Dynamics and Control for Efficient Hopping

An analysis of cross-domain dynamics of simple natural dynamic hoppers, and the implications for energy-optimal control.

Autumn Term 2013
Acknowledgements

I would like to thank my supervisor Fabian Günther, whose never-ending questions, prodding and advice kept this thesis -and my motivation- on track. His approach has always been systematic, clear and seasoned with a touch of humor, something that is not so easy to find and much to be appreciated.
I would also like to thank professor Fumiya Iida for the opportunity and trust in letting me define the goal of the project.
I would also like to thank Fabio Giardina, whose second-generation segmented-beam hopper provided the ideal platform to easily test out the hypothesis’ worked out in this thesis. Fabio also shared an office with me, and the frequent discussion provided great insight as well as the occasionally needed respite from the endless differential equations.
Many more people contributed to making this thesis both valuable and enjoyable, and I wish to thank them all.
Abstract

This thesis deals with the dynamics and control of energy-optimal gaits for simple natural-frequency hoppers. The conclusions extend our understanding of how to design efficient systems. We show an equivalent formulation of the energy-optimization problem which leads to a bang-bang control input. Further, we show that for locomoting at natural-frequency, finding the optimal-control can be done via parameter search. A parameter search with reduced parameter-space was done for the segmented-beam hopper CHIARO, yielding a Cost of Transport of 0.49 at forward velocity of \(0.25\text{ m/s}\). We also propose an extended model for curved-beam hoppers which includes the electro-mechanical dynamics of the motor-pendulum system. The resulting equations of motion shed light on how to design the hoppers. Further, they show that for curved-beam hoppers, designing the physical morphology is of greater importance than control.
## Contents

### Abstract

1 Introduction
1.1 Why Study Legged Robots? ........................................... 1
1.2 On Comparing Gaits and Efficiency of Legged Systems .......... 1
1.3 Natural Dynamics and Morphological Computation ............... 3
1.4 Motivation at BIRL .................................................. 4
1.5 Goal of this Thesis ................................................ 6

2 Mathematical Principles ........................................... 7
2.1 Optimal Control in a nutshell ..................................... 7
   2.1.1 Pontryagin's Minimum Principle ............................ 7
   2.1.2 Bang-Bang as an optimal controller ....................... 8
   2.1.3 Formulating equivalent cost functions .................... 8
2.2 Efforts, Flows and Cross-Domain Power Analysis ............... 9
   2.2.1 The Fundamental System Variables ....................... 10
   2.2.2 Unified System Components ............................... 11
   2.2.3 The DC-Motor as an Electro-Mechanical Example .......... 12
2.3 Positive Work, Negative Work and the Hopping at Natural Frequency 13

3 Efficient Control of Segmented Beam Hoppers ................... 15
3.1 The hopping robot CHIARO ....................................... 15
3.2 Separating the stability problem from the energy-optimality problem 17
3.3 Bang-Bang for CHIARO ........................................... 18
3.4 Picking a Equivalent Cost Function .............................. 18
   3.4.1 Minimizing Damping Losses for a 1-D Hopper ............ 19
   3.4.2 Extending the Cost Function ............................... 23
3.5 Heuristic Parameter Search in the Real World .................. 24
   3.5.1 How to Find the Switching Function via Parameter Search 24
   3.5.2 Reducing the Parameter Space ............................... 25
   3.5.3 Real-World Results with CHIARO ......................... 27

4 C-shaped Curved Beam Hoppers: electro-mechanical dynamics .... 29
4.1 Modeling the Electro-Mechanical Dynamics ...................... 29
   4.1.1 Previous Models of the C-Shaped Hopper .................. 30
   4.1.2 An Extended Model Including the Motor-Pendulum Dynamics 30
   4.1.3 Deriving the Equations of Motion .......................... 31
4.2 Analysis of Scaling Effects ..................................... 33
   4.2.1 Source Voltage as an Indirect Control Input ............ 33
   4.2.2 Scaling to Emphasize Centripetal Force .................. 34
   4.2.3 Some Remarks on Energy-Optimal Control for the Curved-Beam Hopper ................. 35
Chapter 1

Introduction

1.1 Why Study Legged Robots?

Humans have always been fascinated with motion and dynamics, whether for practical purposes such as transport, the amusement of watching complex artistic movements, or the pure exhilaration of high speeds. While this is often achieved with machines, the most immediate source of motion has of course always been our own bodies, or that of animals. While we easily master the control of our bodies intuitively, we’re still far from understanding the underlying mathematical principles that actually govern them. This makes studying motion, of any sort, rewarding in itself. Apart from this, for the specific field robotics and legged locomotion, there are three primary motivations.

1. As an alternative to wheels, legs can be much more versatile: they provide greater ground clearance, can navigate spaces and very rough terrain, and don’t need to trace a continuous path, but can choose as a foothold any point within the leg’s reach. Therefore understanding how to design and control legged robots promises a generation of very versatile and mobile platforms.

2. Legged robots allow an engineering approach to understanding how nature solves this problem. Biologists typically deal with inverse problems: the system is already designed and built, and their task is to understand how it works and why. Building bio-mimetic robots allows the testing of specific theories in a controlled manner.

3. Finally, from a bio-medical point of view, understanding more on how legged systems work would allow engineers to design better prosthetics, in particular active prosthetics. Also important today is the application of robotic rehabilitation, where mechatronic devices assist humans in learning tasks after, for example, suffering a stroke.

1.2 On Comparing Gaits and Efficiency of Legged Systems

Modern understanding of the difference between running and walking dates to the photographic studies of Muybridge in the 1870s, when he demonstrated through chronophotography the presence of a flight phase\(^1\) during running for horses as seen in figure 1.1 and, subsequently, various other animals.

\(^1\)Flight phase: a phase during a single cycle in which no body-parts touch the ground.
Chapter 1. Introduction

Figure 1.1: Muybridge’s famed photographs captured the various stages of horses running, positively proving a much debated question of the time: does running involve a flight phase?

Since then the interest of understanding the bio-mechanics of motion in an analytical manner has increased, with many biologists applying what are generally considered engineering tools to the task. In the 1980s Alexander conducted extensive studies on the relation of posture and scaling with the gaits animals used at varying speeds [1]. One of the major findings is that animals with similar postures but vastly different mass adopt the same gait at equal Froude numbers[25], with the Froude number defined for legged animals as

\[ Fr = \frac{v^2}{gl} \]  

(1.1)

where \( v \) is the forward velocity, \( g \) is the gravitational pull and \( l \) is the leg length\(^2\). However, unlike in engineering, in biology scientists generally attempt to solve inverse problems: In most engineering tasks, the cost function to be optimised generally well defined, and the challenge lies in finding a solution to this problem while fulfilling given constraints. Instead, biologists can generally assume that nature has already solved an optimality problem, and the readily observable animal behavior is the solution: the challenge lies in discerning what the optimised cost function actually is, and what factors led to it, i.e. what are the constraints? Because of this, it is important to exercise caution when comparing animals to engineered machines. The froude number doesn’t explicitly include efficiency, and the constraints, loss-sources and dynamics of feasible electro-mechanical designs still differ significantly from biological design. Because of this, the efficiency and energy-optimal controller could well be very different, with the artificial counterparts adopting completely different gaits at similar froude numbers.

In order to better compare efficiency, Tucker formulated the so-called Cost of Transportation[24], defined as

\[ CoT = \frac{P(t)}{mgv(t)} = \frac{E}{mgd} \]

where \( P(t) \) is power, \( m \) is the mass, \( g \) is gravitational pull and \( v(t) \) is velocity. The alternative formulation simply averages the cost over a specific distance by solving the time integral of power and velocity. When assuming a steady-state locomotion, this tends to be easier as precise real-time measurements are no longer necessary. \( CoT \)

\(^2\)Sometimes the stride length is taken for \( l \) instead.
has since become a commonly used benchmark for comparing efficiency in legged robots, with the ideal reference benchmark usually being the biological counterpart for a specific gait. This dimensionless number also has its drawbacks, namely it doesn’t reward speed, as the alternate formulation shows. Power usually doesn’t scale well with speed, so we see that the alternate formulation is less misleading. If we reconsider that animals adopt specific gaits based on the Froude number, each gait is an energy-optimal solution for that specific speed and morphology, and not the specific distance covered. So it is not surprising that a slower walking gait will have lower $\text{CoT}$ compared to a faster running gait, but are energy-optimal solutions at their respective speeds.

1.3 Natural Dynamics and Morphological Computation

Contemporaneously to Alexander, Tad McGeer also broke ground in the area of efficient legged locomotion, developing the principles of passive dynamics with his Passive Dynamic Walker (PDW)[15]. This mechanical contraption was built similarly to a pair of human legs, including a unilaterally knee-constraint, and could stably walk down gentle slopes. What was so impressive is that it did so with no sensing and no actuation and ergo no control. If the legs were held upright, they would topple as soon as let go, but if started on a gentle slope they would settle into a dynamically stable limit-cycle! The gentle slope causes a continuous conversion of potential to kinetic energy, necessary to make up for minor losses from joint-friction and impact. Nonetheless it proves the possibility of solving extremely efficient, dynamically stable locomotion through morphological design instead of control. Later, Ruina’s lab at Cornell designed the Cornell Ranger[3], a powered ‘passive’ dynamic walker\(^3\) which currently holds the world record for distance travelled on a single battery charge (currently at 65.2 km) and the lowest $\text{CoT}$, at 0.28, very close to the fabled 0.2 of human walking.

