilise the orientation of the pendulum (3) if only the orientation based controller is used outside this terminal region, the pendulum could drift very far from the the origin - far enough to not be able to reach the terminal region. Artola [16] rigorously defines a condition that can be checked to see if the stabilising controller will converge and thus limitation (1) can be solved. However limitation (3) was frequently observed in simulations that attempted this scheme.

Instead, a hybrid cost function that is not changed during the simulation is desirable. A simple normalised sum of the two cost functions can be defined

$$Q_{hybrid} = f_s \frac{Q_{stabilising}}{||Q_{stabilising}||_2} + f_o \frac{Q_{orientation,mod}}{||Q_{orientation,mod}||_2} \tag{5.16}$$

where f_s and f_o represent scalar weights that will be tuned in the next chapter, and the modified orientation based controller is used.

Chapter 6

Numerical Results

In this chapter the results and insights gained through a numerical study are discussed. Since of primary interest is the design of the cost function, comparing different simulations on the convergence or final value of the cost function is irrelevant. Instead, we have physically desirable behaviours: the controller must bring the pendulum from a downright to an upright position. It is desirable for this to occur with minimal control effort, requiring as little volume (equivalent of the rail length for classical cart-pole problems) and as quickly as possible, both in physical time and the computation time. Furthermore, as the beam is flexible, we also need the beam to be 'straight.'

In the first section I define these figures of merit precisely. Next, the simulations performed are presented, and the following key results are identified and supported:

1. The orientation based controller is able to bring the pendulum to an upright configuration
2. The controller strength c controls the time taken to make the pendulum upright
3. Fusing the stabilising controller with the orientation based controller is beneficial

6.1 Figures of Merit, V

- *Control Effort*

  The control input is the (linear) force acting at the base, f_u. I define the controller effort in the traditional sense,

$$V_{\mathrm{u}} = \frac{1}{t_f}\int_0^{t_f} ||\boldsymbol{f}_u - \boldsymbol{f}_{uref}||dt \tag{6.1}$$

  Since the maximum controller input was bounded for all simulations, $f_{u,i} < 10$ N, $i = 1, 2, 3$, it is not included as a figure of merit.

- *Simulation Bounds*

  A significant challenge of controlling the traditional cart-pole is the bounded rail length. In this 3D extension to the cart-pole, the volume in which the base moves maybe considered an equivalent figure of merit. However as the simulations lie in the $\boldsymbol{e}_1, \boldsymbol{e}_3$ plane, we will define two figures of merit based on the minimum area required: (1) the minimum rectangular area needed to contain the base of the simulation and (2) the maximum distance the base gets from the origin, over the simulation duration.

$$V_{\mathrm{rect}} = (\max(x_{1,base}(t)) - \min(x_{1,base}(t))) \times (\max(x_{3,base}(t)) - \min(x_{3,base}(t))) \tag{6.2}$$

$$V_{\mathrm{maxDist}} = \max ||\boldsymbol{x}_{base}(t)|| \tag{6.3}$$

- *'Uprightness' of the beam*

The 'uprightness' of the beam is measured as the space-averaged Euler angle. Its normalised by π such that uprightness is defined between $[0,1]$

$$V_{uprightness}(t) = \frac{1}{L}\int_0^L 1 - \frac{\theta(s,t)}{\pi} ds \tag{6.4}$$

$$\theta(s,t) = 2\arccos(\Phi_{\xi 0} S(s)\boldsymbol{\xi}(t)) \tag{6.5}$$

where $\Phi_{\xi 0} = [1,0,0,0]$ selects the first component of the quaternion. In the tables the final uprightness is reported.

- *Time to upright*

  $V_{\text{t, up}}$ is defined as the time it takes for the uprightness (as above) to be greater than 0.90 for all future time.

- *Simulation time*

  A relative measure of time for computation is assessed, normalised by the first simulation. Since the first time step in the MPC usually takes much longer than the rest, the first ($V_{\text{t, sim1}}$) and the average of the remaining iterations ($V_{\text{t, simAvg}}$) is provided.

The full set of simulations and figures of merits is listed in the appendix. The relevant simulations will be highlighted in the next sections.

6.2 Orientation based controller is able to orient the flexible pendulum

Figure 6.1 demonstrates the performance of a controller using only the orientation based cost function. We can see the controller is able to make the pendulum upright in a short ($\sim$ 0.5 seconds) period of time (Figs a, c). The controller pushes the beam (in the global frame) to the right and then brings the base down thus inverting the pendulum. Intuitively this resembles the behaviour a human might execute, if they did not have any torque control at the base, as is in our simulations.

Fig b shows the control history required to execute this manoeuvre. [1] We can see large, rapidly changing changes in the control history at each timestep. These perturbations continue after the magnitude of the control forces has reduced for $t > 1.5$ seconds. The initial vibrations are necessary to explain the behaviour near the origin. The base starts moving to the left, which causes the tip to curl backwards (snapshots 1-3 of Fig a). However soon after, the tip accelerates further to the left than the base, and allows the bar to swing up anti-clockwise. This can only be explained by the base slowing down (in the e_3 direction) and moving downwards, increasing the curvature in the beam (snapshots 4-6). This is the sign change in u_3 seen at $t = 0.3$ (Fig b).

Inspecting the states at $t > 1.5$ seconds, it seems that they represent little pressure waves along the beam's elastic axis. Tightening the simulation tolerances reduced this slightly, but the high frequency oscillations were not suppressed. The reason for these waves warrants further investigation. My initial hypothesis is that the controller may be exploiting a known stabilising method for rigid linked pendulums: by vibrating the pivot point at a sufficiently large frequency and amplitude, a multi-pendulum with finite number of sections can be stabilised about the inverted position[2] [20]. This theory's extension to flexible beams has not yet been established.

Fig c shows the scalar component of the quaternions at three tracked locations along the beam. While the shape and timescales are similar to the expected results from Section 3.4, later results will show that the the timescales do not match that of the analytic results. This makes sense, as the controller does not reach a situation where $\boldsymbol{\xi}_v \approx -\boldsymbol{\omega}/c$ for $\sim$ 1 seconds (Fig d) by which time the pendulum is already upright. The norm of the errors (Fig d) shows that even though the error at the tip and mid sections is almost twice that of the error at the base, they converge to their final values in the same amount of time.

[1]As these represent the forces in the local frame of reference of the base, they do not visually correspond to the movement of the base in Fig a.

