UNIVERSITY of CALIFORNIA

SANTA CRUZ

OPTIMAL CONTROL OF A NONCIRCULAR WHEEL

A thesis submitted in partial satisfaction of the

requirements for the degree of

MASTER OF SCIENCE

in

SCIENTIFIC COMPUTING & APPLIED MATHEMATICS

by

Paul K. Wintz

September 2020

The thesis of Paul K. Wintz is approved:

Professor Qi Gong, Chair

Professor Hongyun Wang

Professor Dongwook Lee

Quentin Williams
Interim Vice Provost and Dean of Graduate Studies

Contents

List of Figures

Abstract

Optimal Control of a Noncircular Wheel

by

Paul K. Wintz

A model is developed of a non-circular wheel rolling on a road and colliding with a short wall. It is modeled as a hybrid dynamical system with a combination of continuous- and discrete-time dynamics. Input-output feedback linearization is used to design a controller to stabilize the wheel via the application of torque such that the wheel is balanced on the wall. A proof of asymptotic stability is provided for the feedback linearized control. To compute an optimal control signal, the problem is discretized into a finite-dimensional constrained optimization problem and solved with a numerical solver. An elliptical wheel is presented as an example and comparisons between various implementations are provided.

1

Introduction

In this paper, we examine the dynamics and control of a noncircular wheel. In particular, we will consider a smooth convex wheel rolling over a flat road until it collides with a short wall, as shown in Figure 1.1. Our goal, then, is to choose the torque applied to the wheel such that it pivots to the top of the wall with minimal effort. This system has a combination of continuous-time dynamics, while the wheel is rolling on the road or pivoting on the wall, and a discrete event when the wheel impacts the wall. Thus, it lends itself to being modeled as a hybrid dynamical system and provides an opportunity to investigate the optimal control of a nonlinear hybrid system.

Methods for solving optimal control problems are divided between between direct and indirect methods [1]. Direct methods attempt to approximate the solution by

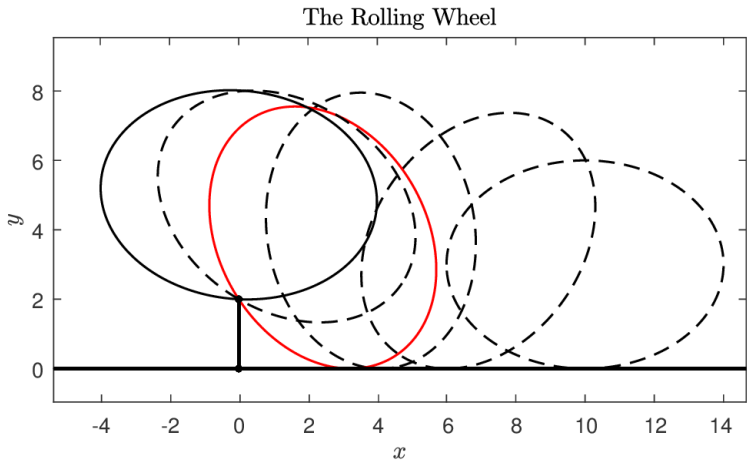


Figure 1.1: An elliptical wheel rolling from right to left at a series of time steps. The wheel is drawn in red at the time of impact with the wall.

discretizing the problem into a finite-dimensional optimization problem that can be solved with methods for numerical optimization. Direct methods have previously been applied to hybrid optimal control problems [5]. In contrast, indirect methods approach the problem by generalizing Hamiltonian-based methods used in calculus of variations. In particular, a famous result from optimal control theory is Pontryagin's maximum principle (PMP), which gives necessary conditions for solutions to optimal control problems. PMP was later generalized to hybrid systems [2]. In this work, however, we will utilize a direct method.

We begin, in Chapter 2, by formulating a mathematical model for the wheel and road. The system is modeled as a hybrid dynamical system to allow for continuous- and discrete-time dynamics. To that end, Section 2.1 introduces fundamental concepts of hybrid dynamical systems that are necessary for the following work. The details of the model for a generic wheel are developed in Section 2.2 and an example

of an elliptical wheel is presented in Section 2.3.

In Chapter 3, we move to the development of a controller. We use input-output linearization, in Section 3.2, to design analytically a non-optimal controller that moves the wheel toward the desired configuration. Next, in Section 3.1, we design the cost function and constraints of the the optimal control problem such that solutions satisfy the hybrid dynamics, reach a desired set point at a given time, and uses minimal energy to control the wheel. Section 3.3 describes how we discretize the optimal control problem to convert it into a finite-dimensional optimization problem, then solve. We investigate several methods for accelerating the calculation of solutions, then return to the example of the elliptical wheel.

Appendix A includes results relating to the geometry of a rotated ellipse, which are used in the example of the elliptical wheel, and Appendix B provides details on the calculation the Jacobian of the cost and constraint functions for the optimization problem.

2

Modeling of a Noncircular Wheel

In this chapter, we derive a mathematical model for a noncirclular wheel on a road with a wall of height $h_0 > 0$. Without loss of generality, we choose to position the wall at $x = 0$ and the wheel to the right of the wall. The road has height zero everywhere except at $x = 0$. Mathematically, the road $h : \mathbb{R} \to \mathbb{R}$, is given as

$$h(x) = \begin{cases} 0, & x \neq 0 \\ h_0, & x = 0. \end{cases}$$

When the wheel impacts the wall, we model the collision as perfectly inelastic, so the component of the velocity directed toward the wall is lost in the impact. As a result, the wheel does not bounce off the wall. We assume that the mass of the wheel is negligible in comparison to the mass of the vehicle. Hence, in our model the wheel's mass is concentrated at the axle rather than distributed throughout the

wheel. We also assume that the wheel does not deform; there is sufficient friction between the wheel and the ground to prevent slipping; and the wheel is strictly convex. We impose convexity for two reasons. The first is that that it is reasonable from an engineering point of view: building a non-convex wheel capable of supporting a vehicle without deformation is impractical. The second reason is that convexity simplifies the problem by eliminating certain undesirable behaviors, such as the wheel having multiple contact points with a flat section of road, which would cause discontinuous rolling on flat ground (thus, discontinuous behavior occurs only at the wall).

To model the dynamics of a rolling wheel, we use a hybrid dynamical system. As described in [3], hybrid systems include both continuous-time *flows* and discrete-time *jumps*. In our system, flows correspond to intervals when the wheel is either rolling over a flat portion of the road or pivoting around the corner of the wall. A single discrete event occurs when the wheel impacts the wall.

2.1 Preliminary Hybrid Systems Theory

Before we model our system, we provide a short introduction to necessary concepts and vocabulary relating to hybrid dynamical systems. A hybrid system $\mathcal{H}$ is specified by its *flow set* C, *jump set* D, *flow map*[1] f, and *jump map* g. When the state

[1] In some fields, the flow map is called the *flow vector field*.

of the system is in C, the system evolves according to continuous-time dynamics $\dot{\mathbf{x}} = f(\mathbf{x})$. If $\mathbf{x} \in D$, then the state jumps from $\mathbf{x}$ to $\mathbf{x}^+$ according to $\mathbf{x}^+ = g(\mathbf{x})$. The full system is then written as

$$\mathcal{H} = \begin{cases} \dot{\mathbf{x}} = f(\mathbf{x}), & \text{if } \mathbf{x} \in C \\ \mathbf{x}^+ = g(\mathbf{x}), & \text{if } \mathbf{x} \in D. \end{cases}$$

Solutions to hybrid systems are given in terms of *hybrid time domains*.

Definition 2.1 (Hybrid time domains)**.** A set $E \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$ is a *compact hybrid time domain* if $E = \bigcup_{j=0}^{J-1} \big([t_j, t_{j+1}], j\big)$ for some finite sequence of times $0 = t_0 \leq t_1 \leq \cdots \leq t_J$. A set E' is called a *hybrid time domain* if for all $(T, J) \in E'$, $E \cap ([0, T] \times \{0, 1, 2, ..., J\})$ is a compact hybrid domain [3, Definition 2.3].

An interval $[t_j, t_{j+1}]$ such that $t_j < t_{j+1}$ is called an *interval of flow*. Conversely, if $(t, j) \in E$ and $(t, j + 1) \in E$, then (t, j) is a *jump time*.

An example of a hybrid time domain is $\big([0, t^+], 0\big) \cup \big([t^+, \infty), 1\big)$, with a jump time $t^+ \geq 0$. If $t^+ > 0$, then $[0, t^+]$ and $[t^+, \infty)$ are both intervals of flow.

Remark. A hybrid time $(t, j) \in E$ is uniquely identified by t except at jumps, so we often write $\mathbf{x}(t, j)$ as $\mathbf{x}(t)$ when there is no ambiguity.

Definition 2.2 (Hybrid Solution)**.** For hybrid system $\mathcal{H}$, a function $\mathbf{x} : E \to \mathbb{R}^n$ is a *solution* of $\mathcal{H}$ if E is a hybrid time domain, $\mathbf{x}(0, 0) \in \overline{C} \cup D$, and during every interval of flow then $\mathbf{x}(t, j) \in C$ and evolves according $\dot{\mathbf{x}}(t, j) = f(\mathbf{x}(t, j))$ and at every jump time (t, j), $\mathbf{x}(t, j) \in D$ and $\mathbf{x}(t, j + 1) = g(\mathbf{x}(t, j))$. For details,

see [3, Definition 2.6]. If E is unbounded, then $\mathbf{x}$ is called a *complete* solution [3, Defintion 2.5].

To classify the asymptotic behavior of the system, we define two more terms.

Definition 2.3. Given a hybrid system $\mathcal{H}$, the origin of the state space is said to be *pre-asymptotically stable* with regards to $\mathcal{H}$ if it is Lyapunov stable and every complete solution approaches the origin as $t + j \to \infty$. If every maximal solution to $\mathcal{H}$ is complete and the origin is pre-asymptotically stable, then the origin is *asymptotically stable* [3].

2.2 Modeling

We are now equipped to design a hybrid model of a rolling wheel. First, we choose two coordinate systems in $\mathbb{R}^2$ with axes shown in Figure 2.1. The first coordinate system is stationary and has the orthonormal basis $\mathcal{B}_0 = \{\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y\}$, where $\hat{\mathbf{e}}_y$ points in the vertical direction. The second coordinate system, with orthonormal basis $\mathcal{B}_\varphi = \{\hat{\mathbf{e}}_a, \hat{\mathbf{e}}_b\}$, is centered at the wheel and oriented to the wheel's angle of rotation, φ, as measured from $\hat{\mathbf{e}}_x$ to $\hat{\mathbf{e}}_a$ (we use the convention that counter-clockwise direction is positive). Unless otherwise stated, vectors are assumed to be written relative to $\mathcal{B}_0$. The wheel's mass, m, is concentrated at the origin of $\mathcal{B}_\varphi$ and the acceleration g due to gravity pulls in the negative $\hat{\mathbf{e}}_y$-direction.