All these designs showed the advantage of locomoting at the natural frequency of the systems. Indeed, it has been shown that human walking can be modeled with this principle as an inverted pendulum during the stance phase, and a regular double-pendulum during swing phase in order to analyze stability and velocities[8]. Further studies, including this time ground reaction forces measured through force-plates, showed the importance of the spring-like nature of muscles and tendons in legs and led to the so-called spring-loaded inverted pendulum (SLIP) model[4]. From an energetic point of view, the springs seem an obvious choice to recycle the impact energy of each step, since legged-locomotion involves a lot of up and downs. Indeed, early models assumed that during walking control minimized vertical displacement of the body mass, under the assumption that this would minimize the necessary work to be done each step. The SLIP model side-stepped this by letting the springs do the work on the body-mass instead of the actuators.

Springs were found to be important not only for energetics, but also for stability. At the AI Lab of the University of Zurich, it was found that often in nature the physical morphology of certain animals is such that they simplify specific tasks that engineers would usually solve through computation. The term morphological computation was coined to describe this phenomenon. Fast and stable running for quadruped and bipeds is generally been considered a difficult control problem\(^4\),

\(^3\)Generally ‘passive’ means in complete absence of actuation, and the term ‘natural dynamics’ are used to indicate that actuation is done in harmony with the un-actuated dynamics of the system.

\(^4\)During most running gaits quadrupeds have less than three contact points on the ground at a time, meaning there is no static support polygon: they must rely on dynamic stability solutions.
but the Puppy robot[19] [10] demonstrated very robust locomotion with only feedforward actuation by greatly enlarging the stable basin of attraction through clever use of springs in the legs.

1.4 Motivation at BIRL

At the Bio-Inspired Robotics Lab of ETH Zurich, under professor Fumiya Iida, we are currently tackling how to exploit natural dynamics for efficient locomotion, starting with very simple systems in order to isolate and identify design principles. As the simplest form of a hopping system, it is useful to reduce the system to a single mass, upon which we can freely apply forces such that ballistic flight is achieved, as seen in figure 1.3. In this case, the path of the mass resembles that of a bouncing ball. In this case, the force applied to the mass must do two things. First, it must accelerate the mass from stand-still until it achieves ballistic flight. Second, it must ‘catch’ the mass again and decelerate it, in other words, it must do negative work on the system: the force applied removes energy from the system. If this force is applied by an actuator without energy-storage, the actuator expends energy once to transfer it to the system, and then once again to remove it from the system. This leads us to the first design principle: Minimize Unsprung Mass. By placing springs under the mass, the kinetic energy of the system at landing can be stored in the potential energy of the spring and released again to achieve the next step. Any kinetic energy carried by unsprung mass is instead lost on impact, and needs to be replaced again by the actuator.

While an ideal system with no unsprung mass would hop perpetually, this is of course not so for real systems: every joint, actuator and even spring has some inherent friction or damping through which energy is also dissipated. The second principle is to Minimize Friction. This can be done in many ways. Simplifying the system to have as few joints as possible elegantly avoids the problem. A proper choice of motors and gearing is also very important: most conventional gears display large amounts of friction and/or are display poor backdriveability: they may efficient when driven from the motor-side, but display extremely high friction when driven...
1.4. Motivation at BIRL

If hopping is treated as a mass moving along ballistic trajectories, the force applied must do all the work on the system: both to accelerate it until lift-off as well as slow it down at landing. (Figure 1.3)

The kinetic energy carried by the sprung mass will be absorbed into the spring then released again for the next step. Instead, the kinetic energy of the unsprung mass is lost at impact, and the input force must do an equal amount of positive work to replace it. Therefore, minimizing unsprung mass is important for achieving efficient locomotion. (Figure 1.4)
from the shaft. Using gearless or low-geared motors is one way of reducing this, however it should be noted that motor-torque output is proportional to current. In order to achieve the same torque without gears, motors must run with much higher current, leading to greater heat-losses in the motors. Therefore it is important to find a good balance in the design.

Treating the system now as a mass-spring system, it displays certain resonant frequencies at which power losses are minimal. This brings us to the final principle, already explored with the passive dynamic walkers: Minimize negative work. Negative work is defined as when an effort, i.e. force, is applied in opposite direction to it’s corresponding flow, i.e. velocity\(^5\), in other words it removes energy from a system and represents negative power. Exciting a system away from it’s eigen-frequency implies doing negative work, in other words, the actuator is acting both as an actuator as well as a brake in order to force the system to follow a motion that is not natural. This should be avoided either by adjusting the control input or adjusting the system dynamics, e.g. by tuning the spring stiffness.

1.5 Goal of this Thesis

So far, efficient locomotion through natural dynamics has been usually approached as a mechanical design challenge. We state the following hypothesis. Applying optimal control theory to the systems locomoting at natural dynamics, such as the C-shaped curved beam hopper and the segmented-beam hoppers, leads to lower Cost of Transport.

\(^5\)For more on efforts and flows, see section 2.2.
Chapter 2

Mathematical Principles

In this thesis, we deal with the energy-optimal control of simple hopping robots, the dynamics of which span both the mechanical and the electrical domain. Before proceeding, we will introduce the mathematical concepts and notation that will be frequently used throughout the rest of this thesis. In particular we will quickly expose the reader to optimal control theory, also a unified approach to modeling cross-domain systems and finally we will touch on stable-limit cycles and what this means for efficient hopping.

2.1 Optimal Control in a nutshell

Optimal control is an extension of the calculus of variations used to find control policies. It is a very powerful tool, as it can prove a certain control trajectory to be optimal, i.e. there exists no other trajectory that yields a better solution. It is important to keep in mind that claiming a control policy is ‘optimal’ has no meaning by itself: it must always be the optimal policy for a specific cost function. Of the many principles belonging to optimal control theory, we will make use of Pontryagin’s Minimum Principle (PMP) which is applied to continuous time systems. Without going into the proofs, the main points of the PMP are presented here. For more details, see [2].

2.1.1 Pontryagin’s Minimum Principle

Consider the standard continuous-time optimality problem, formulated as follows:

\[
\min_{\tilde{u}(t)} \int_0^T g(\tilde{x}(t),\tilde{u}(t)) \, dt
\]

subject to \( \dot{\tilde{x}}(t) = \tilde{f}(\tilde{x}(t),\tilde{u}(t)) \) (2.1)

where \( \tilde{x}(t) \) is the vector of state variables, \( g(\tilde{x}(t),\tilde{u}(t)) \) is the cost function and \( \tilde{f}(\tilde{x}(t),\tilde{u}(t)) \) are the system dynamics. The objective of the optimization problem is to find a trajectory of control inputs \( u(t) \) which minimizes the integral of the cost function \( g(\tilde{x}(t),\tilde{u}(t)) \) over a certain time, usually while fulfilling certain specific boundary conditions. In the 1950s, the Russian mathematician Pontryagin first derived the minimum principle to solve this optimization problem analytically. The optimisation problem is equivalent to solving the following problem:
\[
\tilde{u}^*_t = \arg\min_u \left[ H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t)) \right]
\] (2.2)
\[
H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t)) = g(\tilde{x}(t), \tilde{u}(t)) + \tilde{p}(t)^T \hat{f}(\tilde{x}(t), \tilde{u}(t))
\] (2.3)
\[
\tilde{p}(t) = -\nabla_x H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t))
\] (2.4)

where \( H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t)) \) is the so-called Hamiltonian function, a superscripted star such as in \( \tilde{u}^*_t \) indicates an optimal trajectory and \( \tilde{p}(t) \) is a vector of adjoint variables, with size equal to \( \tilde{x}(t) \). The adjoint variables indicate the sensitivity of the final cost to perturbations to the dynamics, in other words the cost of deviating from the optimal trajectory. In order to solve the equivalent problem, several boundary conditions for the differential equations \( \hat{p}(t) = -\nabla_x H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t)) \) can be used, depending on the exact formulation: whether initial or ending values of \( \tilde{x}(t) \) are known or free, if there is a terminal cost, etc. For more details on the types of boundary conditions and a complete derivation, see [2]. Equation (2.2) shows that the nature of the differential equations to be solved depends on the dynamics of the system. In our case, the dynamics are non-linear and thus so are the differential equations, making them very difficult to solve analytically or in most cases having no analytical solution.

### 2.1.2 Bang-Bang as an optimal controller

There is a special case for the solution to PMP. That is, if the Hamiltonian is linear in \( \tilde{u}(t) \), the optimal controller must be a bang-bang controller. A bang-bang controller is a controller that always takes the extreme values allowed, i.e. either it’s maximum or it’s minimum value. Recall that for a function \( F(x) \) to be linear, it should fulfill the following condition:

\[
F(x_1 + x_2) = F(x_1) + F(x_2)
\]

Let’s clarify this statement: when the Hamiltonian function is linear in \( \tilde{u}(t) \) it can be rewritten as:

\[
H(\tilde{x}(t), \tilde{u}(t), \tilde{p}(t)) = \sigma_1(\tilde{x}(t), \tilde{p}(t)) + \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \tilde{u}(t)
\] (2.5)

where \( \sigma_1(\tilde{x}(t), \tilde{p}(t)) \) gathers all terms that do not multiply \( \tilde{u}(t) \) and \( \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \) gathers all those that do. In order to minimise the Hamiltonian, we must make \( \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \tilde{u}(t) \) as negative as possible. Thus, for each element of \( \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \) that is positive, the corresponding element of \( \tilde{u}(t) \) should be as negative as possible, while for negative element of \( \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \), the corresponding elements of \( \tilde{u}(t) \) should be as positive as possible. The function \( \sigma_{\text{switch}}(\tilde{x}(t), \tilde{p}(t)) \) is called in these cases the switching function, as it’s sign determines whether the maximal or the minimal bang is input. This is an important result from control theory of which we will make use in the next chapters.