[2]Video summary by Mould, 2014: `https://www.youtube.com/watch?v=gnn21smGVrQ`

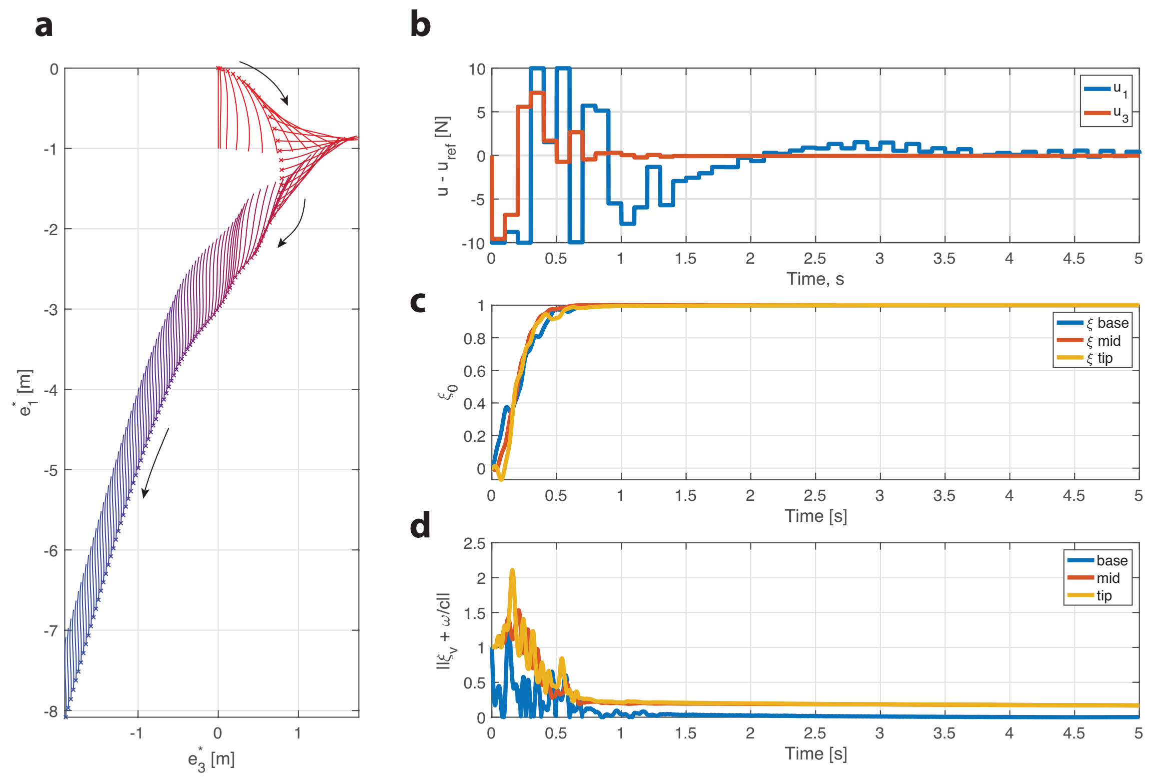


Figure 6.1: Performance of orientation based controller. (a) Visualisation of base path (crosses) and beam shape (lines) over first 2 seconds. Pendulum starts in the down position (red) at the origin, and tends to a constant velocity (blue). Each snapshot represents 20 ms. (b) Control history for this simulation. $u_{ref} = [g_0, 0, 0]^T$ represents the input force to counter gravity when in an upright configuration. (c) Evolution of the scalar component of the tracked quaternions. The beam is upright in about 0.5 seconds. (d) Error in the objective function at each of the three tracked locations. Parameters: $Q = Q_{\text{orientation}}, c = 10$. (Simulation 9)

Finally, we can see that the pendulum converges into a state where it has a non-zero velocity to the lower left. This is reasonable as the controller has no penalty on the linear velocity states. Furthermore, we can observe the control input (Fig b) and the norm of the quaternion error (Fig d) are non-zero 5 seconds into the simulation. These have two causes: (1) since the velocity and q_2 states are not penalised by this controller, the total energy in the beam is constantly oscillating between linear kinetic energy, strain energy and rotational kinetic energy, but the controller only seeks to suppress the rotational kinetic energy. (2) numerical artefacts from the integration schemes used.

These limitations are addressed by the hybrid controllers, presented next.

6.3 Hybrid Controller Performance

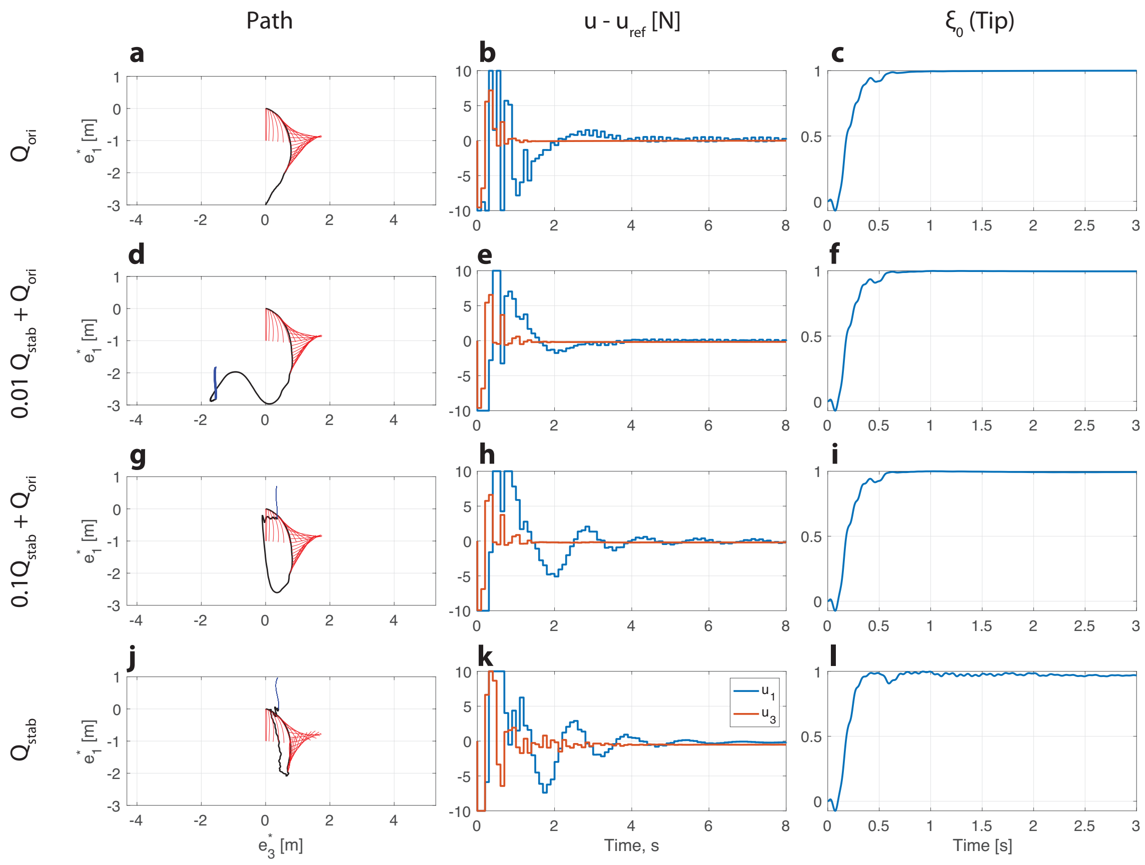


Figure 6.2: Performance of hybrid controllers. Each row represents results from increasing fraction of $Q_{stabilising}$ compared to $Q_{orientation}$. (a, d, g, j) The first column shows the path of the base and the beam shapes. Beam shapes are shown as snapshots, from the start (red) to the end (blue) in increments of 20 ms. For clarity, only the first and last 0.5 seconds are shown. (b,e,h,k) The second column shows the control history. (c,f,i,l) The third column shows the tip quaternion history.

Table 6.1: Comparison of hybrid controller performance. f_s and f_o denote the relative weights of the stabilising and orientating controllers (Equation (5.16)). Full table of parameters is in the appendix.