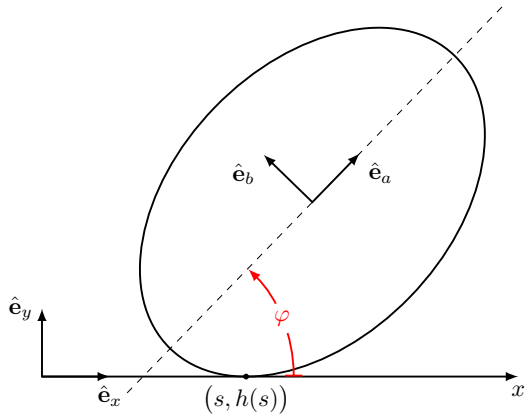


Figure 2.1: Coordinate systems $\mathcal{B}_0 = \{\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y\}$ and $\mathcal{B}_\varphi = \{\hat{\mathbf{e}}_a, \hat{\mathbf{e}}_b\}$ for a wheel with contact point $(s, h(s))$ and rotation angle φ.

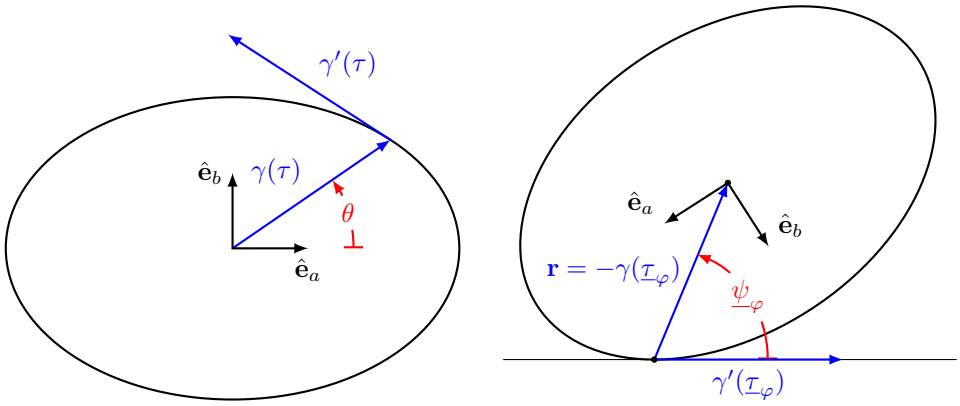
We define the wheel by a smooth curve $\gamma : \mathbb{R} \to \mathbb{R}^2$ with coordinates in basis $\mathcal{B}_\varphi$ given as $[\gamma(\tau)]_{\mathcal{B}_\varphi} = \big(\gamma_1(\tau), \gamma_2(\tau)\big)$. The derivative of γ with respect to τ is written $\gamma'(\tau)$. Vectors γ and γ' in the $\mathcal{B}_\varphi$ reference frame are shown in Figure 2.2a. We also define the horizontal and vertical components of γ relative to the stationary basis $\mathcal{B}_0$ as $x_\varphi(\tau)$ and $y_\varphi(\tau)$. That is, $[\gamma(\tau)]_{\mathcal{B}_0} = (x_\varphi(\tau), y_\varphi(\tau))$.

As mentioned above, we require that γ encloses a strictly convex set.[2] For the model to be physically plausible, we also assume that the center of mass is strictly inside the wheel.

We then define the state space of our system as $\mathcal{X} := \mathbb{R}^4 \times \{-1, 0, 1\}$ and the state vector as $\mathbf{x} := (\varphi, \omega, s, \tau, k) \in \mathcal{X}$ where $\varphi \in \mathbb{R}$ is the rotation angle,[3] $\omega \in \mathbb{R}$ is the angular velocity, $s \in \mathbb{R}$ is the x-coordinate of the contact point, $\tau \in \mathbb{R}$ is the parameter value such that $\gamma(\tau)$ points from the center of the wheel to the contact

[2] A set is called *strictly convex* if for every pair of points in the set, the line segment between them is in the interior of the set.

[3] For φ, we do not identify multiples of 2π. That is, if $\varphi_1 := 0$, and $\varphi_2 := 2\pi$, then $\varphi_1 \neq \varphi_2$.



(a) Diagram of γ and γ' in the wheel's coordinate frame.

(b) The wheel γ rotated and translated to align tangentially with the road.

Figure 2.2: The wheel γ in its original orientation and rotated and translated to align tangentially with the road.

point, and $k \in \{-1, 0, 1\}$ are modes of the system. When the wheel is rolling over flat ground $k = 0$, when the wheel is pivoting around the top of the wall, $k = 1$, and when $k = -1$, solutions terminate.

The initial state of the system is $\mathbf{x}_0 = (\varphi_0, \omega_0, s_0, \tau_0, k_0)$ with φ_0 and ω_0 arbitrary, and $s_0 > 0$ chosen such that the initial position of the wheel does not intersect with the wall $(0, h_0)$. The wheel is initially rolling so $k_0 = 0$ and the initial value of τ must satisfy $\tau_0 = \underline{\tau}_{\varphi_0}$ where

$$\underline{\tau}_\varphi = \underset{\tau}{\operatorname{argmin}} \; y_\varphi(\tau). \tag{2.1}$$

Due to the strict convexity of the wheel, $\underline{\tau}_\varphi$ is unique. When the wheel is at angle φ on a flat portion of road, then $\gamma(\underline{\tau}_\varphi)$ is the contact point between the wheel and the road.

In order to specify the dynamics of the wheel, it useful to define several more variables. We define $\mathbf{r}(\tau)$ to be the vector from the contact point to the center of the wheel, shown in Figure 2.2b. (If γ^* is the contact point on the wheel,[4] then $\mathbf{r} = -\gamma^*$.) We write the length of $\mathbf{r}$ as

$$r(\tau) = \|\mathbf{r}(\tau)\| . \tag{2.2}$$

We define the angle from $\hat{\mathbf{e}}_a$ to $\gamma(\tau)$, shown in Figure 2.2a, as

$$\theta(\tau) = \texttt{atan2}\left(\gamma_2(\tau), \gamma_1(\tau)\right), \tag{2.3}$$

where `atan2` is the 2-argument arctangent function. We also define the angle from $\hat{\mathbf{e}}_x$ to $\mathbf{r}$ as ψ. Angle addition provides the following identity,

$$\psi \equiv \varphi + \theta - \pi. \tag{2.4}$$

Equation (2.4) holds for all ψ, φ, θ, even if the choice of values does not cause the wheel to align tangentially with the road. Thus, we define $\underline{\psi}_{\varphi}$, shown in Figure 2.2b, to be the value of ψ such that the bottom of the wheel contacts the road tangentially when the wheel is at angle φ (assuming the necessary translation). For this to occur, the vectors γ' and $\hat{\mathbf{e}}_x$ must align. Hence, $\underline{\psi}_{\varphi}$ is the angle from $\gamma'(\underline{\tau}_{\varphi})$ to $-\gamma(\underline{\tau}_{\varphi})$ and,

[4] If the wheel is on flat road, then $\gamma^* = \gamma(\underline{\tau}_{\varphi})$, but this is not necessarily true while the wheel is pivoting.

from the cosine dot product formula, we find

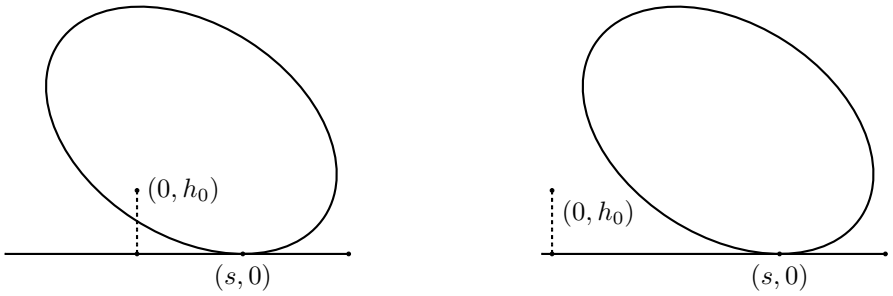
$$\underline{\psi}_{\varphi} = \cos^{-1}\left(\frac{-\gamma'(\underline{\tau}_{\varphi}) \cdot \gamma(\underline{\tau}_{\varphi})}{\left\|\gamma(\underline{\tau}_{\varphi})\right\| \left\|\gamma'(\underline{\tau}_{\varphi})\right\|}\right). \tag{2.5}$$

2.2.1 Jump and Flow Sets

The jump set D is chosen to detect when the wheel impacts the wall. That is, if $\mathbf{x} \in D$ then the wheel has collided with the wall. To construct D, we define an indicator function

$$\mathfrak{I}(\mathbf{x}) = \begin{cases} 1, & \text{if } (0, h_0) \text{ is at or above the lower edge of the wheel in state } \mathbf{x}. \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

With this definition, if the wheel has collided with the wall, then $\mathfrak{I}(\mathbf{x}) = 1$, otherwise, $\mathfrak{I}(\mathbf{x}) = 0$, as shown in Figure 2.3.



$\mathfrak{I}(\mathbf{x}) = 1$ $\qquad$ $\mathfrak{I}(\mathbf{x}) = 0$

Figure 2.3: The indicator function has value 1 if the wheel has collided with the wall and 0 otherwise.

The wheel can only impact the wall if it is rolling and $\mathfrak{I}(\mathbf{x}) = 1$, so the jump set is $D := \{\mathbf{x} \in \mathcal{X} \mid k = 0, \mathfrak{I}(\mathbf{x}) = 1\}$. Conversely, the flow set will consist of any state

where the wheel is rolling ($k = 0$), the wheel is not in contact with the wall ($\mathfrak{I}(\mathbf{x}) = 0$) and the contact point is tangentially aligned with the road ($\tau = \underline{\tau}_\varphi$), or the wheel is already pivoting ($k = 1$). Thus, $C := \left\{\mathbf{x} \in \mathcal{X} \;\middle|\; k = 0, \mathfrak{I}(\mathbf{x}) = 0, \tau = \underline{\tau}_\varphi\right\} \cup \{\mathbf{x} \in \mathcal{X} \mid k = 1\}$.

An attentive read might question whether there are some discrete events that are unaccounted for. In particular, what happens if the wheel is pivoting and rotates far enough that it collides with the road on either side of the wall? Such events are physically possible but our results in Chapter 3 show that the optimal control keeps the wheel far from colliding with the road, so we omit such cases from our model for the sake of simplicity.