### 2.1.3 Formulating equivalent cost functions

Very often the most intuitive cost function to formulate turns out to be extremely difficult to solve. To side-step this, alternative equivalent cost functions are used instead. For example, take the system represented in figure 2.1:

with system dynamics:

\[
\dot{x}(t) = u(t)
\]
2.2 Efforts, Flows and Cross-Domain Power Analysis

Engineers tend to specialize in specific domains: electrical, mechanical, thermal or fluid just to name a few. In this section we will present a unified set of system variables, with which dynamics systems in different domains can be described. In particular, it will be shown how power flows in and out of the system depending on the relationships between these variables, and the equivalents between different domains. Because in this thesis we focus on electro-mechanical systems, we will...
only present equivalencies for these two physical domains. For a more complete
treatment of the subject, see [6].

2.2.1 The Fundamental System Variables

When considering the dynamics of any system, the energy-content and therefore
the state of the system can always be described by a pair of kinetic and a pair
of kinematic variables, regardless of the domain. The kinetic variables are the
momentum $p(t)$ and effort $e(t)$, while the kinematic variables are the displacement
$d(t)$ and flow $f(t)$. These are called the fundamental system variables.

A displacement $d(t)$ is what is conventionally identified as a coordinate of the system.
In the mechanical domain, it can be a linear or a rotational position. In the electrical
domain, it is a charge. The corresponding flow $f(t)$ is the time derivative of the
displacement:

\[
q(t) = \int f(t) dt \tag{2.7}
\]

A big advantage of these variables is that it makes it clearer that work, power and
energy can be defined in the same way regardless of the domain. For example, in
the mechanical domain work we say that a force does work on a body when the
body is displaced, whether the displacement is caused by the force or not. Then an
infinitesimal increment in work is defined as

\[
dW(t) = F(t)dx(t) = e(t) dq(t) \tag{2.10}
\]

where $W(t)$ is work, $F(t)$ is force and $x(t)$ is position. The corresponding power is
simply the rate of work over time, or effort times flow:

\[
P(t) = \frac{dW(t)}{dt} = F(t) \frac{dx(t)}{dt} = F(t)e(t) = e(t) f(t) \tag{2.11}
\]
where $P(t)$ is the power. Plugging in the corresponding efforts and flows of rotational or electrical coordinates, we see that we get exactly what would be expected:

$$
\begin{align*}
\text{General Power:} & \quad P(t) = e(t)f(t) \\
\text{Linear Mechanical:} & \quad P_{\text{lin}}(t) = F(t)\dot{x}(t) \\
\text{Rotational Mechanical:} & \quad P_{\text{rot}}(t) = \tau(t)\dot{\phi}(t) \\
\text{Electrical:} & \quad P_{\text{elec}}(t) = V(t)I(t)
\end{align*}
$$

where $\tau(t)$ is a torque, $\dot{\phi}(t)$ is the corresponding flow, an angular velocity; $V(t)$ is a voltage and $I(t)$ is the corresponding current. Note that while the usual notation for current $I(t)$ doesn’t explicitly show it as a time-derivative, it is: current is the flow of charge, i.e. $I(t) = \dot{Q}(t)$. This is equivalent to the usual notation of angular velocity as $\omega(t) = \dot{\phi}(t)$.

Also in all cases, energy is defined as the time integral of power, and therefore of the effort times the flow:

$$
E(t) = \int_{t_0}^{t} e(t)\dot{f}(t)\,dt
$$

### 2.2.2 Unified System Components

So far we’ve seen how kinetic and kinematic variables are related through power, however they also relate through specific system component. The most important three are the ideal inductors, resistors and capacitors. The naming of these components makes it clear what they represent in the electrical domain; their counterparts in the mechanical domain are presented in table

<table>
<thead>
<tr>
<th>Domain</th>
<th>Electrical</th>
<th>Mechanical</th>
<th>Rotational Mechanical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Displacement</td>
<td>Charge $Q(t)$</td>
<td>Position $x(t)$</td>
<td>Angular Position $\phi(t)$</td>
</tr>
<tr>
<td>Flow $f(t)$</td>
<td>Current $I(t)$</td>
<td>Velocity $\dot{x}(t)$</td>
<td>Angular Velocity $\dot{\phi}(t)$</td>
</tr>
<tr>
<td>Momentum $p(t)$</td>
<td>Flux $\lambda(t)$, Linkage</td>
<td>Mechanical Moment $\dot{p}(t)^2$</td>
<td>Angular Momentum $L(t)$</td>
</tr>
<tr>
<td>Effort $e(t)$</td>
<td>Voltage $V(t)$</td>
<td>Force $F(t)$</td>
<td>Torque $\tau(t)$</td>
</tr>
<tr>
<td>Ideal Inductor</td>
<td>Inductor</td>
<td>Point Mass</td>
<td>Moment of Inertia</td>
</tr>
<tr>
<td>Ideal Resistor</td>
<td>Resistor</td>
<td>Damper</td>
<td>Rotational Damper</td>
</tr>
<tr>
<td>Ideal Capacitor</td>
<td>Capacitor</td>
<td>Spring</td>
<td>Rotation Spring</td>
</tr>
</tbody>
</table>

Table 2.1: Equivalents of electrical and mechanical domains.

The way these components relates the universal variables of state is best shown with the so-called tetrahedron of state, seen in figure 2.2.

In short, both springs and capacitors store a displacement and release a proportional force. Inductors, mass and moments of inertia store momentum and release a proportional flow. Both electrical resistors and mechanical dampers dissipate energy by creating an opposing effort (whether a voltage, force or torque) proportional to the corresponding flow (current, linear or angular velocity).
The Tetrahedron of State

\[ \begin{align*}
\frac{d}{dt} & e(t) \\
\int dt & C \\
R & \frac{d}{dt} d(t) \\
L & \frac{d}{dt} f(t) \\
p(t) & \text{Momentum} \\
e(t) & \text{Effort} \\
d(t) & \text{Displacement} \\
f(t) & \text{Flow} \\
C & \text{Capacitance} \\
R & \text{Resistance} \\
L & \text{Inductance}
\end{align*} \]

Figure 2.2: The tetrahedron of state neatly shows the relationship between the four fundamental system variables, regardless of their domain.

### 2.2.3 The DC-Motor as an Electro-Mechanical Example

During this thesis, one of the most important robot components analyzed is the electro-motor, so let’s use this for our example. DC-motors are often modeled as in figure 2.3.

The electrical dynamics are modeled as a voltage source $V_s$ in series with a resistor $R$, an inductor $L$ and a back-emf voltage $V_{emf}$. This is coupled to the mechanical dynamics of the motor-shaft, generally considered as a torque $\tau_m$ and a rotational degree of freedom $\phi(t)$ having a certain moment of inertia $\Theta$.

To identify the number of coordinates of a system, it is useful to identify all the ideal inductors. This system has 2: the electrical inductor $L$ of the motor and the moment of inertia $\Theta$ of the motor shaft. Correspondingly we have 2 sets of fundamental system variables.

Setting up the equations of motion gives us

\[ \begin{align*}
L \frac{d}{dt} I(t) &= (V_s(t) - V_{emf}(t)) - RI(t) \\
\Theta \ddot{\phi}(t) &= \tau(t)
\end{align*} \]

Note that the electrical effort relating to the identified inductor is the voltage drop across the inductor: ergo $V_s(t) - V_{emf}(t)$. We introduce the electro-mechanical coupling as the following constraints:

\[ \begin{align*}
\tau(t) &= k_t I(t) \\
V_{emf}(t) &= k_t \dot{\phi}(t)
\end{align*} \]

where $k_t$ is the motor constant. This finally give us the following coupled equations of motion:

\[ \begin{align*}
L \frac{d}{dt} I(t) &= (V_s(t) - k_t \dot{\phi}(t)) - RI(t) \\
\Theta \ddot{\phi}(t) &= k_t I(t)
\end{align*} \]

To hit home the equivalence, we will expand the model to include some viscous damping (i.e. friction) at the motor shaft. The effect of this friction is an effort, i.e.
2.3 Positive Work, Negative Work and the Hopping at Natural Frequency

The key concept being exploited at the Bio-Inspired Robotics Lab is natural frequency. All of our hoppers are driven at their natural frequency and settle into stable limit-cycle gaits. A stable limit-cycle defines a behaviour where the phase-plot of a system describes a closed loop, and nearby trajectories spiral into the limit-cycle, as seen in figure 2.4.

The advantage of these gaits is that at their resonant frequency, very little energy is lost from external work. To clarify, we define work as either positive or negative: considering an arbitrary effort, we say that an effort $e(t)$ does positive work on a system when it has the same direction i.e. sign as its corresponding flow $f(t)$. It does negative work if the signs are opposite. Positive work increases the total energy of the system. Negative work removes energy from the system.

**NOTE**: we will consider work done by an actuator, i.e. a control input, as unrecoverable. This is purely a system convention: there are of course systems, e.g. hybrid cars, where the negative work done by the actuator is stored, e.g. example in a battery. In this thesis, any time this is possible we will consider the storage component (which must be either an ideal capacitor or an ideal inductance, see 2.2) as part of the system model. When we talk about work, we mean work done on the system.
entire system and therefore changing the energy content of the entire system. By counter-example, the spring in the SLIP model does positive and negative work on the mass, but the energy of the system as a whole remains the same.
Chapter 3

Efficient Control of Segmented Beam Hoppers

Next to the curved-beam hoppers presented in the next chapter, recently some segmented-beam hoppers have been developed in the BIRL lab. Segmented-beam hoppers feature standard rigid bodies, with standard pull-springs providing elastic energy storage. The first generation segmented beam hopper, developed by Benedikt Mathis [14], proved the viability of this concept as an alternative design to curved-beam hoppers capable of efficient hopping gaits. The advantages of segmented beam hoppers with respect to the curved-beam hoppers are two-fold: First, they are much more straight-forward to model. Second, having a simple design and direct actuation of the joint, control can be applied much more effectively. This chapter focuses on the second part, in particular, how do we approach designing an energy-optimal controller for such a robot?