Sim Index	f_s	f_o	V_u	$V_{\text{uprightness}}$	$V_{\text{t, upright}}$	$V_{\text{rectangle}}$	V_{maxDist}	$V_{\text{t, sim1}}$	$V_{\text{t, sim avg}}$
9	0	1	1.47	1.00	0.60	-	-	433.1	3.3
12	0.01	1	1.32	0.97	0.59	7.56	3.34	310.0	6.7
11	0.1	1	1.75	0.96	0.58	2.43	2.63	168.8	10.1
10	1	0	2.09	0.92	1.99	1.66	2.18	503.6	5.0

Hybrid controllers allow the algorithm to simultaneously achieve two desirable behaviours - quick and efficient uprighting, and the ability to bring the beam towards the origin with damped velocity and strain modes. Figure 6.2 shows the behaviour of the controllers as the relative importance of $Q_{\text{orientation}}$ and $Q_{\text{stabilising}}$ are changed, going down the figure. Across each row, we see snapshots showing the initial behaviour of the controller and the final resting condition after 10 seconds (figs a, d, g, j). By introducing even a small amount of $Q_{stabilising}$ we can see the final state of the beam is upright and only 2.5 meters away from the origin (fig d). This position is not at the origin, as there is a local minima for the optimiser, where it prefers to remain a distance away from the origin and not induce the velocity costs of trying to move to the origin. These offsets could easily be removed by introducing an integral error term as in PID controllers.

With greater weight on $Q_{stabilising}$ the behaviour of the controller shifts - an oscillating controller is seen. Such a controller is able to navigate the beam towards the origin while only roughly maintaining the uprightness of the beam. As it approaches the origin, the controller attempts to straighten the beam, but is struggles to do so easily (vibrations in ξ_0 are visible after the initial 0.5 seconds until about 3 seconds, fig l). [3]

Table 6.1 lists the relevant figures of merit. We can see that the final uprightness decreases as more $Q_{stabilising}$ is introduced. Furthermore, the overall time to upright is significantly greater for the controller with just the stabilising cost.

6.4 Choice of c

Table 6.2: Impact of controller strength on response. each row shows a simulation with a changing controller strength c. Uses $f_o = 1, f_s = 0.1$. Full table of parameters is in the appendix.

Sim Index	c	V_u	$V_{\text{uprightness}}$	$V_{\text{t, upright}}$	$V_{\text{rectangle}}$	V_{maxDist}	$V_{\text{t, sim1}}$	$V_{\text{t, sim avg}}$
15	0.1	2.95	0.96	1.13	36.05	8.49	812.96	12.08
13	1	2.91	0.96	1.12	35.05	8.36	1050.37	13.32
18	$\sqrt{6g_0}$	1.56	0.97	0.64	9.78	3.36	224.46	8.24
12	10	1.32	0.97	0.59	7.56	3.34	309.96	6.75
14	30	1.98	0.97	0.39	5.25	3.65	1043.37	8.66

The controllers are all able to make swing the pendulum to an upright position, to a similar uprightness measure $V_{uprightness}$. As we increase the controller strength, we can see the time required to reach the upright state is decreasing, but is not decreasing as $1/c$ as predicted by the analytic calculations Equation (3.26). This is expression would be only be true if the state satisfied $\boldsymbol{\xi}_v = -\omega/c$ in a timespan much shorter than the timescales of the dynamics, as mentioned earlier. Furthermore, for small c cases, the controller is capped out at the maximum control inputs. For very large c, we can see very high frequency changes in the force input, and demonstrates that if c is too large, numerical issues become apparent and simulations with much finer timesteps are likely required to resolve the behaviour.

For the range of simulations performed, controller strengths of $c = \sqrt{6g_0}$ and $c = 10$ provided similarly high-performing results on controller effort and simulation time. This parallels the rigid pendulum case where $c = \sqrt{6g_0}$ was shown to use the open-loop dynamics of the pendulum to its advantage. Since the flexible pendulum has the same mass properties as the rigid pendulum, a similar controller strength can be expected to show good results. The computation times for these simulations was also the shortest (by at least a factor of 2) which suggests that the optimiser did not have to re-optimise the trajectory multiple times during each iteration of the MPC.

It is also interesting that the control effort V_u is lowest for these two simulations. Further investigation would be needed to understand if this minima holds if more simulations are performed, and if other the parameters (R, u_{max}, t_p, etc) are also varied.

[3]The summary images in the appendix may be useful to verify this.