2.2.2 Flow map

Next, we determine the continuous dynamics of the wheel. By definition, the angular velocity of the wheel is $\omega \equiv \dot{\varphi}$. We can compute $\dot{\omega}$ from conservation of energy because the system is free of dissipative forces. Let u be the torque with corresponding power

$$P = u\omega. \tag{2.7}$$

The total energy of the system is $E = K + V$, where K is the kinetic energy and

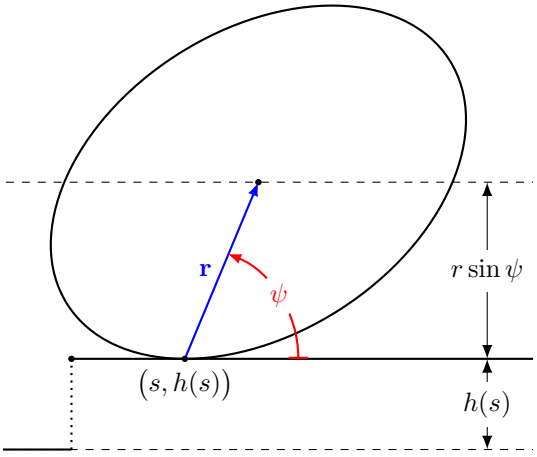


Figure 2.4: The height of the center of mass is the sum of $h(s)$ and $r\sin(\psi)$.

V is the potential energy. Energy conservation gives that

$$\dot{E} = \dot{K} + \dot{V} = P. \tag{2.8}$$

The mass of the wheel is concentrated at the center of mass, so we use the formula for kinetic energy of a point-mass $K = \frac{m}{2}(r\omega)^2$. We then find $\dot{K}$ to be

$$\dot{K} = mr\omega(\dot{r}\omega + r\dot{\omega}). \tag{2.9}$$

Similarly, the potential energy is $V = mg\left(r\sin(\psi) + h(s)\right)$, where $h(s)$ and $r\sin(\psi)$, illustrated Figure 2.4, are the heights of the contact point and the center of mass above the contact point. The value of $\dot{V}$ is then

$$\dot{V} = mg\left(r'(\varphi)\sin(\psi) + r\psi'(\varphi)\cos(\psi)\right)\omega, \tag{2.10}$$

where $r'(\varphi) := \frac{dr}{d\varphi}$ and $\psi'(\varphi) := \frac{d\psi}{d\varphi}$. Expressions for $r'(\varphi)$ and $\psi'(\varphi)$ can be calculated from Equations (2.1), (2.2), (2.4) and (2.5). Details for the general case are

omitted but results for an elliptical wheel are given in Section 2.3. By substitution of Equations (2.7), (2.9) and (2.10) into Equation (2.8) and solving for $\dot{\omega}$, we find

$$\dot{\omega} = \frac{u}{mr^2} - \frac{g}{r^2}\left(r'(\varphi)\sin(\psi) + r\psi'(\varphi)\cos(\psi)\right) - \frac{\dot{r}\omega}{r}. \tag{2.11}$$

Then, because $\tau = \underline{\tau}_\varphi$ when rolling, we find $\dot{\tau} = \frac{d\underline{\tau}_\varphi}{d\varphi}\dot{\varphi} = \underline{\tau}'_\varphi\omega$ where $\underline{\tau}'_\varphi := \dfrac{d\underline{\tau}_\varphi}{d\varphi}$. The wheel does not slip, so the change in s equals the curve length of the portion of the wheel rolled,

$$\dot{s} = \|\gamma'(\tau)\|\,\dot{\tau}. \tag{2.12}$$

We then have fully specified the continuous rolling dynamics

$$f_0(\mathbf{x}, u) = \begin{bmatrix} \omega \\ \dfrac{u}{mr^2} - \dfrac{g}{r^2}\left(r'(\varphi)\sin(\psi) + r\psi'(\varphi)\cos(\psi)\right) - \dfrac{\dot{r}\omega}{r} \\ \|\gamma'(\tau)\|\,\underline{\tau}'_\varphi\omega \\ \underline{\tau}'_\varphi\omega \\ 0 \end{bmatrix}.$$

When pivoting, the contact point is stationary, so $\dot{\tau} = \dot{s} = \dot{r} = 0$ and Equation (2.11) simplifies to

$$\dot{\omega} = \frac{1}{mr^2}(u - mgr\cos\psi). \tag{2.13}$$

Hence, the flow map while pivoting is

$$f_1(\mathbf{x}, u) := \begin{bmatrix} \omega \\ \frac{1}{mr^2}(u - mgr\cos\psi) \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

We can write the flow map in a unified expression for all $\mathbf{x} \in C$ as

$$\dot{\mathbf{x}} = f(\mathbf{x}, u) := \begin{cases} f_0(\mathbf{x}, u), & k = 0 \\ f_1(\mathbf{x}, u), & k = 1. \end{cases}$$

2.2.3 Jump map

When the wheel collides with the wall, it loses kinetic energy and either starts pivoting about the wall, or comes to a full stop. We construct the jump map $\mathbf{x}^+ = g(\mathbf{x})$ to capture these behaviors. Values immediately prior to impact are notated with superscript "$-$" and after the impact with superscript "$+$."

At the jump, the angle of the wheel is continuous, so $\varphi^+ = \varphi^-$. The location of the contact point jumps to the wall, which is at $x = 0$, hence $s^+ = 0$.

The angular velocity ω is discontinuous at the impact and the computation of ω^+ will consist of four steps:

1. Convert from angular to linear velocity.

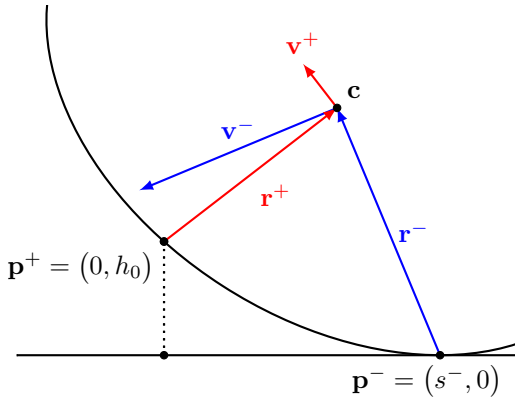


Figure 2.5: Vectors $\mathbf{p}^-, \mathbf{p}^+$, and $\mathbf{c}$ at the time of impact, along with linear velocities before $\mathbf{v}^-$ and after $\mathbf{v}^+$ and the radial vector before $\mathbf{r}^-$ and after $\mathbf{r}^+$. The velocity $\mathbf{v}^+$ is the orthogonal projection of $\mathbf{v}^-$ onto $\mathbf{r}^+$.

2. Decompose the linear velocity into components that are parallel and orthogonal to the radial vector from the new center of rotation to the center of mass.

3. Set the new linear velocity to the orthogonal component.

4. Convert back to rotational velocity.

The actual computation of ω^+ is as follows: Let $\mathbf{p}^- := (s^-, 0)$ be to the contact point immediately before impact, $\mathbf{p}^+ := (0, h_0)$ be the contact point after, and $\mathbf{c} := \mathbf{p}^- + \mathbf{r}^- = \mathbf{p}^+ + \mathbf{r}^+$ be the center of the wheel. By definition, $\mathbf{r}^-$ is the vector from $\mathbf{p}^-$ to $\mathbf{c}$ and $\mathbf{r}^+$ is the vector from $\mathbf{p}^+$ to $\mathbf{c}$. Thus, by vector addition we find $\mathbf{r}^+ = \mathbf{r}^- + \begin{bmatrix} s^- \\ -h_0 \end{bmatrix}$. Let $\mathbf{v}$ be the linear velocity of the center of mass. Converting rotational velocity to linear velocity, we find $\mathbf{v}^- = \omega^- \begin{bmatrix} -r_2^- \\ r_1^- \end{bmatrix}$. The component of $\mathbf{v}^-$ that is parallel to $\mathbf{r}^+$ is absorbed by the wall and the component perpendicular to $\mathbf{r}^+$ is preserved, as seen in Figure 2.5, so $\mathbf{v}^+ = \operatorname{orth}_{\mathbf{r}^+} \mathbf{v}^- = \mathbf{v}^- - \frac{\langle \mathbf{r}^+, \mathbf{v}^- \rangle}{\|\mathbf{r}^+\|^2} \mathbf{r}^+$. Then, $\omega^+ = \|\mathbf{v}^+\| \,/\, \|\mathbf{r}^+\|$ (the angular velocity is positive because the wheel is rotating

counter-clockwise at impact).

The value of τ jumps such that the contact point is on the corner of the wall. The computation of τ^+ will depend on the choice of γ.

If $\langle \mathbf{r}^-, \mathbf{r}^+ \rangle \leq 0$, then $\mathbf{v}^-$ is directed at or below the corner of the wall so the wheel comes to a full stop. We wish to exclude this case, so we set $k^+ = -1$, causing solutions to leave $C \cup D$ and terminate. Otherwise, we set $k^+ = 1$ indicating the wheel begins to pivot. We can unify these cases by writing

$$\mathbf{k}^+ = \operatorname{sgn}'(\langle \mathbf{r}^-, \mathbf{r}^+ \rangle) := \begin{cases} -1, & \text{if } \langle \mathbf{r}^-, \mathbf{r}^+ \rangle \leq 0 \\ 1, & \text{if } \langle \mathbf{r}^-, \mathbf{r}^+ \rangle > 0. \end{cases}$$

Collecting all the jump values, we have

$$\mathbf{x}^+ = g(\mathbf{x}) := \begin{bmatrix} \varphi \\ \|\mathbf{v}^+\| \,/\, \|\mathbf{r}^+\| \\ 0 \\ \tau^+ \\ \operatorname{sgn}'(\langle \mathbf{r}^-, \mathbf{r}^+ \rangle) \end{bmatrix}.$$

We write the hybrid plant with control law u as $\mathcal{H}_\gamma(u)$ or simply $\mathcal{H}_\gamma$ depending on whether the choice of control is relevant,

$$\mathcal{H}_\gamma(u) := \begin{cases} \dot{\mathbf{x}} = f(\mathbf{x}, u), & \mathbf{x} \in C := \left\{ \mathbf{x} \in \mathcal{X} \;\middle|\; k = 0, \mathfrak{I}(\mathbf{x}) = 0, \tau = \underline{\tau}_\varphi \right\} \cup \{ \mathbf{x} \in \mathcal{X} \mid k = 1 \} \\ \mathbf{x}^+ = g(\mathbf{x}), & \mathbf{x} \in D := \{ \mathbf{x} \in \mathcal{X} \mid k = 0, \mathfrak{I}(\mathbf{x}) = 1 \}. \end{cases} \tag{2.14}$$

2.3 Example: Elliptical wheel

For a concrete example, consider an ellipse with semi-axes a and b, given by $[\gamma(\tau)]_{\mathcal{B}_\varphi} = (a\cos\tau, b\sin\tau)$. We choose, without loss of generality, that $a \geq b$ (if $b > a$, the wheel can be rotated by $\frac{\pi}{2}$). Differentiation of γ produces $[\gamma'(\tau)]_{\mathcal{B}_\varphi} = (-a\sin\tau, b\cos\tau)$.