3.1 The hopping robot CHIARO

The segmented-beam hopper we used for our analysis is the CHIARO hopper developed by Fabio Giardina [9]. It’s design is explained in figure 3.1. This robot is the second generation of segmented-beam hoppers. It’s construction uses light-weight rigid carbon-fibre tubes for the leg-structures and curved wooden feet. Mass is concentrated at the motor which directly actuates the joint in parallel to a pair of pull-springs. This configuration of springs is equivalent to a single virtual push-pull spring. The large pulley system attached to the motor allows a relatively high gear-ratio, without a large sacrifice in efficiency. The large curved foot-plate provides self-stability. The actuator is a Maxon Motors EC 45 Flat motor with 70 Watts power, controlled by a Maxon ESCON 50/5 motor controller. This motor controller allows torque control: we can easily choose a open-loop torque trajectory for the motor to follow, and the ESCON 50/5 takes care of the close-loop current control. Because of this, it is possible for us to ignore the motor dynamics, as the ESCON controller ensures that the motor output is what we actually expect. This is not the case of the curved-beam hoppers, where the motors are generally given a constant voltage as input.

\[^1\] Torque is directly proportional to current in electro-motors, making torque-control and current-control equivalent.
Figure 3.1: The CHIARO hopper used for the analysis. The design was kept as light as possible, with most of the 0.7 Kg mass concentrated at the motor. The motor actuates the single joint through a drive belt over a large pulley. The springs act in parallel to the actuator. A large, curved foot-plate provides both lateral and sagittal stability. This foot-plate is designed so that it can be easily replaced or adjusted. A key result from Giardina’s work is that the stable limit-cycle behaviour of the robot is strongly influenced by the curvature of the foot-plate and the angle of attack with which it is fixed to the leg.
3.2 Separating the stability problem from the energy-optimality problem

For most systems, robotic or not, control traditionally has the role of maintaining stability. We use a novel approach based on the following hypothesis: we can disregard stability in the design of an energy-optimal controller. This claim is based on the principle of *morphological computation*. Morphological computation is when a computational task no longer needs to be computed because the problem has been offloaded to the physical morphology of the system. During the design of CHIARO, Giardina showed the robot was self-stable\(^2\) through a series of experiments. Specific design parameters, namely the foot curvature and foot angle-of-attack were modified. For each combination of these two parameters, the robot was driven several times with an open-loop sinusoidal torque at different frequencies. If the robot fell over, that frequency was considered an unstable input, if it didn’t fall over, it was marked as a stable input. For many combinations of foot curvature and foot angle-of-attack were modified, the robot would settle into a stable limit-cycle for a wide range of frequencies. This indicates a large region of self-stability of the robot for sinusoidal control inputs, as represented in figure 3.2.

This is a key feature of the CHIARO robot. In appendix A we show that this concept of self-stability can be extended to a wide range of periodic control inputs and more importantly, that energy-optimal control inputs must be stable. This allows two things: first, the design of an energy-optimal controller can be done unrestricted by stability issues. Second, this assumption allows for the design of a periodic open-loop controller input, since for any periodic control input we assume that

\(^2\)Self-stable it is meant that the robot settles into a stable limit-cycle for a wide range of open-loop control inputs. In other words, feed-back control is unnecessary to achieve stable limit-cycle gaits.
the system will settle into a stable limit-cycle. It is important to note that this gait optimization is done over two stages: indeed, for each specific combination of design parameters, a specific control input frequency was found to yield the lowest Cost Of Transport. Only by iterating over both the design parameters as well as the control parameter is it possible to identify the overall most efficient gait. This is due to the fact that, by delegating stability to the morphology, important factors such as the lift-off and landing angle of the robot are determined by the physical morphology and no longer by the controller input. Thus, whenever we speak of an energy-optimal control input, it is optimal for a specific morphology.

3.3 Bang-Bang for CHIARO

Bang-bang controllers (see subsection 2.1.2) are attractive and often used in practice due to their simplicity. In our case, if the energy-optimal controller can be assumed to be a bang-bang controller it is actually not strictly necessary to solve the switching function; finding the energy-optimal controller can be can be reduced to a parameter search, as will be explained in section 3.5. However, equation (2.5) shows that the optimal controller is only proven to be a bang-bang controller if the Hamiltonian function is linear in $u(t)$. In our case, the system dynamics $\dot{\vec{x}}(t;\vec{x}(t),u(t))$, while being complicated and non-linear in the state variables, are indeed linear in the motor torque, our only control input\(^3\). Recalling that the Hamiltonian function has the following form

$$H(\vec{x}(t),u(t),\vec{p}(t)) = g(\vec{x}(t),u(t)) + \vec{p}(t)^T \dot{\vec{x}}(t;\vec{x}(t),u(t))$$

the only remaining condition for the Hamiltonian function to be linear is that the cost function $g(\vec{x}(t),u(t))$ also be linear in $u(t)$. Note that the vector notation on $u(t)$ has been dropped because we only have a single control input for our robots. Thus, for us to be able to claim that the energy-optimal control input is bang-bang, we must find a cost function $g(\vec{x}(t),u(t))$ that fulfills two conditions:

1. The cost function $g(\vec{x}(t),u(t))$ must be linear in $u(t)$.

2. Minimising $\int g(\vec{x}(t),u(t)) \, dt$ must be equivalent to minimising energy-input.

3.4 Picking a Equivalent Cost Function

A common cost function used for energy optimization problems is the quadratic cost on the control input: $u(t)^2$. The objective of this cost function is to minimise the torque to zero and not to negative, since the sign only indicates direction. The square elegantly removes the sign problem, while making it nicely convex and penalising larger errors exponentially more. Most importantly, quadratic problems have been extensively studied and easy to set up and dispose of powerful algorithms to solve, meaning numerical solutions can be found for readily, including for legged gaits [22]. Unfortunately in our case this is unsatisfactory as it would violate the linearity requirement. Further, minimising torque is only equivalent to optimising for energy under certain conditions, since energy depends on both effort and flow i.e. both torque and angular velocity and at the same time voltage and current. So how do we formulate a cost function that is equivalent to minimising negative work and is linear in $u(t)$?

The cost function we will choose is the following:

\(^3\)See [9] for the dynamics of a model of the CHIARO robot validated with real-world results.
3.4. Picking an Equivalent Cost Function

\( g(\vec{x}(t), \vec{u}(t)) = \sum_{i=1}^{n} R_i f_i(t) \) \hspace{1cm} (3.1)

subject to \hspace{1cm} (3.2)

\( u_j(t) = [0, u_{jMax}]\) \hspace{1cm} (3.3)

\( u_j(t) = [u_{jMin}, 0] \) \hspace{1cm} (3.4)

where \( f_i(t) \) are the flow variables of the system and \( R_i \) are the ideal resistors (e.g. damping or resistance) related to those flows, as described in 2.2. In other words, our cost function is to minimise the damping losses of the system. To the cost function, we’ve added the following constraint: the control inputs (which we assume to be effort variables) are not allowed to do negative work, they must exert effort in the same direction as their corresponding flow. In the subsection we will explain this concept more clearly through an example and show under what conditions it is equivalent to minimising energy input.

3.4.1 Minimizing Damping Losses for a 1-D Hopper

To clearly explain our solution we will introduce very simple hoppers, represented in figure 3.3. The dynamics of these hoppers are much simpler, but the linearity of their dynamics in terms of \( u(t) \) is the same as in the segmented-beam hoppers, making them equivalent for our purpose.

This system has only a single coordinate making it easier to analyse, and it’s equations of motion are as follow:

\[
\begin{align*}
\dot{x}_1(t) & = \dot{y}(t) \\
\dot{x}_2(t) & = \ddot{y}(t) = \frac{k_s(y_0 - x_1(t)) - dx_2(t) - g + u(t)}{m}
\end{align*}
\]
where $\vec{x}(t)$ is the state vector, formed by the displacement variable $y(t)$ and the flow variable $\dot{y}(t)$: the vertical position and velocity, respectively. $k_s$ and $d$ are the spring and damper constants respectively, $g$ is the gravitational pull, $u(t)$ is the the control input, i.e. the force acting on the mass, and finally $m$ is the mass. For clarity, from now on we will use $y(t)$ and $\dot{y}(t)$ directly instead of the state vector $\vec{x}(t)$, and we will assume $m$ to have unitary value.

For this system, the solution proposed ((3.11)) takes the following form:

$$g(y(t), u(t)) = d\dot{y}(t)$$

(3.5)

$$u(t) = [0, u_{\text{min}}] \text{ if } \dot{y}(t) < 0$$

(3.6)

$$u(t) = [0, u_{\text{max}}] \text{ if } \dot{y}(t) \geq 0.$$  

(3.7)

In the following it will be shown why this is equivalent to minimising energy input. It is assumed that we are at steady-state hopping with a peak hopping height of $h_0$ and no energy losses during flight-phase. We will further limit ourselves to the stance-phase: this is done on the assumption that the force of the control input $u(t)$ is achieved by the leg pushing against the ground. During flight the robot has nothing to push against and therefore cannot achieve any force.$^4$ Switching between flight and stance phase is determined by the crossing of $y(t) = y_0$, where $y_0$ is the unsprung leg length. Velocity at touch-down can be calculated by the change in potential energy from maximum hopping height $h_0$ to touch-down height $y_0$, as $\dot{y}(TD) = -v_0 = -\sqrt{2mg(h_0 - y_0)}$, where $TD$ indicated time at touch-down. To fulfill the assumption of steady-state hopping, kinetic energy at lift-off must be the same as at touch-down, ergo $\dot{y}(TO) = v_0$. So the boundary conditions of the stance phase are:

$$y(TD) = y_0$$

$$\dot{y}(TD) = -v_0$$

$$y(TO) = y_0$$

$$\dot{y}(TO) = v_0$$

Every state pair $\{y(T), \dot{y}(T)\}$ (with $T$ an arbitrary point in time) represents a specific energy content, as together they represent the total potential energy (both from gravitational pull and energy stored in the spring) as well as kinetic energy. The initial and terminal state-pairs are completely equivalent, not only in total energy but also in division between potential and kinetic energy. The sole difference is the velocity having opposite direction. In the case of an ideal spring-mass system with no losses (figure 3.3a), the system’s state trajectory will automatically move from initial to terminal condition, as seen in figure 3.4.