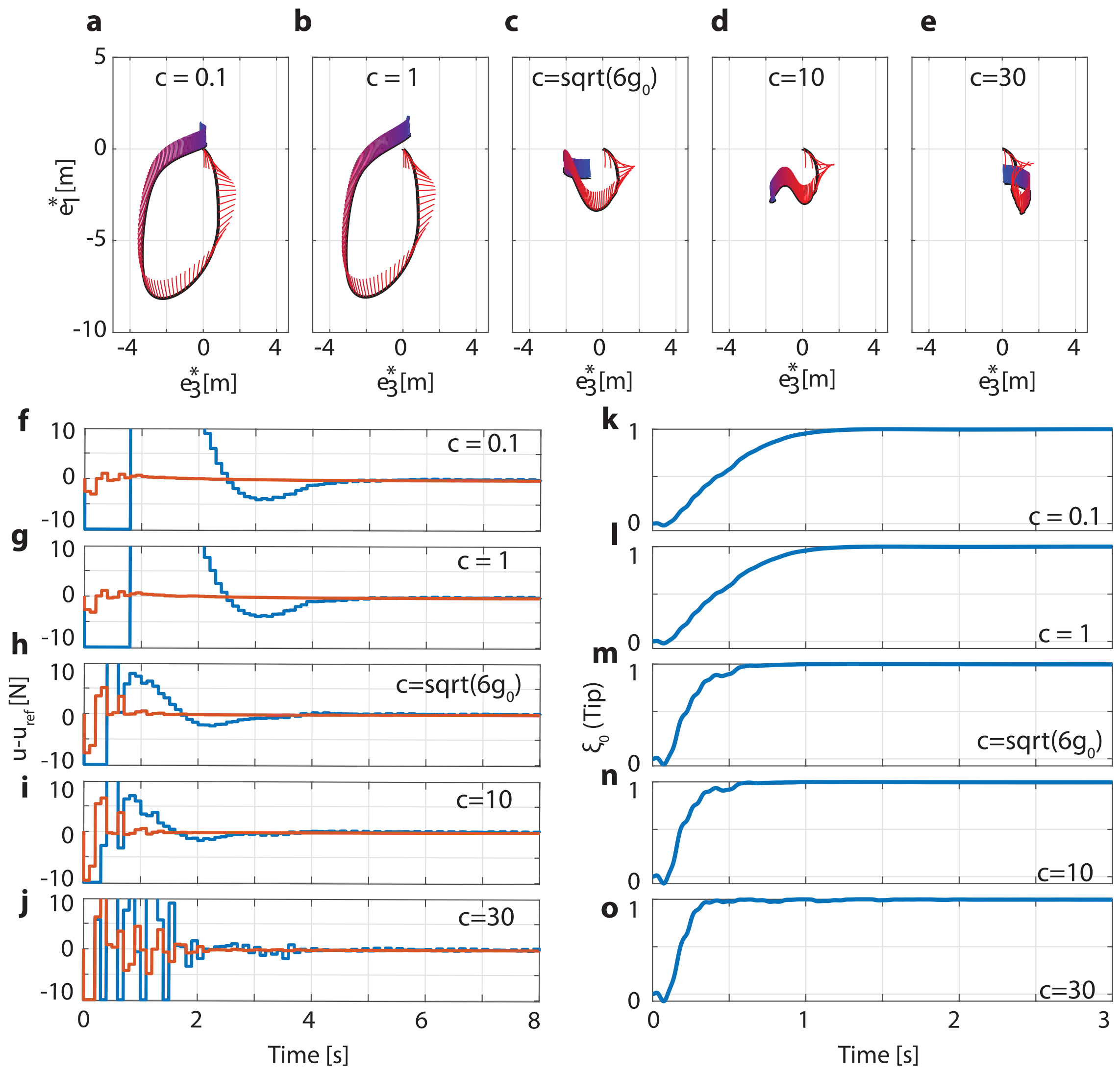


Figure 6.3: Impact of changing the controller strength c. (a-e) show the beam path and snapshots of the beam. The colour indicates time, starting at red and ending at blue. Each snapshot shows 20 ms. (f-j) show the required control history and (k-o) show the tip quaternion's scalar component against time. $f_o = 1, f_s = 0.1$ for these simulations.

Chapter 7

Conclusion

7.1 Contributions

In conclusion, this work uses an insight into rotational dynamics to define a controller that implicitly can rotate a body to a known state. The controller's behaviour is firs investigated in the rigid body case as a motivator, demonstrating that this single control scheme can behave like an energy shaping method far from the inverted state, but also reduces to the stabilising LQR solution near the inverted state.

Applying this to a flexible beam, this work presents for the first time, the full swing up control of a very flexible pendulum using a geometrically exact description of the beam.

7.2 Future Work

In this work, the impact of controller strength and the hybrid cost functions was the focus. As noted in the above sections, there still remain many open questions on the performance and behaviour of these controllers. Of particular importance I believe, are

1. impact of introducing a large controller input penalty (R),
2. behaviour of a hybrid cost function that primarily penalises the location of the base rather than using $Q_{stabilising}$[1]
3. a more detailed study of the implications of choosing/shaping the controller strength c, including time, state or spatial location dependence,
4. impact of different levels of beam compliance, or mass distribution
5. the controller's robustness to modelling errors and disturbances introduced during the simulation
6. the extension of this problem to match a target orientation of the tip (instead of simply being upright), and therefore allowing end-effector control in soft robotic manipulators.

[1]This was briefly explored. While the base location is easily penalised under the current formulation, it was extremely important to balance the relative weight of the penalties - if the penalty is too small, the controller establishes local minima where the beam is upright but far from the origin, and if the penalty is too large, the optimiser often fails to find a solution. The simulation times were also much greater than the other simulations, and therefore more work is needed to understand if this is a suitable approach.

Appendix A

Code: Rigid Pendulum Optimal Control

The rigid pendulum optimal control problem was solved using `OpenGoddard` which implements a Legendre-Psuedospectral scheme. It was chosen for its simple Python interface. The code is listed below.

```
import numpy as np
from OpenGoddard.optimize import Problem, Guess, Condition, Dynamics

class Beam:
  """
  Defines a simple rigid beam of uniform mass distribution.
  """
  def __init__(self):
    self.mu = 1 # mass per unit length
    self.L = 1
    self.M = self.mu*self.L
    self.g = 9.81
    self.I = self.mu * self.L**3 / 3
    self.copt = np.sqrt(6*self.g)

def dynamics(prob, obj, section):
  th = prob.states(0, section)
  thd = prob.states(1, section)
  F1 = prob.controls(0,section)

  dx = Dynamics(prob, section)
  dx[0] = thd
  dx[1] = (obj.M * obj.g * obj.L * np.sin(th) / 2  + F1) / obj.I

  return dx()

def equality(prob, obj):
  th = prob.states_all_section(0)
  thd = prob.states_all_section(1)

  result = Condition()

  # initial conditions
  result.equal(th[0], np.pi)
  result.equal(thd[0], 0)

  # final conditions
  # (only used for the minumum torque case)
  # result.equal(th[-1], 0.0)
  # result.equal(thd[-1], 0.0)

  return result()

def inequality(prob, obj):
  tf = prob.time_final(-1)

  result=Condition()

  # lower bounds
  result.lower_bound(tf, 2)
  result.upper_bound(tf, 10)

  return result()

def cost(prob, obj):
  return 0.0

def running_cost(prob, obj):

  th = prob.states_all_section(0)
  thd = prob.states_all_section(1)
  F1 = prob.controls_all_section(0)

  c = obj.copt

  return (np.sin(th/2) + thd/c)**2 + 1e-6*F1**2

def display_func():
  tf = prob.time_final(-1)
  print(f"tf: {tf:.5f}")
  return

### define problem parameters
time_init = [0.0, 4.0]
n = [100]
num_states = [2]
num_controls = [1]
max_iteration = 80

prob = Problem(time_init, n, num_states, num_controls, max_iteration)
obj = Beam()

prob.dynamics = [dynamics]
prob.knot_states_smooth = []
prob.cost = cost
prob.running_cost = running_cost
prob.equality = equality
prob.inequality = inequality

# define initial guess
zeroGuess = Guess.linear(prob.time_all_section, 0,0)
thGuess = Guess.linear(prob.time_all_section,np.pi, 0)
prob.set_states_all_section(0, thGuess)
prob.set_states_all_section(1, zeroGuess)

## solve the optimal control problem
prob.solve(obj, display_func, ftol=1e-8)

## save the solution
th = prob.states_all_section(0)
thd = prob.states_all_section(1)
F1 = prob.controls_all_section(0)
time = prob.time_update()

np.savez('output', th=th,thd=thd,F1=F1,t=time)
```

Appendix B

Full Simulation Results

Table B.1: Summary of simulation results

Sim Index	c	f_s	f_o	Modified Q_o?	Tolerance	V_u	$V_{\text{uprightness}}$	$V_{t,upright}$	$V_{\text{rectangle}}$	V_{maxDist}	$V_{\text{t, sim1}}$	$V_{\text{t, sim avg}}$
5	10	0	1	True	0.01	1.08	0.947	0.61	17.22	4.14	55.79	1.66
9	10	0	1	True	0.001	1.47	0.998	0.60	645.69	44.14	433.13	3.30
15	0.1	0.01	1	True	0.001	2.95	0.962	1.13	36.05	8.49	812.96	12.08
13	1	0.01	1	True	0.001	2.91	0.962	1.12	35.05	8.36	1050.37	13.32
16	1	0.01	1	False	0.001	1.56	0.951	0.45	6.11	2.43	202.78	2.83
18	7.672	0.01	1	True	0.001	1.56	0.970	0.64	9.78	3.36	224.46	8.24
1	10	0.01	1	True	0.01	1.12	0.943	0.59	14.89	3.82	87.93	1.66
12	10	0.01	1	True	0.001	1.32	0.968	0.59	7.56	3.34	309.96	6.75
17	10	0.01	1	False	0.001	1.47	0.935	0.42	3.06	2.67	118.31	3.21
14	30	0.01	1	True	0.001	1.98	0.972	0.39	5.25	3.65	1043.37	8.66
2	10	0.1	1	True	0.01	1.48	0.961	0.57	4.26	2.71	125.72	1.86
11	10	0.1	1	True	0.001	1.75	0.958	0.58	2.43	2.63	168.83	10.05
3	10	1	1	True	0.01	1.80	0.942	0.46	1.76	2.27	143.40	2.13
6	10	1	0	True	0.01	1.90	0.920	0.37	1.67	2.19	144.71	1.97
10	10	1	0	True	0.001	2.09	0.919	1.99	1.66	2.18	503.61	4.97
4	10	10	1	True	0.01	1.97	0.937	0.45	1.75	2.20	508.91	3.00