By Equations (2.2) and (2.3), we find $\theta = \texttt{atan2}(b\sin\tau, a\cos\tau)$ and the radius is $r = \sqrt{a^2\cos^2\tau + b^2\sin^2\tau}$. Differentiating r, we find $\dot{r} = \dfrac{b^2 - a^2}{r}(\cos\tau\sin\tau)\dot{\tau}$. Then, from Equation (2.12), $\dot{s} = \sqrt{a^2\sin^2(\tau) + b^2\cos^2(\tau)} \cdot \dot{\tau}$. We use Equation (A.3) to compute that $\underline{\tau}_\varphi = 3\pi \,/\, 2 - \texttt{atan2}\,(a\sin\varphi, b\cos\varphi)$ which has the derivative $\underline{\tau}'_\varphi = \frac{-ab}{a^2\sin^2(\varphi) + b^2\cos^2(\varphi)}$.

Next, we construct the indicator function $\mathfrak{I}(\mathbf{x})$. For an ellipse centered at the origin and rotated by angle φ, Equation (A.6) gives the height of the lower branch of the ellipse as a function of x. Shifting the variables to correspond with an ellipse centered at $(c_x, c_y) = (s, 0) + \mathbf{r}$, we find

$$y_\varphi(x) = C_y \sin\left(-\cos^{-1}\left(\frac{x - c_x}{C_x}\right) - \hat{\varphi}_x + \hat{\varphi}_y\right) + c_y, \tag{2.15}$$

with the constants $C_x, C_y, \hat{\varphi}_x, \hat{\varphi}_y$ defined in Lemma A.1. If $h_0 \geq y_\varphi(0)$, then the wheel has collided with the wall. Thus for the elliptical wheel the indicator function

is

$$\mathfrak{I}(\mathbf{x}) = \begin{cases} 1, & h_0 \geq y_\varphi(0, \mathbf{x}) \\ 0, & h_0 < y_\varphi(0, \mathbf{x}) \quad \text{or} \quad (0, \mathbf{x}) \notin \operatorname{dom} y_\varphi. \end{cases}$$

Finally, at the jump, τ^+ can be found via Equation (A.5): $\tau^+ = \cos^{-1}\left(c_x \,/\, C_x\right) - \hat{\varphi}_x$.

We compute solutions with the `HyEQsolver` function in the Hybrid Equations Toolbox [7]. Figure 1.1 shows the wheel at various times for $a = 4, b = 3, \varphi_0 = 0, \omega_0 = 10, s_0 = 10, m = 1, g = -9.8$, and $h_0 = 2.0$. The state of the wheel at the time of impact is shown in red.

3

Stabilization and Optimal Control

We now move from modeling the system to designing a controller that will produce desired system behavior. In particular, we want to use minimal control effort to move the wheel to top of the wall with the center of mass balanced directly above the wall.

First, we develop the optimal control problem formulation in Section 3.1. Then, in Section 3.2, we use input-output linearization to derive, analytically, a non-optimal controller that moves the wheel asymptotically toward the desired set-point. Finally, Section 3.3 describes our method for computing, numerically, an approximation of the optimal control.

3.1 Problem formulation

Our goal is to move the wheel from its initial state $\mathbf{x}_0$ onto the wall with the center of mass directly above the contact point $(0, h_0)$ and no angular velocity. Additionally, we want the wheel to arrive at the set point at a given time $T > 0$. For a particular $\mathbf{x}_0$, the desired configuration corresponds to a unique angle, φ^*, because the wheel rolls without slipping. Thus, we are interested in controlling the first two components of the state vector: $\varphi \to \varphi^*$ and $\omega \to 0$. To this end, we define an output vector, $\mathbf{y}(\mathbf{x}) = (y_1, y_2) := (\varphi - \varphi^*, \omega)$ and we want $\|\mathbf{y}(\mathbf{x}(T))\| = 0$. Additionally, at the set point, $\mathbf{r}$ points directly upwards, so $\psi = \psi^* := \frac{\pi}{2}$.

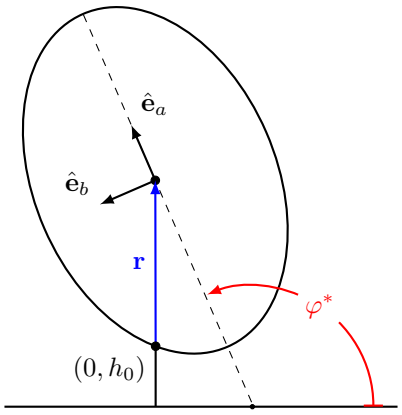


Figure 3.1: The wheel at the set-point φ^* with the center of mass directly above $(0, h_0)$.

To calculate the set point φ^* for a given initial condition $\mathbf{x}_0$, we modify the initial condition by setting $\omega_0 > 0$ and we modify system $\mathcal{H}_\gamma$ by making $\dot{\omega} = 0$. We then propagate the dynamics of the modified system from $\mathbf{x}_0$ using `HyEQsolver`

until the wheel hits the wall—which is guaranteed to happen because the wheel is moving toward the wall at a constant rate. Once the collision occurs, we calculate the angles φ^+ and ψ^+. Additionally, because the contact point is a fixed point on the wheel while pivoting, $\dot{\varphi} = \dot{\psi}$, so, by the fundamental theorem of calculus, $\varphi^* - \varphi^+ = \psi^* - \psi^+$. Therefore, the set point is $\varphi^* = \pi \,/\, 2 - \psi^+ + \varphi^+$.

Assumption 1. For initial state $\mathbf{x}_0$, let φ^- be the unique angle of the wheel such that wheel would contact the wall when rolling from $\mathbf{x}_0$. We assume that $\mathbf{x}_0$ is chosen such that $s_0 > 0$ and $\varphi_0 < \varphi^- < \varphi^*$

Simply stated, this assumption requires that the wheel is not initially in contact with the wall and that it is not immediately at the set point the moment it hits the wall. Thus, if we let t^+ be the jump time when the wheel hits the wall, then there will be an interval of flow before t^+, and another interval of flow t^+ before the wheel reaches the set point.

Additionally, while the wheel is pivoting on the wall, there are minimum and maximum angles, $\varphi_{\min}$ and $\varphi_{\max}$, respectively, that the wheel can rotate without colliding with the ground on either side of the wall. The values of $\varphi_{\min}$ and $\varphi_{\max}$ can be calculated via method similar to the one used to calculate φ^*. In fact, the value of $\varphi_{\min}$ is the angle at which the wheel hits the wall, φ^+. In practice, however, we find that the angle constraints can often be omitted from the optimal control problem without changing the solution.

Next, we design a cost function that represents total energy expenditure. Recall that the power applied to the wheel is $P = \omega \cdot u$. Thus, for the cost function we choose the total work used to control the wheel,

$$\int_0^T |P(t)|\, dt = \int_0^T |\omega(t)u(t)|\, dt. \tag{3.1}$$

The full problem is then written as

Problem 3.1 (Optimal Control)**.**

$$\begin{aligned}
\underset{u}{\text{minimize}} \quad & \int_0^T |\omega(t)u(t)|\, dt \\
\text{subject to} \quad & \mathbf{x}(0) = \mathbf{x}_0 \\
& \dot{\mathbf{x}} = f(\mathbf{x}, u), \quad \forall \mathbf{x} \in C \\
& \mathbf{x}^+ = g(\mathbf{x}), \quad \forall \mathbf{x} \in D \\
& \mathbf{y}(\mathbf{x}) = (\varphi - \varphi^*, \omega) \\
& \|\mathbf{y}(\mathbf{x}(T))\| = 0 \\
& \varphi_{\min} \leq \varphi(t) \leq \varphi_{\max} \quad \forall t \in \left[t^+, T\right]
\end{aligned}$$

3.2 Construction of Stabilizing Controller

We see in Equation (2.14) that the continuous dynamics of $\mathcal{H}_\gamma$ are nonlinear, complicating the design of a controller. Therefore, before searching for an optimal control, we use input-output linearization, as described in [6], to analytically construct a non-optimal controller that will asymptotically stabilize the wheel to the set point. The resulting controller gives a baseline for comparison of the cost-improvement of

the optimal solution and allows for long-term stabilization of the wheel.

3.2.1 Input-Output Linearization

Recall that $\dot{\varphi} = \omega$ and the dynamics for ω, given in Equation (2.11), are

$$\dot{\omega} = \frac{u}{mr^2} - \frac{g}{r^2}\left(r'\sin(\psi) + r\psi'\cos(\psi)\right) - \frac{\dot{r}\omega}{r}.$$

We see that the dynamics are linear for φ but not for ω. Therefore, we want to linearize $\dot{\omega}$ such that

$$\dot{\omega} = \nu(\mathbf{y}), \tag{3.2}$$

where $\nu : \mathbb{R}^2 \to \mathbb{R}$ is a function we can choose. Then, ν will act as feedback control law for the output dynamics. To this end, we combine Equation (2.11) and Equation (3.2) and choose $u = u_\ell$ such that

$$\nu = \frac{u_\ell}{mr^2} - \frac{g}{r^2}\left(r'\sin(\psi) + r\psi'\cos(\psi)\right) - \frac{\dot{r}\omega}{r}.$$

Solving for u_ℓ produces

$$u_\ell(\nu, \mathbf{x}) = mr^2\nu + gm(r'\sin\psi + r\psi'\cos\psi) + mr\dot{r}\omega. \tag{3.3}$$

Remark. If we expand the composition of functions in Equation (3.3) we find the notation is quite cumbersome: $u_\ell\left(\nu(\mathbf{y}(\mathbf{x})), \mathbf{x}\right)$. Therefore, to simplify references to the composite control law, we write, by abuse of notation, that $(u_\ell \circ \nu)(\mathbf{x}) :=$

$u_\ell \left(\nu(\mathbf{y}(\mathbf{x})), \mathbf{x}\right)$.