For easier comparison, the position $y(t)$ during stance is mapped to a continuous phase between $-\pi$ representing touch-down and $\pi$ representing take-off, with the zero-point representing maximum compression, at which point $\dot{y}(\text{phase}=0) = 0$, as represented in figure 3.5. This is a piece-wise linear mapping, splitting the stance-phase into two equivalent phases: spring-compression, spanning from $-\pi$ to 0, and spring-expansion, spanning from 0 to $\pi$.

Actuator input is required once damping is considered, as in model 3.3b), as the energy lost from damping will result in a lower velocity, and therefore lower total energy, at take-off, as seen in figure 3.6.

$^4$For the CHIARO robot similar arguments can be made indicating energy-input during flight should be negligible. In either way though it does not affect the resulting cost-function: dealing with only stance-phase merely simplifies the explanation.
Figure 3.4: A phase plot of the state variables for an undamped hopper at stance: velocity over position. In the undamped hopper control input is unnecessary, as the state-trajectory moves naturally from the initial condition (red circle) to the terminal condition (green triangle).

Figure 3.5: For the purpose of comparison, the stance phase is mapped to a phase spanning from $-\pi$, representing touch-down, to $\pi$, representing take-off, with zero representing maximum compression where velocity is also zero.
Figure 3.6: Damping losses in the system are directly proportional to the velocity of the system, in other words to the area under the curve. To ensure steady-state hopping, energy must be added again to the system to push the velocity to the terminal condition (green triangle); doing this however also increases the damping losses, making the problem non-trivial.

The task of control is to then add energy to the system to push the trajectory back up to terminate at the proper terminal condition, while minimising energy-input. The path of the trajectory is important as the damping factor penalises higher speeds\(^5\), making the control input non-conservative. The most intuitive cost-function would be to simply minimise work of the actuator:

\[ g(y(t), u(t)) = |u(t) \cdot \dot{y}(t)| \]

but the absolute value, which is necessary since both negative and positive work should to be minimised, makes the cost function non-linear. Considering that the boundary conditions fix the energy at take-off, and the fact that during stance stability considerations are unnecessary, the input force must do a total amount of positive work equal the total amount of negative work done on the system. This leads us to two conclusions: first, the control input should never do negative work. All the negative work done by the actuator must also be replaced by the actuator. This leads to the conditions on \(u(t)\) in (3.5). The second conclusion is that minimising the negative work performed by the damper is equivalent to minimising the energy input of the actuator. Taking a closer look at the energy losses, we see that

\[
F_d = -d\dot{y}(t)
\]

\[
W_d = \int P_d(t)dt = \int F_d\dot{y}(t)dt = \int -d\dot{y}(t)dt = -d\int \frac{dy}{dt^2}dt = \int -d\dot{y}(t)dy.
\]

Since the mapping of the position to phase is a linear mapping\(^6\) this is equivalent

---

\(^5\)We assume a standard linear damper, proportional to velocity.

\(^6\)It involves a mere scaling and mirroring about the maximum-compression point.
3.4. Picking an Equivalent Cost Function

Figure 3.7: The blue line represents an energy-optimal trajectory with limits placed on the control input $u(t)$. The shaded green area is the bounded area between the passive damped and undamped system trajectories, in which the optimal control input trajectory should move, depending on its limits.

to minimising the absolute of the velocity over the stance phase. If we combine these two conclusions, we get the following cost:

$$g(y(t), u(t)) = |\dot{y}(t)|$$  (3.8)

$$u(t) = [0, u_{\text{min}}] \quad \text{if } \dot{y}(t) < 0$$  (3.9)

$$u(t) = [0, u_{\text{max}}] \quad \text{if } \dot{y}(t) > 0.$$  (3.10)

The constraints on the control input in (3.5) is to ensure that the control input only contributes positive work to the system. Doing positive work with the control input $u(t)$ means increasing the velocity $\dot{y}(t)$, thus pushing the trajectory away from the 0-axis and increasing damping force. While it may seem tempting to do negative work with $u(t)$ to minimise damping forces, the actuator would simply be replacing the physical damping and acting as a virtual damper with even higher damping coefficient. The constraint on $u(t)$ sidesteps this problem.

Since these constraints do not allow the control input to push the velocity trajectory closer to 0, the passive, damped trajectory (solid blue line) represents the inner limit. And since the cost function minimises the curve under the trajectory, the passive undamped trajectory (dashed black line) represents the outer limit. From this, the minimizing control input would be to allow the hopper to passively follow the damped trajectory until the very last instant, then apply an infinite force to bring the trajectory to the terminal condition. Since real-world actuators are limited in the amount of force or torque they can apply, it becomes necessary to input the maximum force over as short a period of time as possible, as late as possible, as shown in figure 3.7.

3.4.2 Extending the Cost Function

Thus we have proved that minimizing energy input is equivalent to minimizing the damping losses when hopping at natural frequency. In the most general formulation
3.11 (see below) this has simply been extended to include an arbitrary number of coordinates:

\[ g(\vec{x}(t), \vec{u}(t)) = \sum_{i=1}^{n} R_i f_i(t) \]  
\[ \text{subject to} \]  
\[ u_j(t) = [0, u_{j,max}] f_j(t) \geq 0 \]  
\[ u_j(t) = [u_{j,min}, 0] f_j(t) < 0 \]  

where \( f_i(t) \) are the flow variables of the system, and \( R_i \) are the dissipation (e.g. damping or resistance) related to those flows, as described in 2.2. This generalized formulation also allows the consideration of damping losses in other domains, such as the internal resistance of the motor. In electric motors, power loss stems from the internal resistance of the motor and is proportional to current. This fits perfectly with our original statement (3.11): as we saw in section 2.2, current is the flow variable in the electrical domain, and electrical resistors are equivalent to mechanical dampers. Taking further into consideration that motor torque is directly proportional to the motor current via the motor constant, the cost function of our simple 1-D hopper example can be extended as so:

\[ g(\vec{x}(t), \vec{u}(t)) = b \dot{y}(t) + R_{mot} I(t) = b \dot{y}(t) + R_{mot} \frac{u(t)}{k_t} \]

where \( k_t \) is the motor constant. While it is easy for us to write it this way, such that we can still consider the system as the familiar 1-D hopper, we’ve actually already extended it to 2-D: one dimension being the vertical displacement \( y(t) \) in the mechanical domain, the other being the displacement of charge \( Q(t) \) in the electrical domain. Recall that the current \( I(t) \) is a flow variable and merely the time derivative it’s corresponding displacement: i.e. \( I(t) = \frac{dQ(t)}{dt} \). In a similar fashion, the system can be extended to arbitrary degrees of freedom. Impact losses can also be accounted for as an instantaneous loss of kinetic energy at impact. This would merely change the initial boundary condition of the stance-phase \( \{y(T_D), \dot{y}(T_D)\} \) to contain less energy than the terminal condition of lift-off. The control input would then have to not only replace energy lost through damping but also that lost through impact.

3.5  Heuristic Parameter Search in the Real World

In the previous sections we have, under specific circumstances, proven that the energy-optimal control input must be a bang-bang controller. However, we do not know yet what exact shape it will have. In the following, we will show that as an alternative to solving the switching function of the Hamiltonian, it is theoretically possible to find the optimal control input via parameter search. Then, we will present a reduced parameter search that is practically feasible and apply it to finding an energy-optimal control input for the CHIARO robot.

3.5.1  How to Find the Switching Function via Parameter Search

We claim that finding the switching function for the energy-optimal control input can be reduced to a parameter search. This claim hinges on an important assumption: the energy-optimal control input is periodic. To justify this assumption, we remind the reader that we are searching for the energy-optimal control input
Some arbitrary, possible bang-bang trajectories are shown in the figures above. In the previous section it was proven that the energy-optimal control input must have a bang-bang shape, with the bangs being of value \([u_{\min}, 0, u_{\max}]\) depending on the sign of the flow variable; because of this, none of the above control trajectories can be excluded outright.

Stable limit cycles are per definition periodic. We can then solve the optimization problem over a single cycle, which, repeating itself, ensures that the total infinite-horizon solution is also periodic. Therefore, not only must the energy-optimal control input be periodic, it’s period is the same as the stable-limit cycle.

Now that we have determined that the control input is periodic, we can tackle the question: how do we determine its shape? In the previous section we proved only that the controller must be a bang-bang controller. The number and timing of the switches can be arbitrary, e.g. any of the trajectories of \(u(t)\) shown in figure 3.8 is theoretically possible.

Since the control input is periodic, it is possible to represent by decomposing it into a series of adequately chosen periodic functions, such as a Fourier series

\[
u(t) = \frac{a_0}{2} + \sum_{n=1}^{N} [a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right)]
\]

where \(P\) is the period and \(a_n\) and \(b_n\) are the coefficients. Since it is known that \(u(t)\) is bang-bang, the decomposition can be further simplified. For example, it would make sense more sense to use a square wave instead of sin and cos waves.

We’ve chose an even simpler approach. Instead of directly representing \(u(t)\), we search for a direct equivalent to the switching function: since the control input \(u(t)\) is periodic, the zero-crossings of the switching function must also be periodic. Hence, finding a fourier series with the same zero-crossings is equivalent to finding the switching function itself.