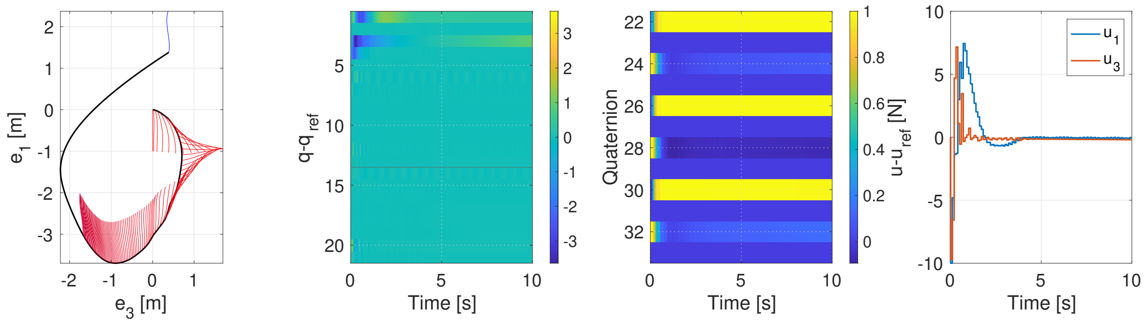


Figure B.1: Summary Image for Simulation 1

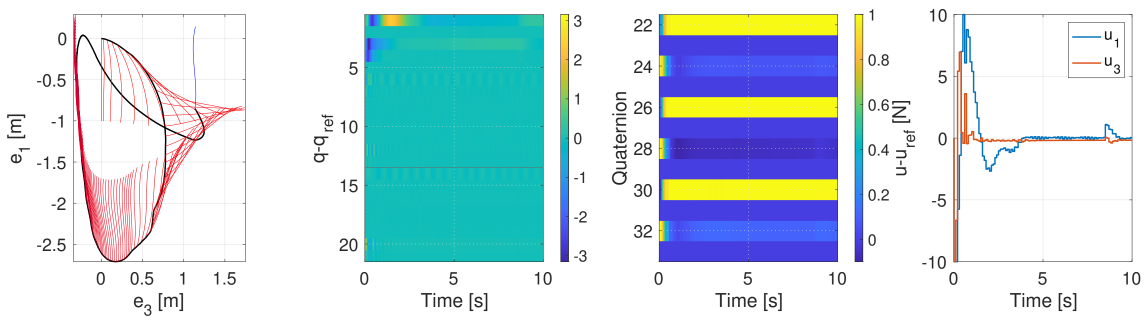


Figure B.2: Summary Image for Simulation 2

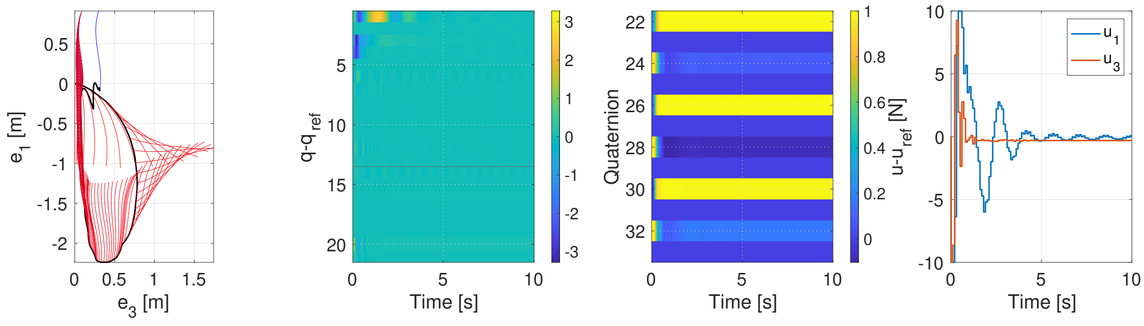


Figure B.3: Summary Image for Simulation 3

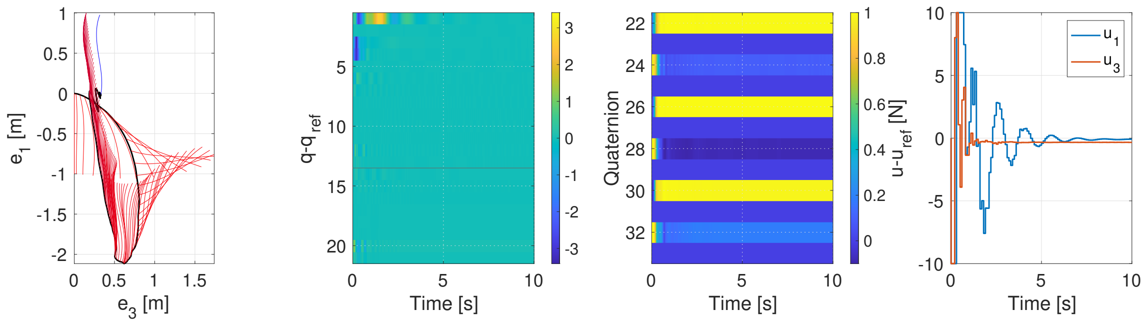


Figure B.4: Summary Image for Simulation 4

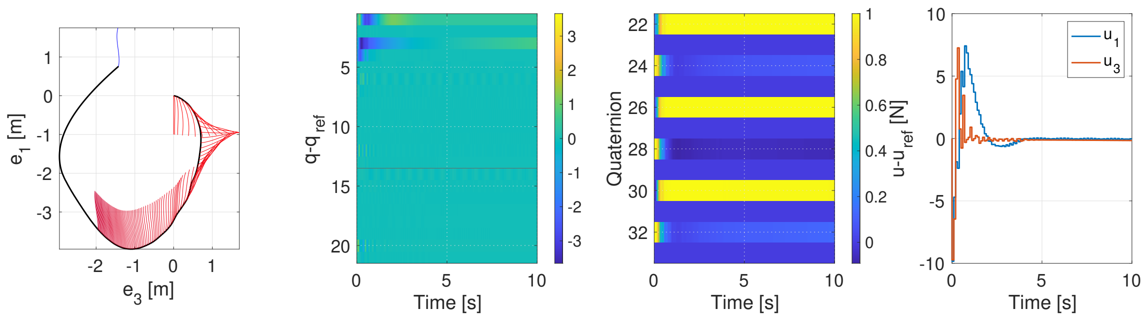


Figure B.5: Summary Image for Simulation 5

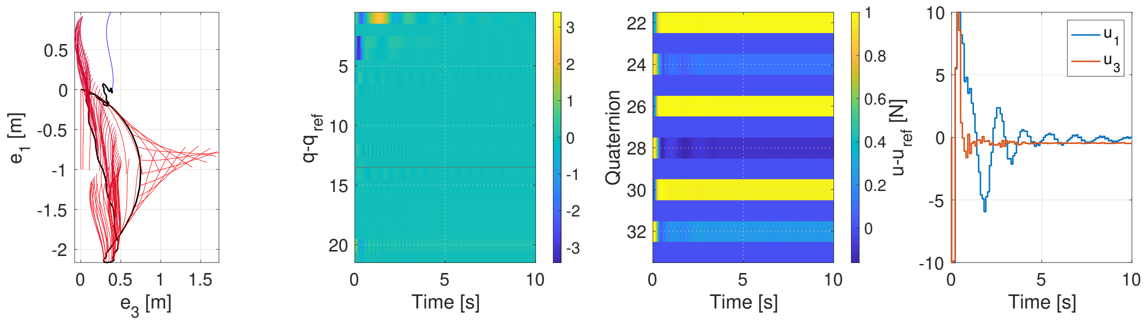


Figure B.6: Summary Image for Simulation 6

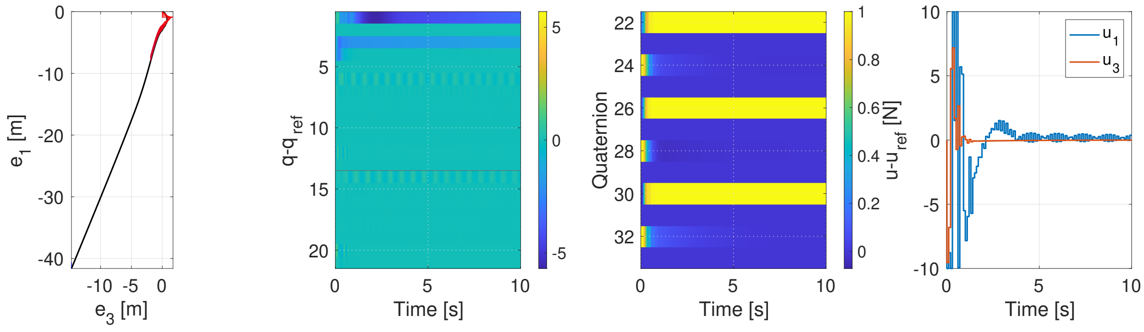


Figure B.7: Summary Image for Simulation 9

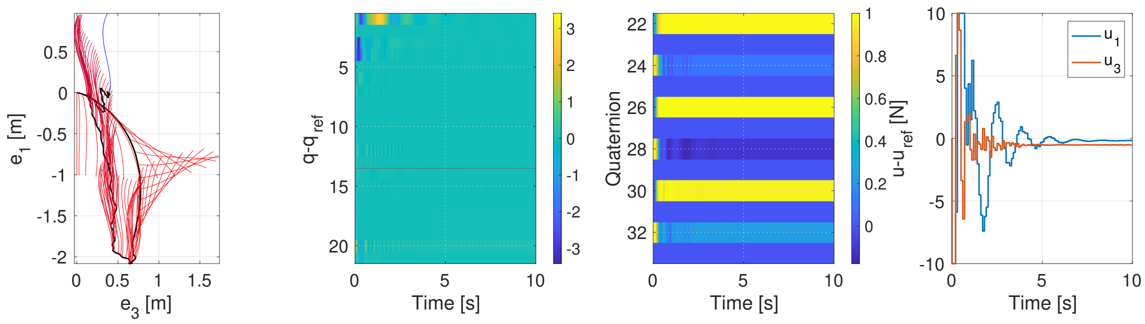


Figure B.8: Summary Image for Simulation 10

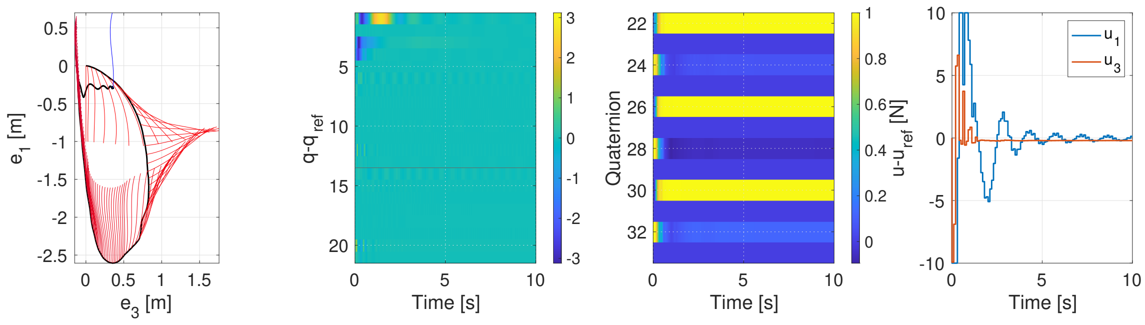


Figure B.9: Summary Image for Simulation 11

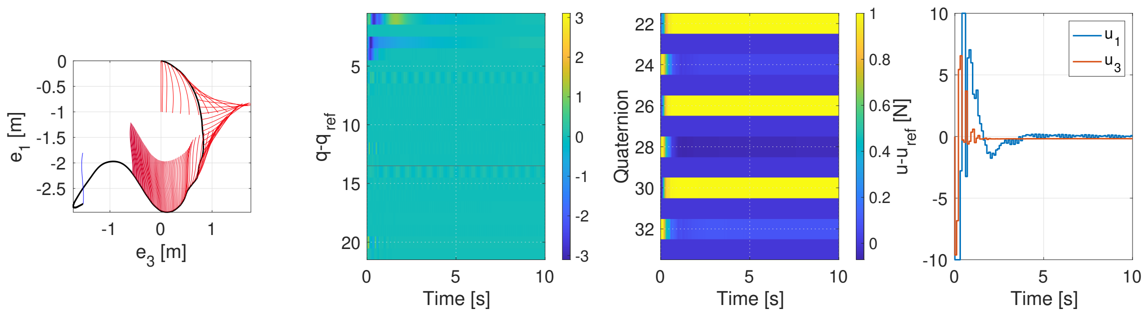