We then have that the output dynamics of $\mathcal{H}_\gamma(u_\ell \circ \nu)$ during intervals of flow are given by

$$\mathcal{S}(\nu) := \begin{cases} \dot{\varphi} = \omega \\ \dot{\omega} = \nu(\mathbf{y}) \end{cases} = \begin{cases} \dot{y}_1 = y_2 \\ \dot{y}_2 = \nu(\mathbf{y}). \end{cases} \tag{3.4}$$

Now, because $\mathcal{S}(\nu)$ is a linear system, $\mathbf{y}$ can be controlled using proportional feedback. In particular, let $\kappa \in \mathbb{R}_{>0}$ and let $\nu_\kappa(\mathbf{y})$ be

$$\nu_\kappa(\mathbf{y}) := -\kappa^2 y_1 - 2\kappa y_2. \tag{3.5}$$

with $\kappa > 0$. We write $\mathcal{S}(\nu_\kappa)$ as a linear system,

$$\dot{\mathbf{y}} = \begin{bmatrix} 0 & 1 \\ -\kappa^2 & -2\kappa \end{bmatrix} \mathbf{y}. \tag{3.6}$$

We find that (3.6) has one eigenvalue $\lambda = -\kappa$ with multiplicity 2. Therefore, for $\kappa > 0$, the origin $\mathbf{y} = \mathbf{0}$ is globally asymptotically stable with respect to $\mathcal{S}(\nu_\kappa)$.

Note that the preceding result only describes the behavior of the output during intervals of flow, but since we are interested in the behavior of the full system, including jumps, we must consider $\mathcal{H}_\gamma$ with control law $u_\ell \circ \nu_\kappa$. To this end, we introduce two lemmas regarding the discrete jumps of $\mathcal{H}_\gamma$, which lead to a theorem that concludes that origin is pre-asymptotically stable with respect to $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$.

Lemma 3.1. *For any control u, the set $S = \{\mathbf{x} \in \mathcal{X} \mid k \neq 0\}$ is forward invariant*

with regards to $\mathcal{H}_\gamma(u)$.

Proof. For every $\mathbf{x} \in D$, we have that $k = 0$. Thus, S and D are disjoint. Additionally, $\dot{k} = 0$ for all $\mathbf{x} \in C$. Thus, if $k = 1$, jumps are impossible and k is constant. Therefore if $\mathbf{x}$ is a solution to $\mathcal{H}_\gamma$ such that $\mathbf{x}(0,0) \in S$, then $\mathbf{x}(t,j) \in S$ for all $(t,j) \in \operatorname{dom} \mathbf{x}$. That is, S is forward invariant with regards to $\mathcal{H}_\gamma(u)$. □

Lemma 3.2. *Every solution to $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ with initial condition $\mathbf{x}_0$ satisfying Assumption 1 has at exactly one jump.*

Proof. Let $\mathbf{x}$ be any solution with initial condition $\mathbf{x}_0$ that satisfies Assumption 1. First, we will prove that $\mathbf{x}$ cannot have multiple jumps. The jump map g dictates that after a jump, the mode is $k^+ \in \{-1, 1\}$. Thus $g(D) \subset S = \{\mathbf{x} \in \mathcal{X} \mid k \neq 0\}$. The set $g(D) \subset S$ is disjoint from D and, by Lemma 3.1, S is forward invariant, therefore $\mathbf{x}$ can have at most one jump.

Now, assume $\mathbf{x}$ has no jumps. In this case, the system behaves as a purely continuous system, so the output dynamics of $\mathbf{y}$ in $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ are given by $\mathcal{S}(\nu_\kappa)$. Furthermore, by Assumption 1, $\varphi_0 < \varphi^- < \varphi^*$. But φ is continuous and $\varphi \to \varphi^*$, so there is a time when $\varphi = \varphi^-$, hence a jump occurs. Thus, by contradiction, each solution to $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ has at least one jump and therefore has exactly one jump. □

Theorem 3.1. *For $\kappa > 0$, $\mathbf{y} = 0$ is pre-asymptotically stable with regards to $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ for all $\mathbf{x}_0$ satisfying Assumption 1.*

Proof. By Lemma 3.2, every solution $\mathbf{x}$ with $\mathbf{x}_0$ satisfying Assumption 1 has exactly one jump at some time t^+. There are two cases after the jump, $\mathbf{x}$ either terminates immediately or behaves only according to its continuous dynamics for all $t > t^+$. The continuous output dynamics of $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ are given by $\mathcal{S}(\nu_\kappa)$ and $\mathbf{y} = 0$ is asymptotically stable with respect to $\mathcal{S}(\nu_\kappa)$. Thus, after an impact, every solution to $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ either terminates after finite time (which means the solution is not complete) or it converges asymptotically to $\mathbf{y} = 0$. Therefore the origin is pre-asymptotically stable for all $\mathbf{x}_0$ satisfying Assumption 1. □

Remark. In general, not all maximal solutions of $\mathcal{H}_\gamma$ are complete because when $\langle \mathbf{r}^-, \mathbf{r}^+ \rangle < 0$ then $k^+ = -1$, causing solutions to jump out of $C \cup D$ and terminate. In fact, if the height of the wall is taller than the wheel, then *no* solutions are complete. Thus, we can only say that $\mathbf{y} = 0$ is pre-asymptotically stable, but not asymptotically stable.

3.2.2 Example: Elliptical Wheel

We apply the linearized controller to the elliptical wheel developed in Section 2.3. The parameters we use here are $a = 1$, $b = 4$, $m = 1, g = 9.8$, and $h_0 = 2$, with initial conditions $\varphi_0 = 0$, $\omega_0 = -1$ and $s_0 = -10$. The trajectories of φ, ω and P are shown in. Figure 3.2. We see that $\varphi \to \varphi^*$, $\omega \to 0$, as expected. We note that $P \to 0$, as well.

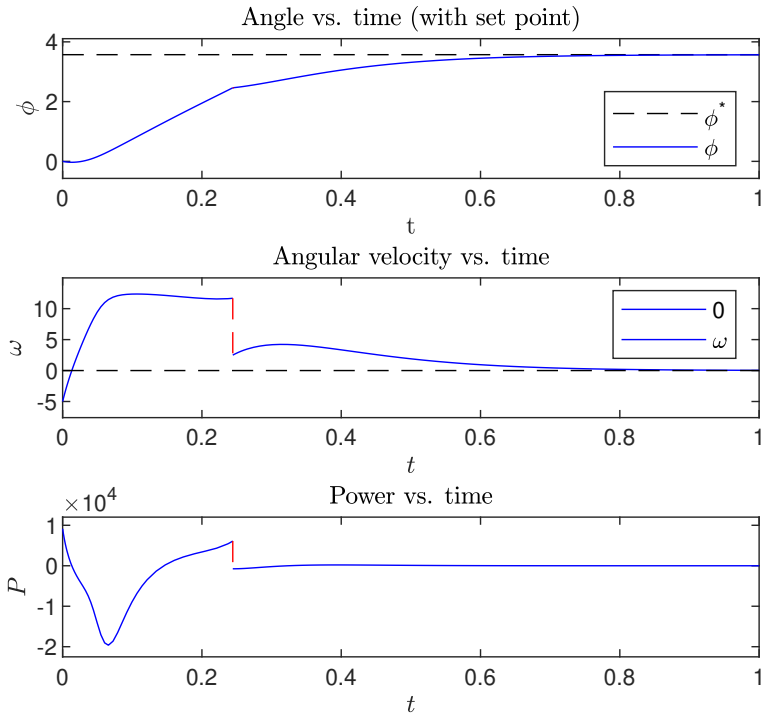


Figure 3.2: Trajectories of the system $\mathcal{H}_\gamma(u_\ell \circ \nu_\kappa)$ with $\kappa > 0$. We see the convergence of $\varphi \to \varphi^*$ and $\omega \to 0$, as well as $P \to 0$.

We note, however, that the maximum value of P is large—on the order 10^4—and P oscillates prior to the impact, which indicates that a portion of the control effort is being wasted by forcing $\mathbf{y}$ to behave according to linearized dynamics. Hence, this motivates the search for an optimal control.

3.3 Numerical Computation of Optimal Control

In order to compute a solution to Problem 3.1, we convert the optimal control problem into a discrete optimization problem, then solve.

3.3.1 Discretization of Optimal Control Problem

We discretize time into two intervals of flow, $[0, t^+]$ and $[t^+, T]$, with jump time $t^+ \in (0, T)$. Each interval has M discretization nodes evenly distributed with spacings of

$$\Delta t_1 := \frac{t^+}{M-1} \quad \text{and} \quad \Delta t_2 := \frac{T-t^+}{M-1}.$$

We write the times at each discretization node as $\mathbf{t}_1 := (t_{11}, t_{21}, ..., t_{M1})$ and $\mathbf{t}_2 = (t_{12}, t_{22}, ..., t_{M2})$ and the state value at time t_{ij} as $\mathbf{x}_{ij}$, so we want $\mathbf{x}_{ij} \approx \mathbf{x}(t_{ij})$ for $i = 1, 2, ..., M$ and $j = 1, 2$. Then, the state vectors at each time step are written as columns of matrices $X_1 := [\mathbf{x}_{11}\, \mathbf{x}_{21}\, \cdots\, \mathbf{x}_{M1}]$, and $X_2 := [\,\mathbf{x}_{12}\, \mathbf{x}_{22}\, \cdots\, \mathbf{x}_{M2}]$.

Similarly, we discretize the control signal as $\mathbf{u}_1 := (u_{11}, u_{21}, \cdots, u_{(M-1)1})$ and $\mathbf{u}_2 := (u_{12}, u_{22}, \cdots, u_{(M-1)2})$. We consolidate $X_1, X_2, \mathbf{u}_1, \mathbf{u}_2$ into two variables $X := [X_1\ X_2] \in \mathbb{R}^{n \times 2M}$ and $\mathbf{u} = \left[\begin{smallmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{smallmatrix}\right] \in \mathbb{R}^{2(M-1)}$. All together, the decision variables are $(t^+, X, \mathbf{u})$.

We approximate the cost function (3.1) using a Riemann sum,

$$\int_0^T |P(t)|\,dt = \int_0^T |\omega(t)u(t)|\,dt \approx \sum_{j=1}^{2} \Delta t_j \sum_{i=1}^{M-1} |\omega_{ij}u_{ij}|\,. \tag{3.7}$$

Now, we construct the constraints of our optimization such that solutions approximate Problem 3.1. The initial state is given as a constraint $\mathbf{x}_{11} = \mathbf{x}_0$. To compute numerical solutions to $\dot{\mathbf{x}} = f(\mathbf{x}, u)$, we use an explicit one-step numerical method in the form $\mathbf{x}_{n+1} = L(f, \mathbf{x}_n, u_n, \Delta t)$, where L is an operator determined by the choice of method. A simple example is the *forward Euler* method: $L_E(f, \mathbf{x}_i, u_i, \Delta t) := \mathbf{x}_i + \Delta t \cdot f(x_i, u_i)$. We then include the constraints $\mathbf{x}_{(i+1)j} = L(f_j, \mathbf{x}_{ij}, u_{ij}, \Delta t_j)$ for $i = 1, 2, ..., M-1$ and $j = 1, 2$.