### 3.5.2 Reducing the Parameter Space

The final test of a theory is its capacity to solve the problems which originated it.

-George Dantzig
Figure 3.9: The switching function has a mere 3 parameters which can be easily swept through: Frequency $f$, offset and $u_{\text{max}}$. The sign of the switching function determines whether the control input is the high ban $u_{\text{max}}$ or 0. The offset determines the duration of the bang by shifting the sinusoid up or down: a positive offset will result in a longer $u_{\text{max}}$, a negative offset in a shorter $u_{\text{max}}$.

Previously we’ve shown that finding the proper switching function can be reduced to a parameter search of a fourier series. In other words, it is equivalent to a system identification problem. Instead of searching the infinitely large parameter space, the space is can be greatly reduced through the use of heuristics, e.g. by cutting the high-frequency content. In defining a set of parameters, we’ve closely followed the design principle of ecological balance: i.e. there should be a balance in the complexity of the physical morphology and the control of the system. To this end, we have defined the simplest switching function possible likely to produce good results:

$$f_{\text{switch}}(t) = \sin(2\pi ft) + \text{offset}$$  \hspace{1cm} (3.15)  

$$u(t) = u_{\text{max}} \text{ if } f_{\text{switch}}(t) \geq 0$$ \hspace{1cm} (3.16)  

$$= 0 \text{ if } f_{\text{switch}}(t) < 0$$ \hspace{1cm} (3.17)  

where $f_{\text{switch}}(t)$ is the switching function. The search-space contains a single frequency, $f$, an offset $\text{offset}$ and the maximum value $u_{\text{max}}$. The resulting trajectory of the switching function and control input can be seen in figures 3.9. The frequency is used to match the natural frequency of the hopping gait. The offset is used to determine the duration of the bang. Finally, the torque applied during the bang is limited to $u_{\text{max}}$ in order to limit the total energy input to the system. This is dependent on both the duration (and therefore the offset) and $u_{\text{max}}$; we have found however through testing that with extremely high torques the hops become more aggressive, less stable and less efficient. The decrease in efficiency is presumably due to increased impact losses. Since kinetic energy scales quadratically with velocity, so do the impact losses of unsprung mass: while they might be negligible for medium velocities, they become exponentially more costly at higher vertical velocities.

It may be surprising that the control input is limited to $[0, u_{\text{max}}]$. This is equivalent to limiting the motor to do work in a single direction. There are two reasons for this. First, since the CHIARO robot has only a single actuator and a single spring, it is reasonable to approximate the energy-optimal control input with a single direction, that of spring expansion. Second, extending the search space to include $u_{\text{min}}$ is not trivial as it brings the complication of coordinating two separate signals. We therefore argue that this is a first complexity wall, and to surmount goes against
the principle of ecological balance. Our experimental results lend weight that this simplification does not sacrifice much in efficiency.

### 3.5.3 Real-World Results with CHIARO

Having reduced the parameter space to a mere 3 parameters, sweeping through it becomes very easy to do. Our best result was found after roughly an hour of testing on the best model of all: the actual robot. The parameters and the resulting performance are shown in table 3.5.3, next to the best results using a sinusoidal input. For comparison, the sinusoid control input has the shape

\[ u_{\text{sin}}(t) = C \sin(2\pi ft) \]

where \( C \) is the maximum current and \( f \) is the frequency. The physical configuration used for CHIARO is the same in both cases. We remind the reader that the stable limit-cycle, and therefore the optimal gait, depends on both the control input as well as the physical design parameters. While the results presented here are specific to these design paramters, it also keeps the comparison with the sinusoidal input unbiased.

<table>
<thead>
<tr>
<th>Control Input</th>
<th>Bang-Bang</th>
<th>Sinusoid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency ([Hz])</td>
<td>4</td>
<td>3.7</td>
</tr>
<tr>
<td>Maximum Current ([A])</td>
<td>(u_{\text{max}} = 0.57)</td>
<td>(C = 0.33)</td>
</tr>
<tr>
<td>Offset</td>
<td>-0.7</td>
<td>-</td>
</tr>
<tr>
<td>Cost of Transport</td>
<td>0.49</td>
<td>1</td>
</tr>
<tr>
<td>Forward Velocity ([\frac{m}{s}])</td>
<td>0.25</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Interestingly the bang-bang controller achieved a better result at a slightly higher frequency and higher forward velocity. The result is most impressive however when put into context. The CHIARO hopper uses hopping gaits, so an appropriate biological system to use as a benchmark should also use a hopping gait. The closest animal to which we have data at this time is the tammar wallaby, the smallest of the wallaby family with an average weight of about 5 Kgs and 7 Kgs for females and males respectively, and an average height of 45 cm. The cost of transport for Tammar wallabies has been recorded at 0.4373 [23]. From this, we see that the bang-bang input is not simply twice as efficient as the previous sinusoidal input: **it is over ten times closer to it’s biological counterpart!**
Chapter 4

C-shaped Curved Beam Hoppers: electro-mechanical dynamics

C-shaped hoppers, such as the one in figure 4.1, are designed around an elastic curved beam: like a hunting bow, these curved beams have a stiff elastic property with very little damping. By using the curved beam as both the main body-structure and the spring, it is possible have both very low damping in the spring as well as minimize unsprung mass. A large foot is used at the bottom of the spring to guarantee stability, and a simple DC-motor and a payload are fixed to the top of the elastic curved-beam. The only joint connects the motor to a pendulum. The motor inputs energy to the system by spinning the pendulum: with the proper choice of rotational speed, the system is excited at it’s natural resonant frequency and begins to hop.

One of the benefits of this actuation is that the actuator never has to support the entire payload, so a very low-power motor is adequate. In various trials, power input has even been selected so low that the motor cannot start swinging the pendulum from it’s resting position by itself, but requires an initial push\(^1\): once the swinging was started, the power input was able to make up for the damping losses and to locomote properly. On the other hand, the indirect actuation makes it difficult to properly understand the system dynamics. It becomes even more daunting to consider the dynamics both in the mechanical as well as the electrical domain. In this chapter we will explain however that this is both very important as well as very useful.

4.1 Modeling the Electro-Mechanical Dynamics

In this thesis, we propose a new model of the hopper which include the electro-mechanical dynamics of the motor-pendulum system. This is done to better understand how to control the system efficiently, however it also sheds light on the effects of scaling on the dynamics.

\(^1\)Much in the same way old prop-airplanes required someone to give the propeller an initial swing.
4.1.1 Previous Models of the C-Shaped Hopper

So far, various simplified models of the system, such as those seen in figure 4.2, have been proposed [27] [21], all of which make three assumptions:

- The pendulum rotates at constant angular velocity and therefore
- The resulting force on the principle mass is the centripetal force of the pendulum and can therefore be modeled as a sinusoidal input in the horizontal and vertical axes.
- The motor never performs negative work since the pendulum’s angular velocity is constant.

For the case of gearless motors\(^2\), this simplification has however been found unsatisfactory for the purpose of control [28]. This is because in the simplified models, the motor-pendulum system is treated as a black-box: it’s effect on the hopper is modeled but not it’s own dynamics. Therefore the effect of the hopper on the motor-pendulum system is ignored. Gearless electro-motors are highly backdriveable, so their electrical dynamics are strongly coupled to the mechanical dynamics of the robot! In this thesis we propose a model of the hopper which includes an electro-mechanical model of the motor-pendulum system and explore its implications.

4.1.2 An Extended Model Including the Motor-Pendulum Dynamics

The focus of our work is on the electro-mechanical dynamics of the motor-pendulum system. As such, we have found it useful to use a simplified model of the curved-beam body. We base our model on one similar to the one proposed by Reis[21], which essentially replaces the curved-beam with a linear spring and rotation joint, as shown in figure 4.3 a). For clarity, we further simplify our model to hop in place, i.e. constrain the body mass to the vertical coordinate. We will refer to this part, i.e. the body mass and spring, as the hopping system. To this we add the

\(^2\)We remind the reader that reducing friction, and therefore gears, is one of the design principles.
4.1. Modeling the Electro-Mechanical Dynamics

Figure 4.2: These simplistic models effectively capture the dynamic behavior of the curved-beam hoppers without needing to fully describe the system.

pendulum’s rotational coordinate and the charge coordinate of the electro-motor\(^4\) to describe the motor-pendulum system. The full model can be seen in figure 4.3 b). The physical parameters of the model are shown in figure 4.4.

Just as was shown in 2.2.3, we proceed to derive the equations of motion of this system by first identifying the ideal inductors, of which there are three: the body mass \(m_b\), the pendulum mass moment of inertia \(\Theta_p = m_p l_p^2\) and electrical inductor of the motor \(L\). The state vector of the system is therefore identified as

\[
\vec{q}(t) = \begin{pmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \end{pmatrix} = \begin{pmatrix} y(t) \\ \phi(t) \\ Q(t) \end{pmatrix}
\]

where \(y(t)\) is the vertical position of the body mass, \(\phi(t)\) is the angular position of the pendulum and \(Q(t)\) is the electrical charge in the motor. Note that in the motor we’re only interested in the corresponding flow: the current \(\dot{I}(t) = \dot{Q}(t)\). However we will leave the notation \(Q(t)\) for consistency.