Figure B.10: Summary Image for Simulation 12

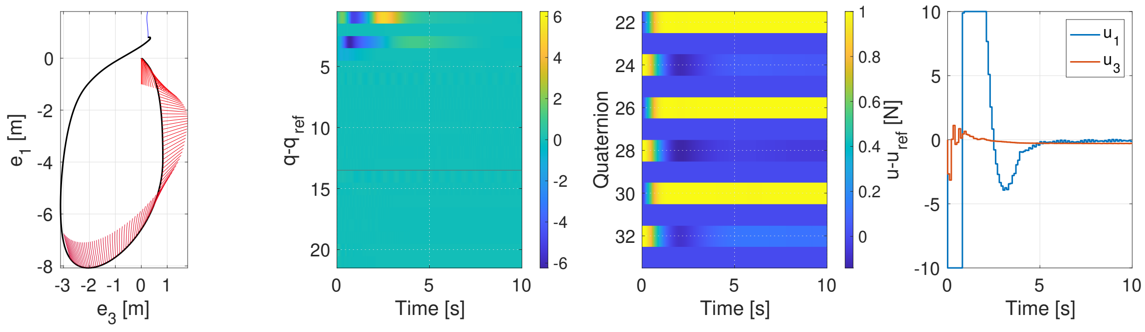


Figure B.11: Summary Image for Simulation 13

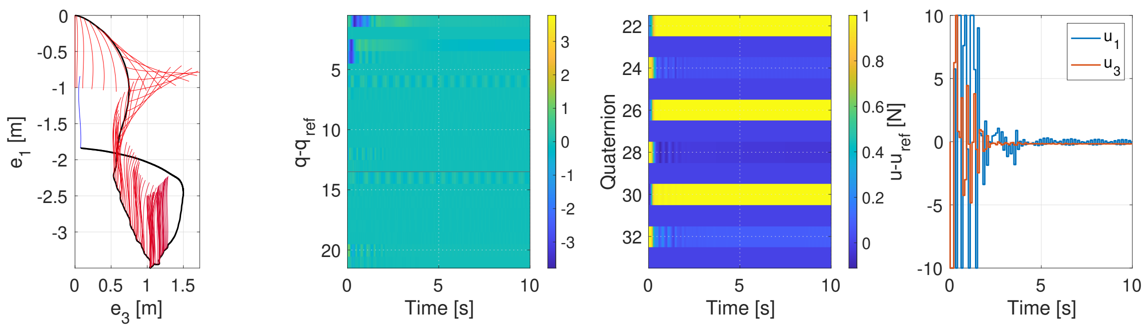


Figure B.12: Summary Image for Simulation 14

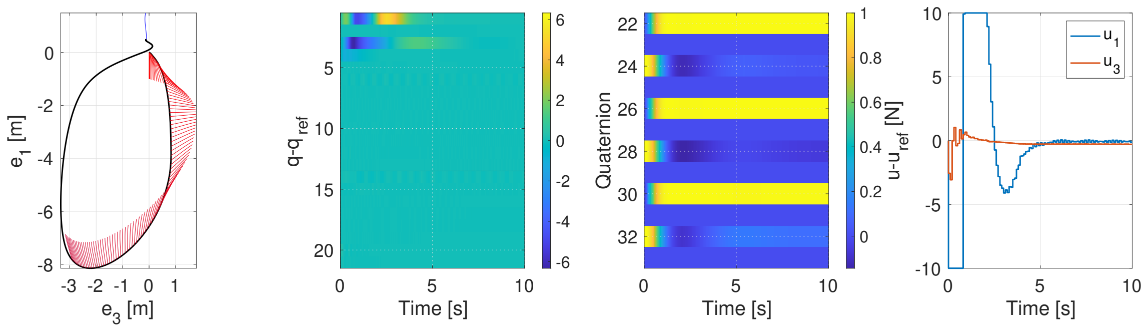


Figure B.13: Summary Image for Simulation 15

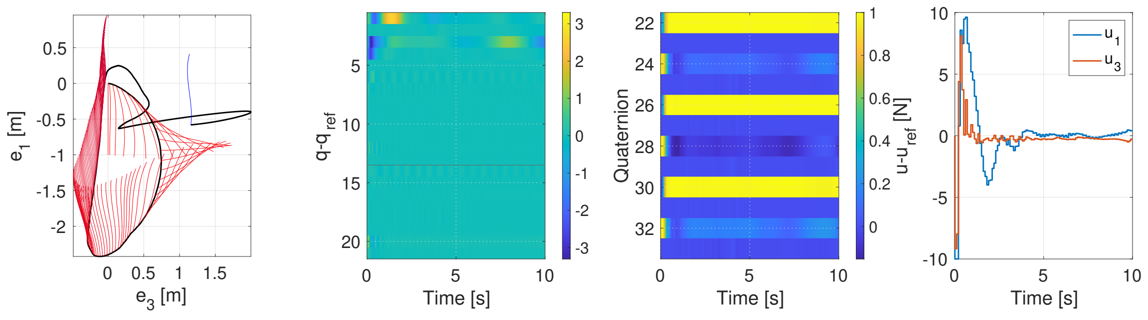


Figure B.14: Summary Image for Simulation 16

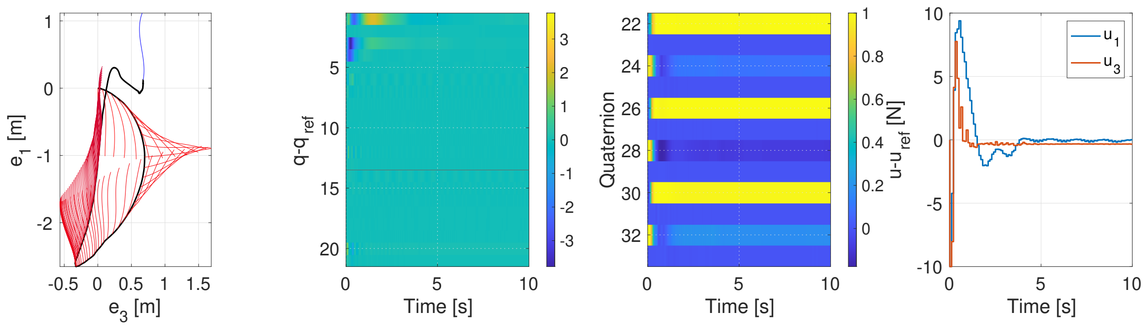


Figure B.15: Summary Image for Simulation 17

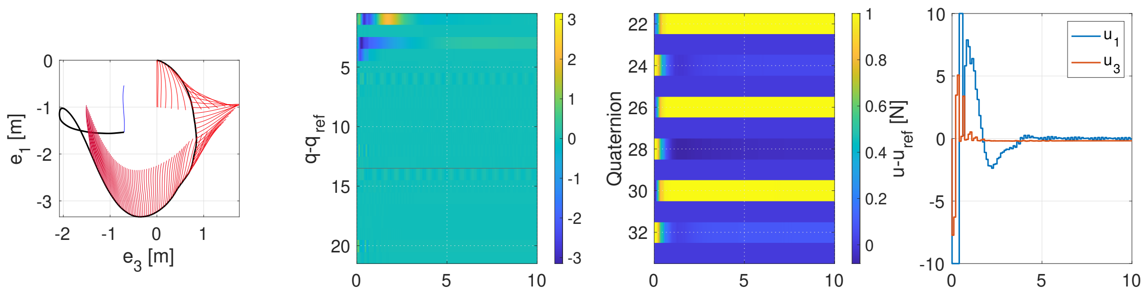


Figure B.16: Summary Image for Simulation 18

Appendix C

Catmull-Rom Interpolation for quaternions

The derivation of the Catmull Rom interpolation extends `https://qroph.github.io/2018/07/30/smooth-paths-using-catmull-rom-splines.html` and `https://en.wikipedia.org/wiki/Cubic_Hermite_spline#Interpolation_on_the_unit_interval_with_matched_derivatives_at_endpoints`

Any polynomial spline (parameterised by t) through a set of M N-dimensional points can be written as

$$p_j(t) = w_i(t)p_{ij} \tag{C.1}$$

using Einstein notation for $i = 0, 1, ...M$ and $j = 0, 1, 2...N$

A Catmull-Rom interpolation uses a cubic spline, with matched points and derivatives at each control point. Therefore we can write the spline between any two points p_1, p_2 as a weighted sum with the neighbouring points,

$$p(t) = a(t)p_0 + b(t)p_1 + c(t)p_2 + d(t)p_3 \tag{C.2}$$

for $t \in [0, 1]$ where we need

$$p(0) = p_1 \tag{C.3}$$
$$p(1) = p_1 \tag{C.4}$$
$$m(0) = \frac{p_2 - p_0}{2} \tag{C.5}$$
$$m(1) = \frac{p_3 - p_1}{2} \tag{C.6}$$

where m represents the derivative at the central knot points.