Next, in order to implement collision detection in discrete time with fixed time steps, we must change the method for how impacts are modeled. Recall that in Chapter 2, jumps were modeled by defining a set D such that if $\mathbf{x} \in D$, then an impact has occurred. Roughly speaking, this is implemented within `HyEQsolver` by propagating trajectories until $\mathbf{x} \in D$. The solver then backtracks to pinpoint the time that $\mathbf{x}$ entered D. In our optimization scheme, such a process is infeasible and, thankfully, unnecessary. We replace the jump set D with an *impact detection function* $d : \mathcal{X} \to \mathbb{R}$ which satisfies the following criteria, as shown in Figure 3.3,

$$\begin{cases} d(\mathbf{x}) < 0, & \text{if the wall is inside the wheel} \\ d(\mathbf{x}) = 0, & \text{if the wall is on the boundary of the wheel} \\ d(\mathbf{x}) > 0, & \text{if the wall is outside the wheel.} \end{cases} \tag{3.8}$$

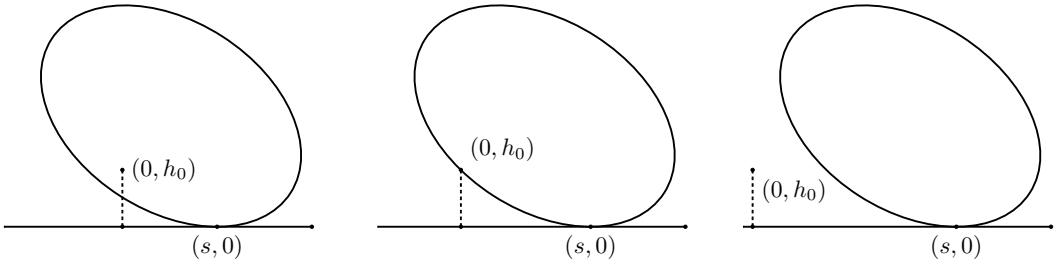


Figure 3.3: The three cases in the definition of the impact detection function d.

We then impose a constraint $d(\mathbf{x}_{M1}) = 0$, forcing the wheel to impact the wall at the end of the first interval of flow. We also include the the jump map as a constraint $\mathbf{x}_{12} = g(\mathbf{x}_{M1})$.

Returning to the example in Section 2.3, we construct an impact detection function d for the elliptical wheel. For an unrotated ellipse with semi-major a and semi-minor axes b, the foci of the ellipse are at $\pm(a^2 - b^2, 0)$. Then, for an ellipse in an arbitrary configuration, let d_1, d_2 be the Euclidean distances from each foci of the ellipse to $(0, h_0)$. It is a well-known property of ellipses that $(0, h_0)$ is on the ellipse if and only if $d_1 + d_2 = 2a$. For a wheel in state $\mathbf{x}$ with angle φ, and center (c_1, c_2), the vectors from $(0, h_0)$ to each foci are given by

$$\mathbf{d}_{1,2}(\mathbf{x}) = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \pm \begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} a^2 - b^2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ h_0 \end{bmatrix}.$$

Therefore, we choose $d(\mathbf{x}) = \|\mathbf{d}_1(\mathbf{x})\| + \|\mathbf{d}_2(\mathbf{x})\| - 2a$. The full optimization problem is then given as follows.

Problem 3.2 (Optimization)**.**

$$
\begin{aligned}
\underset{t^+, X, \mathbf{u}}{\text{minimize}} \quad & \sum_{j=1}^{2} \Delta t_j \sum_{i=1}^{M-1} |\omega_{ij} u_{ij}| \\
\text{subject to} \quad & 0 \leq t^+ \leq T \\
& \mathbf{x}_{11} = \mathbf{x}_0 \\
& \mathbf{x}_{(i+1)1} = L(f_0, \mathbf{x}_{i1}, u_{i1}, \Delta t_1) \quad \text{for } i = 1, 2, ..., M-1 \\
& d(\mathbf{x}_{M1}) = 0 \\
& \mathbf{x}_{12} = g(\mathbf{x}_{M1}) \\
& \mathbf{x}_{(i+1)2} = L(f_1, \mathbf{x}_{i2}, u_{i2}, \Delta t_2) \quad \text{for } i = 1, 2, ..., M-1 \\
& \|\mathbf{y}(\mathbf{x}_{M2})\| = 0 \\
& \varphi_{\min} \leq \varphi_{i2} \leq \varphi_{\max}, \quad \text{for } i = 1, 2, ..., M.
\end{aligned}
$$

We write a solution to Problem 3.2 as $(t^{+*}, X^*, \mathbf{u}^*)$.

3.3.2 Computation of Discretized Solution

To compute a numerical solution to Problem 3.2, we use `fmincon`, Matlab's solver for constrained nonlinear optimization. We find, however, that calculating a solution is not as straightforward as merely plugging in the data to `fmincon` and immediately getting a good solution. Our problem has a high number of dimensions, so iterations can be computationally expensive. Additionally, the nonlinearity of the problem means that good convergence to an optimum is not ensured. As a result, we must be careful in our implementation of the numerical optimization. Therefore,

in this section, we will discuss two aspects of the implementation that significantly affects the quality of results, namely the method used to compute the Jacobians of the cost and constraint functions, and the choice of ODE operator L.

First, however, we construct a means to check that our results accurately represent the dynamics of $\mathcal{H}_\gamma$. To evaluate the accuracy of the dynamics of the solution found with `fmincon`, we compare the results to trajectories produced by `HyEQsolver`, which provides a good baseline because it uses a high-order adaptive numerical ODE solver to compute solutions. In order to run `HyEQsolver`, however, we need to evaluate the control at arbitrary times. Thus, we interpolate $\mathbf{u}^*$ using constant interpolation. The result is an open-loop hybrid control function $\tilde{u}^* : \mathbb{R} \times \{1, 2\} \to \mathbb{R}$. For the control signal, we use constant interpolation to interpolate the values of $\mathbf{u}_1$ and $\mathbf{u}_2$. That is, the value of the control u is constant, equal to u_{ij}, throughout each interval $[t_i, t_{i+1}]$, as shown in Figure 3.4. We then compute a solution to $\mathcal{H}_\gamma(\tilde{u}^*)$

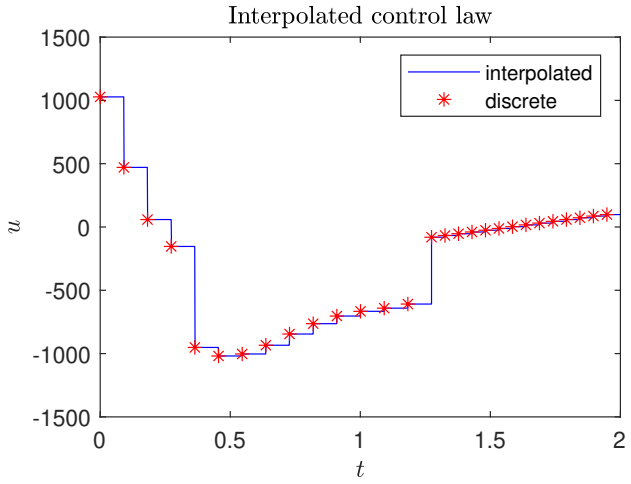


Figure 3.4: A control signal interpolated using constant interpolation in a discretization with $M = 15$ steps per interval.

with `HyEQsolver` and use spline interpolation to evaluate the solution at each discretization time, producing $\tilde{\mathbf{x}}_{ij}$ for $i = 1, 2, ..., M$ and $j = 1, 2$. If the `fmincon` solution is accurate, then $\tilde{\mathbf{x}}_{ij} \approx \mathbf{x}^*_{ij}$ for each i and j. Thus, the error at each step is given by $e_{ij} := \left\|\mathbf{x}^*_{ij} - \tilde{\mathbf{x}}_{ij}\right\|$ and the final error is $e_f := e_{M2}$.

To evaluate the optimality of solutions, we rely on the numerical optimality conditions used in `fmincon` (namely, the KKT conditions). For a more rigorous consideration of optimality, it would be necessary to apply Pontryagin's maximum principle adapted for hybrid control systems, as described in [2]. This is left for future work.

An important facet of our implementation is how to efficiently handle the large number of nonlinear constraints in our system. Numerical optimization methods rely on the Jacobian of the cost function and constraint functions to choose the values of the decision variables at the next step of the iteration. By default, `fmincon` uses finite differences to compute the Jacobians but this is computationally expensive—for example, if $M = 100$, then the constraints Jacobian contains $807,192$ entries, so a finite difference calculation requires over 1.6 million function evaluations each time the Jacobian is computed. For our constraints, however, the Jacobian is sparse and the number of nonzero entries grows linearly whereas the number of total entries grow quadratically. Therefore, in order to shorten the computation time, we calculate and provide the Jacobians to `fmincon`, making use of the sparsity patterns. In Table 3.1, we see that providing the Jacobian improves the number of iterations and

amount of real time required to compute solutions. Our Jacobian calculations are provided in Appendix B.

$M = 50$		
Jacobian	**Iterations**	**Time (hh:mm)**
Finite Differences	811	02:06
Analytical	131	00:01
$M = 100$		
Jacobian	**Iterations**	**Time (hh:mm)**
Finite Differences	166	01:56
Analytical	56	00:02

Table 3.1: Comparison of optimization with and without providing the Jacobian to `fmincon`.

Another factor that will affect the quality of solution is the choice ODE numerical method operator L. In particular, in Table 3.2 we compare the accuracy of the forward Euler method to the classic Runge-Kutta method (RK4) with $M = 20$ and with $M = 100$ time steps per interval. As expected, we find that RK4, which is fourth-order, vastly outperforms the first-order forward Euler method. Surprisingly, however the error of RK4 does not appear to improve greatly as M increases. The reason for this is that the `HyEQsolver` solution itself, which is used to calculate e_f, has numerical errors that are on the order of 10^{-5}.

	e_f **error**	
	$M = 20$	$M = 100$
Euler	3.8	9.0×10^{-1}
RK4	7.2×10^{-4}	2.4×10^{-5}

Table 3.2: Comparison of numerical ODE methods.

From these results, we conclude providing the Jacobian to the `fmincon` and us-

ing the RK4 method operator provide significant advantages when calculating the optimal control.