In the robot setup, we can only perform open-loop voltage control at the moment, meaning our control input vector is

\[
\vec{u}(t) = \begin{pmatrix} 0 \\ 0 \\ V_s(t) \end{pmatrix}
\]

4.1.3 Deriving the Equations of Motion

The equations of motion for this model are derived with the Euler-Lagrange method and can be perused in appendix B. They have the form of

\[
M(\vec{q}(t)) \ddot{\vec{q}}(t) = F(\dot{\vec{q}}(t), \vec{q}(t), \vec{u}(t))
\]

where \(M(\vec{q}(t))\) is the mass-matrix multiplying the state accelerations, and \(F(\dot{\vec{q}}(t), \vec{q}(t), \vec{u}(t))\) contains all other terms. The matrix \(M(\vec{q}(t))\) is

\(^4\)For more details on the electro-motor model, see subsection 2.2.3.
Figure 4.3: The model we use is based on that derived by Reis, a). For simplicity is constrained to hop vertically. This is done to more easily analyze the coupling with the motor-pendulum system dynamics, which add two dimensions as seen in b).

\[ F(t) = A \left( \cos(\omega t) \right) \sin(\omega t) \]

Figure 4.4: The model we use has the mechanical parameters listed in the figure.

- \( m_p \): pendulum mass
- \( l_p \): pendulum length
- \( m_b \): body mass
- \( k_s \): spring coefficient
- \( y_0 \): spring resting length
- \( d \): damping coefficient
- \( k_t \): motor constant
- \( R \): motor internal resistance
- \( L \): motor inductance
which is non-diagonal and clearly shows the implicit, non-linear coupling between the pendulum dynamics and the body mass dynamics. The electrical dynamics of the motor are linearly coupled with the dynamics of the pendulum. However, because our only control input is the open-loop source voltage \( V_s(t) \), it is important to include the electrical dynamics of the motor. These non-linear differential-algebraic equations can be solved as are by using the implicit-Euler method. Alternatively, we can solve them to yield explicit first-order differential equations of the form

\[
\ddot{q}(t) = M^{-1}(\ddot{q}) F(\dot{q}, q, u)
\]

which results in very long equations. However they can be directly integrated with the simpler explicit-Euler method. Also, it allows us to directly analyze the effects of scaling on each individual coordinate, which we will do in the following two subsections.

### 4.2 Analysis of Scaling Effects

The resulting equations of motions describing the system dynamics are as one might expect, highly non-linear and very complicated. To be able to make sense out of them, it is useful to make certain assumptions on the parameters in order to simplify the equations of motion by excluding negligible terms. In the following subsections we will do exactly this understand specific aspects of the curved-beam hopper behavior, shedding light on principles for designing hoppers with a predictable behavior.

#### 4.2.1 Source Voltage as an Indirect Control Input

The goal of our controller is to input energy to the hopping system, in other words we want to do positive work on the body mass \( m_b \). For this purpose it is helpful to explicitly describe the corresponding effort of the body mass coordinate: the vertical force exerted on the body mass by the motor-pendulum system. Through free-body diagram analysis this force can be calculated as

\[
F_y(t) = -m_p \ddot{y}(t) - l_p \cos(\phi(t)) \dot{y}^2(t) - l_p \sin(\phi(t)) \dot{y}(t) + g - \frac{\sin(\phi(t)) \tau_m}{l_p}
\]

\[
\tau_m = k_l \dot{Q}(t) = \frac{V_s(t) - k t \dot{\phi}(t)}{R}
\]

This shows the effect of our control input \( V_s(t) \) on the body mass is very indirect. The instantaneous-force exerted is much more dependent on the state of the system rather than on the control input, meaning it is more important to design the system to have an efficient limit-cycle than to control it.

This can be further exasperated by design: from equation (4.2.1) we see that the dependence of \( F_y(t) \) on the system state is directly proportional to the pendulum mass \( m_p \) and length \( l_p \), while it’s dependence on \( V_s(t) \) is inversely proportional to \( l_p \) and independent on \( m_p \).

---

4 Also known as backwards Euler method
With a proper selection of design parameters, the term $V_s(t)$ can therefore be made negligible, and the vertical force can be approximated as:

$$F_y(t) = -m_p(\ddot{y}(t) - l_p \cos(\phi(t))\dot{\phi}^2(t) - l_p \sin(\phi(t))\ddot{\phi}(t) + g)$$

### 4.2.2 Scaling to Emphasize Centripetal Force

In the previous subsection we analyzed the the vertical force exerted by the motor-pendulum on the hopper body to see the effect of the control input $V_s(t)$ on the hopping system. From equation (4.2.1) we can also understand under what parameter-set the effect of the motor-pendulum on the hopper can be approximated as a sinusoidal force source. This lends mathematical weight to the assumptions in made previous models (see subsection 4.1.1) and shows us how to design new hoppers with predictable behaviors.

From equation (4.2.1) we can directly see that picking a large pendulum length $l_p$ increases the importance of the centripetal force of the pendulum and minimizes that of the motor torque. Further, choosing small values for the pendulum weight $m_p$ minimizes the importance of the hopper’s acceleration as well as the weight of the pendulum, $m_pg$. Thus for long, light pendulums, the resulting force can be approximated as

$$F_y(t) = m_p l_p (\cos(\phi(t))\dot{\phi}^2(t) - \sin(\phi(t))\ddot{\phi}(t))$$

The second term cannot be simply neglected as it factors both $m_p$ as well as $l_p$. Instead, we will check the conditions under which the angular acceleration becomes minimal. Intuitively, this is the case when the impacts of the hopper landing at the end of a flight-phase do not influence the the dynamics of the pendulum.

To observe this, we make use of the previously derived explicit equations of motion 4.1.3. Having already made the assumption of a long and light pendulum, these equations of motion can be simplified to:

$$\ddot{y}(t) = \frac{k_s(y_0 - y(t)) - \frac{k_t \sin(\phi(t))}{l_p} Q(t) + m_p l_p \cos(\phi(t))\dot{\phi}^2(t) - m_b g}{m_b}$$

$$\ddot{\phi}(t) = \frac{k_t Q(t)}{m_p l_p} + \sin(\phi(t))\left[\frac{2g}{l_p} + \frac{d\ddot{y}(t) - k_s(y_0 - y(t))}{m_b l_p}\right]$$

$$\ddot{Q}(t) = \frac{R}{L} Q(t) + \frac{V_s(t) - k_t \dot{\phi}(t)}{L}$$

These equations show that the effect of impacts on the pendulum dynamics is minimized by selecting large values for pendulum length $l_p$ and body-mass $m_b$, as well as selecting low values for the spring’s inherent damping $d$ and the spring stiffness $k_s$. While the first three parameter conditions are generally respected in the hoppers built at BIRL, high spring stiffness is usually desired as it also important in determining the gait frequency[7][11].

Thus we see that the centripetal force of the pendulum is the dominant force acting on our hoppers, but not the only one!
4.2.3 Some Remarks on Energy-Optimal Control for the Curved-Beam Hopper

Like the segmented-beam hopper presented in chapter 3, the curved-beam hoppers also locomote by hopping at natural frequency. Therefore we can make the same claim: to minimize damping losses of the system is equivalent to minimizing energy input to the system. The energy-optimal control problem can therefore be set up with exactly the same approach as for the segmented-beam hopper. We have seen that the coupling of the efforts of each coordinate system to the control input \( u(t) = V_s(t) \) is very complicated but always linear! It should be of little surprise then that the energy-optimal control input should have a bang-bang shape.

Because the couplings are so complicated however, it becomes much more difficult to reduce the parameter space sufficiently to find the switching function via a parameter search, as we did for the segmented-beam hopper CHIARO. However, previous discussion on how scaling affects the system dynamics show that through proper selection of parameters the coupling of the various degrees of freedom can be greatly simplified. It is surmised that under the circumstances described, the energy-optimal control input could very well be that used in the real-world experiments: drive the motor at constant voltage. In other words, the switching function maintains the same sign all the time, so the optimal control input never changes. This would explain the spectacularly low Cost of Transport achieved in some hoppers[11], and only mediocre ones for other hoppers with only slight differences in design. So the true question we should asking ourselves is not 'what is the proper voltage to drive the hopper?' but rather *what is the proper morphology so that a constant voltage is the energy-optimal controller?* This new way of looking at things does not, however, detract the importance of control: rather, *it emphasizes the importance of considering the relationship between the controller design and the morphological design*, and indirectly, that they should be designed simultaneously instead of as two separate steps.
Chapter 5

Conclusions

In this thesis we have treated energy-optimal control for simple natural-frequency hoppers built at the Bio-Inspired Robotics Lab. The groundwork has been laid for cross-domain dynamics analysis of the hoppers as well as for applying optimal control theory, in particular for energy-efficient locomotion. Also, we’ve shown that for hopping at natural frequency, minimizing damping and impact losses is equivalent to minimizing energy-input for a system under the constraint of doing only positive work. With this equivalence, it was shown that the Hamiltonian function resulting from Pontryagin’s Minimum Principle is linear in the control input $\hat{u}(t)$: the energy-optimal control input must therefore have a bang-bang shape.

It has also been demonstrated that the control input for stable-limit cycle hopping must also be periodic: from this result, the switching function of the bang-bang control input can be represented as a Fourier sum and solved as a parameter search. In cases where the parameter space can be sufficiently reduced, this makes the search easy to find in the real-world. We have proposed a highly reduced parameter space for the segmented-beam hopper CHIARO, and achieved excellent real-world results: $CoT = 0.49$ at a forward velocity of $0.25\text{ m/s}$. As reference, the previous results with the CHIARO hopper using a sinusoid input yielded $CoT = 1$ at a velocity of $0.2\text{ m/s}$. The closest biological counterpart we could find data for is the Tammar Wallaby, with $CoT = 0.437$. Our results are therefore not only a big improvement on previous results, but also extremely close to the biological counterpart. The excellent results also emphasizes the principle of ecological balance: for systems with simple physical morphology, the neural control should be equally simple.

The model of the curved-beam hopper has also been extended to include the electro-mechanical dynamics of the motor-pendulum system. From the resulting equations of motion the following statements can be made:

- First, they show that the control input, the motor source voltage $V_{\text{in}}(t)$, has little direct effect on the body mass dynamics of the hopper. This suggests that feed-back control is of little importance. Designing the system’s morphology to enter efficient stable limit-cycles is much more important.