Solving for the coefficients a, b, c, d we can write

$$p(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ -s & 0 & s & 0 \\ 2s & s-3 & 3-2s & -s \\ -s & 2-s & s-2 & s \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \tag{C.7}$$

For a centripetal Catmull Rom, we use $s = 1/2$, therefore

$$p(t) = \begin{bmatrix}1 & t & t^2 & t^3\end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 1 & -5/2 & 2 & -1/2 \\ -1/2 & 3/2 & -3/2 & 1/2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \tag{C.8}$$

$$p(t) = \frac{1}{2} \begin{bmatrix} -t^3 + 2t^2 - t \\ 3t^3 - 5t^2 + 2 \\ -3t^3 + 4t^2 + t \\ t^3 - t^2 \end{bmatrix}^T \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \tag{C.9}$$

which can be easily used when all four knot points are known. However in the boundaries, the p_0 or p_3 are not known. These can arbitrarily be chosen, although it will change the shape of the curve.

One possible assumption we can make is that $p_0 = p_1$ on the left boundary and $p_3 = p_2$ on the right. This gives us a modified linear combination at the boundaries

$$p_{\text{left}}(t) = (a + b)p_1 + cp_2 + dp_3 \tag{C.10}$$

$$p_{\text{right}}(t) = ap_0 + bp_1 + (c + d)p_2 \tag{C.11}$$

However this does not treat the boundaries well. Instead, we can define the points as $p_0 = 3p_1 - 3p_2 + p_3$ on the left and $p_3 = 3p_2 - 3p_1 + p_0$ on the right. This gives us

$$p_{\text{left}}(t) = (3a + b)p_1 + (-3a + c)p_2 + (a + d)p_3 \tag{C.12}$$

$$p_{\text{right}}(t) = (a + d)p_0 + (b - 3d)p_1 + (c + 3d)p_2 \tag{C.13}$$

Both methods seem to be rather similar, but the second one is more accurate.

The extension to multi dimensional p is straightforward. Demonstrating the 2D case,

$$p_A(t) = a(t)p_{A0} + b(t)p_{A1} + c(t)p_{A2} + d(t)p_{A3} \tag{C.14}$$

$$p_B(t) = a(t)p_{B0} + b(t)p_{B1} + c(t)p_{B2} + d(t)p_{B3} \tag{C.15}$$

where A, B represent components of each dimension of p. Therefore when we stack these together, we get

$$\begin{bmatrix} p_A \\ p_B \end{bmatrix}(t) = \begin{bmatrix} a & 0 & b & 0 & c & 0 & d & 0 \\ 0 & a & 0 & b & 0 & c & 0 & d \end{bmatrix} \begin{bmatrix} p_{A0} \\ p_{B0} \\ p_{A1} \\ p_{B1} \\ p_{A2} \\ p_{B2} \\ p_{A3} \\ p_{B3} \end{bmatrix} \tag{C.16}$$

Therefore in general we have for a d dimensional point p,

$$p(t) = \underbrace{\begin{bmatrix} a(t)I_d & b(t)I_d & c(t)I_d & d(t)I_d \end{bmatrix}}_{S(t)} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \tag{C.17}$$

where the coefficients get modified at the edges appropriately.

Bibliography

[1] Thomas E Noll, John M Brown, Marla E Perez-Davis, Stephen D Ishmael, Geary C Tiffany, and Matthew Gaier. Investigation of the Helios prototype aircraft mishap Volume I Mishap Report. 2004.

[2] Cecilia Laschi, Barbara Mazzolai, and Matteo Cianchetti. Soft robotics: Technologies and systems pushing the boundaries of robot abilities. *Sci. Robot*, 1(1):eaah3690, 2016.

[3] Fumiya Iida and Cecilia Laschi. Soft robotics: Challenges and perspectives. *Procedia Computer Science*, 7:99–102, 2011.

[4] Enrico Franco and Timothy Brown. Energy shaping control for robotic needle insertion. *2019 23rd International Conference on System Theory, Control and Computing, ICSTCC 2019 - Proceedings*, pages 436–441, 2019.

[5] Vincent Duindam, Jijie Xu, Ron Alterovitz, Shankar Sastry, and Ken Goldberg. Three-dimensional motion planning algorithms for steerable needles using inverse kinematics. *The International Journal of Robotics Research*, 29(7):789–800, 2010.

[6] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832).* Russ Tedrake, 12 June 2020.

[7] Mark W Spong. Swing up control of the acrobot using partial feedback linearization. *IFAC Proceedings Volumes*, 27(14):833–838, 1994.

[8] Mark W Spong. Energy based control of a class of underactuated mechanical systems. *IFAC Proceedings Volumes*, 29(1):2828–2832, 1996.

[9] Yue-Qing Yu, Larry L Howell, Craig Lusk, Ying Yue, and Mao-Gen He. Dynamic modeling of compliant mechanisms based on the pseudo-rigid-body model. 2005.

[10] Santosha Kumar Dwivedy and Peter Eberhard. Dynamic analysis of flexible manipulators, a literature review. *Mechanism and machine theory*, 41(7):749–777, 2006.

[11] Prasanna S Gandhi, Pablo Borja, and Romeo Ortega. Energy shaping control of an inverted flexible pendulum fixed to a cart. *Control Engineering Practice*, 56:27–36, 2016.

[12] J. C. Simo. A finite strain beam formulation. The three-dimensional dynamic problem. Part I. *Computer Methods in Applied Mechanics and Engineering*, 49(1):55–70, 1985.

[13] Rafael Palacios. Nonlinear normal modes in an intrinsic theory of anisotropic beams. *Journal of Sound and Vibration*, 330(8):1772–1792, 2011.

[14] Marc Artola, Andrew Wynn, and Rafael Palacios. A nonlinear modal-based framework for low computational cost optimal control of 3D very flexible structures. *2019 18th European Control Conference, ECC 2019*, pages 3836–3841, 2019.

[15] Nicola Fonzi, Steven L Brunton, and Urban Fasel. Data-driven nonlinear aeroelastic models of morphing wings for control. *arXiv preprint arXiv:2002.03139*, 2020.

[16] Marc Artola, Andrew Wynn, and Rafael Palacios. Modal-Based Nonlinear Model Predictive Control for 3D Very Flexible Structures. Under review in IEEE Transactions on Automatic Control.

[17] Andrew Hanson. *Visualizing quaternions.* Morgan Kaufmann series in interactive 3D technology. Morgan Kaufmann, Elsevier Science, Amsterdam ; London, 2006.

[18] Takahiro Inagawa and Kazuki Sakaki. OpenGoddard. Available at `https://github.com/istellartech/OpenGoddard`.

[19] Dewey H. Hodges. Erratum: Geometrically Exact, Intrinsic Theory for Dynamics of Curved and Twisted Anisotropic Beams. *AIAA Journal*, 47(5):1308–1309, 2009.

[20] David Acheson. *From calculus to chaos: An introduction to dynamics.* Oxford University Press on Demand, 1997.