3.3.3 Results for Elliptical Wheel

As a concrete example, we again consider an elliptical wheel, with parameters $a = 4, b = 3, h_0 = 2.0$, and $T = 2.0$; initial conditions $\varphi_0 = -1.0, \omega_0 = 10.0, s_0 = 10.0$. In Figure 3.5, the state and control trajectories are shown for a solution with each interval of flow discretized into $M = 50$ time steps. The solver converged to a solution in 131 iterations after 116 seconds, the control cost was 295.2 and the final error was $e_f = 1.5 \times 10^{-5}$.

Examining the graph of power in Figure 3.5, we see that the choice of control makes intuitive sense. The wheel is initially rolling toward the wall so no power is necessary prior to the impact—if the controller were to accelerate the wheel toward the wall, then a portion of that energy would be lost in the collision. Once the wheel hits the wall, torque is applied, as needed, to lift the wheel onto the wall then gradually declines as the wheel nears the set point, where only a small amount of effort is required to stabilize it.

For comparison, we use the linearized feedback controller $u_\ell \circ \nu_\kappa$ with $\kappa = 3.0$, which achieves the set point reasonably well, with $\|\mathbf{y}(T)\| = 0.08$. We could reduce $\|\mathbf{y}(T)\|$ further by choosing a larger κ, but even for $\kappa = 3.0$, the control cost

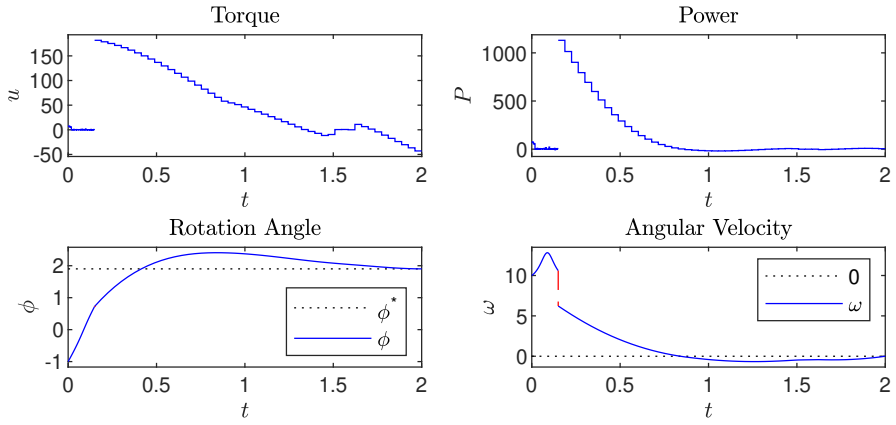


Figure 3.5: Control and state trajectories for an optimal solution with $a = 4, b = 3, \varphi_0 = -1.0, \omega_0 = 10.0, s_0 = 10.0, h_0 = 2.0$.

is 945.0, which exceeds the optimal control cost by more than a factor of three. A comparison of the power curves is shown in Figure 3.6.

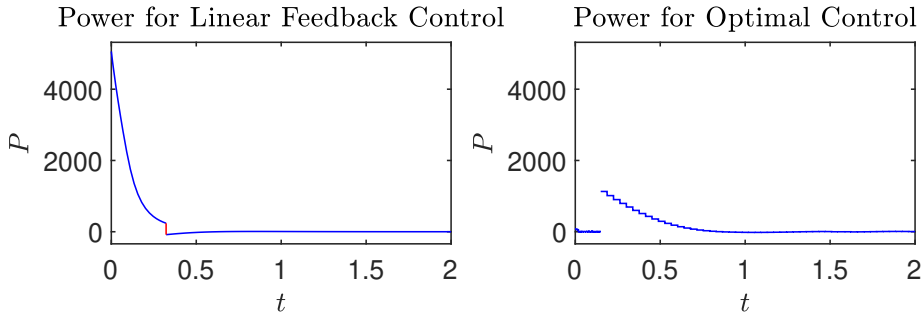


Figure 3.6: Comparison of the power used by feedback linearized control and the optimal control. The control cost for $u_\ell \circ \nu_\kappa$, with $\kappa = 3.0$, is 945.0 and the optimal control cost is 295.2.

Now, in light of the power savings, it might seem that the optimal control is always preferable to the linearized feedback control $\underline{\tau}_\varphi \circ \nu_\kappa$, but $\underline{\tau}_\varphi \circ \nu_\kappa$ has several benefits. First, the computation is much faster, requiring a fraction of a second to compute, compared to several minutes for the optimal solution. Additionally, $\underline{\tau}_\varphi \circ \nu_\kappa$ is a

feedback controller, so it is robust to perturbations of the state, whereas the optimal controller is open-loop, which means it will not adapt the control signal to account for deviations from the expected trajectories. In practice, if the optimal solution is used to control the wheel to the set point, we would switch to $u_\ell \circ \nu_\kappa$ once the set point is reached to provide long-term stability.

4

Conclusion

In Chapter 2, we modeled a noncirclular wheel using a hybrid dynamical system and presented an elliptical wheel as an example.

Next, in Chapter 3 we used input-output linearization to analytically construct an asymptotically stabilizing controller, which is useful as a baseline for comparing the optimal control as well as providing infinite-horizon stability, after the set point is reached. By discretizing the optimal control problem into an finite-dimensional optimization problem, we calculated an optimal control signal and the corresponding state trajectories such that solutions closely match trajectories produced by simply propagating the dynamics with `HyEQsolver`. We implemented the optimization problem in the case of the elliptical wheel and provided comparisons that showed that providing the Jacobian significantly improves performance and using the RK4

ODE solver method provides a large improvement to accuracy.

There are a number of ways that this work can be further developed. The model itself can made more physically accurate by including slipping, elasticity, or deformation. The setting of the problem can also be expanded by requiring that the wheel traverses a series of steps and walls, which would result in a more challenging optimal control problem due to a longer time horizon and increased number of intervals of flow. In fact, for our method to have broad applicability, it would need to be extended to allow for an arbitrary number of jumps, which are not known *a priori*. The process of calculating optimal solutions could also be improved. A particularly efficient class of direct methods are pseudospectral methods, which improve convergence by careful selection of the discretization nodes and using polynomial interpolation [4]. By using a pseudospectral method we could achieve faster convergence of the numerical results.

Appendix A

Properties of a Rotated Ellipse

Let $\mathcal{B}_0$ and $\mathcal{B}_\varphi$ be orthonormal bases of $\mathbb{R}^2$ such that $\mathcal{B}_\varphi$ is rotated by angle φ relative to $\mathcal{B}_0$. Consider the ellipse $\gamma(\tau)$ with coordinates $(a\cos(\tau), b\sin(\tau))$ relative to the $\mathcal{B}_\varphi$.

Lemma A.1. *The coordinates of γ in $\mathcal{B}_0$ are*

$$[\gamma(\tau)]_{\mathcal{B}_0} := \begin{bmatrix} x_\varphi(\tau) \\ \\ y_\varphi(\tau) \end{bmatrix} = \begin{bmatrix} C_x \cdot \cos\left(\tau + \hat{\varphi}_x\right) \\ \\ C_y \cdot \sin\left(\tau + \hat{\varphi}_y\right) \end{bmatrix}$$

where $C_x := \sqrt{a^2\cos^2\varphi + b^2\sin^2\varphi}$, $C_y := \sqrt{a^2\sin^2\varphi + b^2\cos^2\varphi}$,

$\hat{\varphi}_x := \texttt{atan2}\left(b\sin\varphi, a\cos\varphi\right)$, *and* $\hat{\varphi}_y := \texttt{atan2}\left(a\sin\varphi, b\cos\varphi\right)$.

Proof. The change of basis matrix from $\mathcal{B}_\varphi$ to $\mathcal{B}_0$ is the rotation matrix $\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$.

We then compute

$$[\gamma(\tau)]_{\mathcal{B}_0} = \begin{bmatrix} x_\varphi(\tau) \\ \\ y_\varphi(\tau) \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} a\cos(\tau) \\ \\ b\sin(\tau) \end{bmatrix} = \begin{bmatrix} a\cos\varphi\cos\tau - b\sin\varphi\sin\tau \\ \\ a\sin\varphi\cos\tau + b\cos\varphi\sin\tau \end{bmatrix}.$$

To simplify $x_\varphi(\tau)$ and $y_\varphi(\tau)$, we define $C_x = \sqrt{a^2\cos^2\varphi + b^2\sin^2\varphi}$ and $\hat{\varphi}_x := \texttt{atan2}(b\sin\varphi, a\cos\varphi)$. It follows from identities that $\sin\hat{\varphi}_x = \frac{b}{a}\tan\varphi\cos\hat{\varphi}_x$. We then find

$$\begin{aligned} C_x\sin\hat{\varphi}_x &= \sqrt{a^2\cos^2\varphi\sin^2\hat{\varphi}_x + b^2\sin^2\varphi\sin^2\hat{\varphi}_x} \\ &= \sqrt{a^2\cos^2\varphi\left(\frac{b}{a}\tan\varphi\cos\hat{\varphi}_x\right)^2 + b^2\sin^2\varphi\sin^2\hat{\varphi}_x} \\ &= \sqrt{b^2\sin^2\varphi\cos^2\hat{\varphi}_x + b^2\sin^2\varphi\sin^2\hat{\varphi}_x} \\ &= b\sin\varphi \end{aligned}$$

Similarly, $C_x\cos\hat{\varphi}_x = a\cos\varphi$. Then, substitution of $C_x\sin\hat{\varphi}_x$ and $C_x\cos\hat{\varphi}_x$ into $x_\varphi(\tau)$ produces $x_\varphi(\tau) = C_x(\cos\hat{\varphi}\cos\tau - \sin\hat{\varphi}\sin\tau)$. Then, by an additive cosine identity $x_\varphi(\tau) = C_x\cos(\tau + \hat{\varphi})$.

By the same process, if we choose

$$C_y = \sqrt{a^2\sin^2\varphi + b^2\cos^2\varphi} \quad \text{and} \quad \hat{\varphi}_y = \texttt{atan2}\left(a\sin\varphi, b\cos\varphi\right)$$

then $C_y\sin\hat{\varphi}_y = a\sin\varphi$ and $C_y\cos\hat{\varphi}_y = b\cos\varphi$, thus $y_\varphi(\tau) = C_y\sin(\tau + \hat{\varphi}_y)$

Therefore

$$[\gamma(\tau)]_{\mathcal{B}_0} = \begin{bmatrix} C_x \cdot \cos(\tau + \hat{\varphi}_x) \\ C_y \cdot \sin(\tau + \hat{\varphi}_y) \end{bmatrix}.$$

□

Corollary A.1. *The values of τ that give the extreme values of $x_\varphi(\tau)$, and $y_\varphi(\tau)$ are $\tau_{x,min} = \pi - \hat{\varphi}_x$, $\tau_{x,max} = -\hat{\varphi}_x$, $\tau_{y,min} = \frac{3\pi}{2} - \hat{\varphi}_y$, and $\tau_{y,max} = \frac{\pi}{2} - \hat{\varphi}_y$, respectively.*

Proof. It follows directly from Lemma A.1 that the values of the parameter τ at extrema are as follows

$$\tau_{x,\max} = \underset{\tau}{\operatorname{argmax}} \ \cos(\tau + \hat{\varphi}_x) = -\hat{\varphi}_x \tag{A.1}$$

$$\tau_{x,\min} = \underset{\tau}{\operatorname{argmin}} \ \cos(\tau + \hat{\varphi}_x) = \pi - \hat{\varphi}_x \tag{A.2}$$

$$\tau_{y,\min} = \underset{\tau}{\operatorname{argmin}} \ \sin(\tau + \hat{\varphi}_y) = \frac{3\pi}{2} - \hat{\varphi}_y \tag{A.3}$$

$$\tau_{y,\max} = \underset{\tau}{\operatorname{argmax}} \ \sin(\tau + \hat{\varphi}_y) = \frac{\pi}{2} - \hat{\varphi}_y \tag{A.4}$$

□

Lemma A.2. *For the rotated ellipse centered at the origin, the lower branch of y_φ as a function of x is*

$$y_\varphi(x) = C_y \sin\left(-\cos^{-1}\left(\frac{x}{C_x}\right) - \hat{\varphi}_x + \hat{\varphi}_y\right)$$

Proof. From Lemma A.1, it follows that $\operatorname{range}(x_\varphi) = [-C_x, C_x]$, and $\operatorname{range}(y_\varphi) =$

$[-C_y, C_y]$. Thus, for a given x, we can solve for τ.

$$\tau(x) = -\cos^{-1}\left(\frac{x}{C_x}\right) - \hat{\varphi}_x. \tag{A.5}$$

From Lemma A.1, we have $y_\varphi(x) = C_y \sin\left(\tau(x) + \hat{\varphi}_y\right)$, therefore,

$$y_\varphi(x) = C_y \sin\left(-\cos^{-1}\left(\frac{x}{C_x}\right) - \hat{\varphi}_x + \hat{\varphi}_y\right). \tag{A.6}$$

□

Appendix B

Jacobian Calculations

In order to discuss the Jacobian of the cost and constraints, we can write our optimization variable $t^+, X, \mathbf{u}$ as single vector

$$\mathbf{z} = \begin{bmatrix} \mathbf{x}_{11}^\top & \cdots & \mathbf{x}_{M1}^\top & \mathbf{x}_{12}^\top & \cdots & \mathbf{x}_{M2}^\top & \mathbf{u}_1^\top & \mathbf{u}_2^\top & t^+ \end{bmatrix}^\top.$$

The cost function for Problem 3.2 is

$$I(\mathbf{z}) := \sum_{j=1}^{2} \Delta t_j \sum_{i=1}^{M-1} \left| \omega_{ij} u_{ij} \right|.$$

Computing the Jacobian is complicated by the fact that the derivative of the absolute value function is undefined at the origin. We mitigate this issue, however, by using

the following generalization of the derivative,

$$\frac{d\,|x|}{dx} = \operatorname{sgn}(x) := \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1 & x > 0. \end{cases}.$$

This works well for our purposes because the only nonsmooth point in the absolute value function is also the minimal point and is where $\frac{\partial |x|}{\partial x} = 0$, allowing `fmincon` to correctly choose its search directions. We then find

$$\begin{aligned} \frac{\partial I}{\partial t^+} &= \frac{1}{M-1}\sum_{i=1}^{M-1} |\omega_{i1} u_{i1}| - \frac{1}{M-1}\sum_{i=1}^{M-1} |\omega_{i2} u_{i2}| \\ \frac{\partial I}{\partial \omega_{ij}} &= \Delta t_j \operatorname{sgn}(\omega_{ij} u_{ij}) u_{ij} \\ \frac{\partial I}{\partial u_{ij}} &= \Delta t_j \operatorname{sgn}(\omega_{ij} u_{ij}) \omega_{ij} \end{aligned}$$

Then the Jacobian is

$$\nabla_{\mathbf{z}} I(\mathbf{z}) = \begin{bmatrix} \nabla_{\mathbf{x}_{11}} I(\mathbf{z}) \\ \vdots \\ \nabla_{\mathbf{x}_{M1}} I(\mathbf{z}) \\ \nabla_{\mathbf{x}_{12}} I(\mathbf{z}) \\ \vdots \\ \nabla_{\mathbf{x}_{M2}} I(\mathbf{z}) \\ \nabla_{\mathbf{u}_1} I(\mathbf{z}) \\ \nabla_{\mathbf{u}_2} I(\mathbf{z}) \\ \frac{\partial I}{\partial t^+} \end{bmatrix}$$

Next, we write the nonlinear constraints as $G(\mathbf{z}) = \mathbf{0}$ where

$$G(\mathbf{z}) = \begin{bmatrix} \mathbf{x}_{11} - \mathbf{x}_0 \\ \mathbf{x}_{21} - L(f_0, \mathbf{x}_{11}, u_{11}, \Delta t_1) \\ \vdots \\ \mathbf{x}_{M1} - L(f_0, \mathbf{x}_{(M-1)1}, u_{(M-1)1}, \Delta t_1) \\ \mathbf{x}_{22} - L(f_1, \mathbf{x}_{12}, u_{12}, \Delta t_2) \\ \vdots \\ \mathbf{x}_{M2} - L(f_1, \mathbf{x}_{(M-1)2}, u_{(M-1)2}, \Delta t_2) \\ d(\mathbf{x}_{M1}) \\ \|\mathbf{y}(\mathbf{x}_{M2})\| \end{bmatrix}$$

Let $F_{i1} = -\nabla_{\mathbf{x}} L(f_0, \mathbf{x}_{i1}, u_{i1}, \Delta t_1)$. If the Euler method $L = L_E$ is used, then $F_{i1} = -(I_4 + \Delta t_1 \nabla_{\mathbf{x}} f_1(\mathbf{x}_i, u_i))$. Similarly, $T_{i1} := -\nabla_{t^+} L(f_0, \mathbf{x}_{i1}, u_{i1}, \Delta t_1) = \dfrac{1}{M-1} f_0^{\top}(\mathbf{x}_{i1}, u_{i1})$. We then find

$$U_{i1} := -\nabla_{\mathbf{u}} L(f_0, \mathbf{x}_{i1}, u_{i1}, \Delta t_1) = \frac{1}{m r_{i1}^2} \begin{bmatrix} \mathbf{0} & \mathbf{e}_i & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

where $\mathbf{e}_i \in \mathbb{R}^{M-1}$ is the vector containing 0's in every entry except a 1 in the i-th entry. The definitions of F_{i2}, U_{i2} and T_{i2} are similar. Finite differences is used to calculate the values of $\nabla g(\mathbf{x}_{M1})$ and $\nabla_{\mathbf{x}} \|\mathbf{y}(\mathbf{x}_{M2})\|$. The Jacobian $\nabla_{\mathbf{z}} G$ is then shown in Figure B.1.

$$
\nabla G(\mathbf{z}) = \left[\begin{array}{cccc|cccc|cc|c}
t_{11} & t_{21} & \cdots & t_{M1} & t_{12} & t_{22} & \cdots & t_{M2} & \begin{array}{c}\text{impact}\\ \text{distance}\end{array} & \begin{array}{c}\text{set-point}\\ \text{distance}\end{array} & \\
I_4 & F_{11} & \cdots & O & O & O & \cdots & O & \mathbf{0} & \mathbf{0} & \mathbf{x}_{11} \\
O & I_4 & \ddots & O & O & O & \cdots & O & \mathbf{0} & \mathbf{0} & \vdots \\
\vdots & \vdots & \ddots & F_{(M-1)1} & O & O & \cdots & O & \mathbf{0} & \mathbf{0} & \mathbf{x}_{(M-1)1} \\
O & O & \ddots & I_4 & \nabla g(\mathbf{x}_{M1}) & O & \cdots & O & \nabla_{\mathbf{x}} d(\mathbf{x}_{M1}) & \mathbf{0} & \mathbf{x}_{M1} \\
O & O & \ddots & O & I_4 & F_{12} & \cdots & O & \mathbf{0} & \mathbf{0} & \mathbf{x}_{12} \\
O & O & \ddots & O & O & I_4 & \ddots & \vdots & \vdots & \vdots & \vdots \\
O & O & \ddots & \ddots & \ddots & \ddots & \ddots & F_{(M-1)2} & \mathbf{0} & \mathbf{0} & \mathbf{x}_{(M-1)2} \\
O & O & \ddots & O & O & O & \ddots & I_4 & \mathbf{0} & \nabla_{\mathbf{x}} \left\| \mathbf{y}(\mathbf{x}_{M2}) \right\| & \mathbf{x}_{M2} \\
O & U_{11} & \cdots & U_{(M-1)1} & O & U_{12} & \cdots & U_{(M-1)2} & \mathbf{0} & \mathbf{0} & \mathbf{u} \\
O & T_{11} & \cdots & T_{(M-1)1} & O & T_{12} & \cdots & T_{(M-1)2} & \mathbf{0} & \mathbf{0} & t^+
\end{array}\right]
$$

Figure B.1: The Jacobian pattern for the nonlinear constraint function $G(\mathbf{z})$.

Bibliography

[1] John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Advances in Design and Control. Society for Industrial and Applied Mathematics, Jan. 1, 2010. 442 pp.

[2] A.V. Dmitruk and A.M. Kaganovich. "The Hybrid Maximum Principle is a consequence of Pontryagin Maximum Principle". In: *Systems & Control Letters* 57.11 (Nov. 2008), pp. 964–970.

[3] Rafal Goebel, Ricardo G. Sanfelice, and Andrew R. Teel. *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. Princeton University Press, 2012. 232 pp.

[4] Qi Gong, Wei Kang, and I.M. Ross. "A pseudospectral method for the optimal control of constrained feedback linearizable systems". In: *IEEE Transactions on Automatic Control* 51.7 (July 2006). Conference Name: IEEE Transactions on Automatic Control, pp. 1115–1129.

[5] Maryam Kamgarpour et al. "Hybrid Optimal Control for Aircraft Trajectory Design with a Variable Sequence of Modes". In: *IFAC Proceedings Volumes*. 18th IFAC World Congress 44.1 (Jan. 1, 2011), pp. 7238–7243.

[6] Hassan K. Khalil. *Nonlinear Systems*. Third Edition. Pearson, 2014.

[7] Ricardo Sanfelice. *Hybrid Equations Toolbox*. Version 2.04. MATLAB Central File Exchange, 2020.