- Second, they show under what conditions the effect of the motor-pendulum system can be approximated as a sinusoidal force-source. The benefits of this are twofold. It gives us an important design principle to build hoppers with predictable behavior: the pendulum should be long and light compared to the body mass. Also it gives mathematical weight to the assumptions made in (to the best of my knowledge) all previous models of the curved-beam hopper.
Finally, they show that the energy-optimal control problem for the curved-beam hopper can be set up in the same way as discussed earlier, with the same conclusion: the optimal control-input must be bang-bang in shape. A reduction of the parameter space for this system was not explored. However it seems likely that with the same design-parameters discussed above this would lead to control-input that is currently being used: a constant voltage input or in terms of bang-bang control, a single constant bang. This is supported by the fact that, using this control input some curved-beam hopper demonstrate extremely high efficiency while other don’t. It also suggests an alternative question to finding the optimal control input: instead, we should ask ‘How do we design the morphology so that a given, simple control-input is optimal?’: This puts emphasis on the importance of morphological computation when designing efficient systems.

**Recommended Further Work** For the segmented-beam hoppers, for which we have accurate models that can be simulated, we recommend developing a numerical framework for solving the optimization problem proposed here. Numerically solving the differential equations obtained from PMP would further justify the current results. More importantly, it allow solving the optimization problem for more complicated system, where the parameter space cannot be reduced as easily or when the system in question renders the Hamiltonian function non-linear in $\vec{u}(t)$.

In the case of the curved-beam hoppers, we have here shed light on the dynamics of one of the most important components: the motor-pendulum system. However, there is still another critical component for which an accurate model remains elusive: the curved-beam spring. More specifically, the non-linear elasticity is the feature that is difficult to model but also what makes it so attractive as a spring. A systematic method for building curved-beam springs with predictable elasticity properties, and the identification of these properties would be very beneficial. After all, with the focus placed now more heavily on designing the morphology, it is fundamental to have the tools to do so!
Bibliography


Appendix A

Why energy-optimal control inputs must be stable

In his work, Giardina showed that for specific design parameter selections, the CHIARO hopper displayed large regions of stability to sinusoid torque inputs[9]. We will further extend this to arbitrary periodic control inputs by claiming the following;

If there exists a control input that leads to a stable-limit cycle with non-zero forward velocity, the energy-optimal control input must also lead to a stable limit-cycle. We now proceed to show this to be true.

Definitions

An unstable control input indicates the robot falls over and then converges to zero velocity. The average forward velocity \( v_f \) is defined as the average distance covered over time

\[
v_f = \frac{\int_0^\infty v(t)dt}{\int_0^\infty 1dt}
\]

where \( v(t) \) is the instantaneous velocity. An energy-optimal control input is an input \( u(t) \) which minimizes the Cost of Transport (CoT), defined as

\[
\text{CoT} = \frac{\mathcal{E}}{mgd} = \int_0^T \frac{\mathcal{P}(t)}{mgv_f} dt = \frac{\mathcal{P}_{avg}}{mgv_f}
\]

Where \( \mathcal{E}(t) \) is energy expended, \( \mathcal{P}(t) \) is power expenditure and \( \mathcal{P}_{avg} \) is the averaged power expenditure.

We treat this as an infinite-horizon problem, meaning \( T \to \infty \). Assuming all control inputs are efforts, the energy expended by the control input is

\[
\mathcal{E} = \int_0^\infty \mathcal{P}(t) dt = |\bar{e}(t) \cdot \bar{f}(t)| = |\bar{u}(t) \cdot \bar{f}(t)|
\]

where \( \bar{f}(t) \) is the flow corresponding to the actuator effort. As an example, for an electro-motor the effort would be the source voltage \( V_{\text{eff}}(t) \) and the flow would be the motor current \( I_{\text{eff}}(t) \). If the control input is taken to be the mechanical output of the motor, the effort would be the motor torque \( \tau_{\text{mech}}(t) \) and corresponding flow would be the rotational velocity of the shaft \( \omega_{\text{shaft}}(t) \).
Proof by contradiction  Let us assume that the energy-optimal control input is not stable, i.e. it causes the robot to fall over eventually and converge to zero velocity. The average forward velocity then tends towards zero, $v_f \to 0$. From equation (A.1) we see that power is greater equal to zero, and since the control input is a non-zero periodic input, the averaged power expenditure is strictly greater than zero, $P_{avg}$. Equation (A) shows that the Cost of Transport then tends towards infinity, $CoT \to \infty$.

From our statement, we have made the assumption that there exists a (possibly non-optimal) periodic control input that leads to a stable gait with $v_f > 0$. In order to not be the energy-optimal control input, this input must have a higher $CoT$ than the energy-optimal input, ergo it must have a $CoT$ that tends towards infinity more quickly. We will now show that the averaged power expenditure of any stable limit-cycle gait is bounded, meaning the $CoT$ is also bounded and therefore lower than that of any unstable gait.

We have previously defined a stable gait as one that converges towards a stable limit-cycle. In a limit cycle the phase portrait of the kinematic variables $\{d(t), f(t)\}$ of the system describe a closed loop. The kinematic variables together describe the total energy content of the system: the displacement determines the potential energy and the flow determines the kinetic energy. Since in a limit cycle the kinematic variables are bounded, then so is the energy content of the system. This in turn means the power applied to the system, i.e. the rate at which the system energy content can be changed, is also bounded, and the averaged power expenditure of the control input cannot tend to infinity.

---

\(^1\)Displacement $d(t)$ and flow $f(t)$. For more, see chapter 2.2
Appendix B

Equations of Motion for the 1-D Hopper with Motor-Pendulum

These equations of motion are derived with the standard Euler-Lagrange approach. For more on Euler-Lagrange equations for non-mechanical systems, see [26][6]. The equations of motion are:

\[ M(\ddot{\mathbf{q}}(t)) \dot{\mathbf{q}}(t) = F(\dot{\mathbf{q}}(t), \mathbf{q}(t), \mathbf{u}(t)) \]

The state vector and control input vector are:

\[
\mathbf{q}(t) = \begin{cases} y(t) \\ \phi(t) \\ Q(t) \end{cases} \quad \mathbf{u}(t) = \begin{cases} 0 \\ 0 \\ V_s(t) \end{cases} \tag{B.1}
\]

The matrices are:

\[
M(\ddot{\mathbf{q}}(t)) = \begin{bmatrix} (m_b + m_p) & -m_p l_p \sin(\phi(t)) & 0 \\ -m_p l_p \sin(\phi(t)) & m_p l_p^2 & 0 \\ 0 & 0 & L \end{bmatrix}
\]

\[
F(\ddot{\mathbf{q}}(t), \mathbf{q}(t), \mathbf{u}(t)) = \begin{bmatrix} m_p l_p \cos(\phi(t)) \dot{\phi}(t) - m_b g + k_s (y_0 - y(t)) - d \dot{y}(t) \\ k_s \dot{Q}(t) + m_p l_p \sin(\phi(t)) g \\ V_s(t) - k_t \phi(t) - R \dot{Q}(t) \end{bmatrix}
\]

These equations can be solved to yield explicit equations of motion in the following form:

\[ \ddot{\mathbf{q}}(t) = M^{-1}(\ddot{\mathbf{q}}(t)) F(\ddot{\mathbf{q}}(t), \mathbf{q}(t), \mathbf{u}(t)) \]

The explicit equations are shown on the next page.
\[\ddot{y}(t) = \left( m_b + m_p \right) \left[ k \dot{Q}(t) + m_p l_p \sin(\phi(t)) g - m_p l_p \sin(\phi(t)) \left( m_p l_p \cos(\phi(t)) \frac{\dot{\phi}^2(t)}{2} + k_s (y_0 - y(t)) - d \dot{y}(t) - (m_b + m_p) g \right) \right] - \frac{m_p l_p^2 (m_b + m_p) (1 - \sin^2(\phi(t)))}{m_p l_p \cos(\phi(t))} \frac{\dot{\phi}(t)}{2} - \frac{k_s (y_0 - y(t))}{m_p l_p} - d \dot{y}(t) - (m_b + m_p) g \]

\[\ddot{\phi}(t) = -\frac{R}{L} \dot{Q}(t) \]

\[\ddot{Q}(t) = -\frac{V_{so}}{L} \frac{\dot{\phi}(t)}{k} \]
Appendix C

Matlab Code: Simulations of curved beam hopper

```matlab
function [trajT, traj] = motorPendulumHopper
% 1D spring hopper with moto-pendulum electro-mechanical dynamics included
% v.2 corrects state (r0-r), previously (r-r0).
% This version uses matlab's provided ode45 solver.
% For more control (and faster runtime), use the regular implicit-euler
% solver, or the explicit-euler (along with explicit EoM)

%% Tabula Rasa
clear all % clear all variables from workspace
close all % close all open figures
format long

%%
% Parameters from paper "Minimalistic Models of an Energy Efficient
% Vertical Hopping Robot

g = 9.81; % acceleration due to gravity [m/s^2]

%%%%%% Motor Parameters
 kb = 0.0199; % Torque Constant: 19.9 [mNm/A] (datasheet)
 kt = 0.0199; % kb = kt. Speed Constant: 479 rpm/V, converted to back-emf
 % constant, gives 0.0199.
 R = 0.358; % [ohm] (datasheet)
 L = 0.000070; % 0.07 [mH] (datasheet)
 Im = 3.59e-4; % 35.9 [gcm^2] (datasheet)

%%%%%% Robot parameters
 mp = 0.3; % pendulum ~ 15 [g]
 lp = 0.145; % estimated CoG
 mb = 5.76; % [Kg]
 ks = 1200; % [N/m]Spring Constant measurement from Fabien
 r0 = 0.5;
 b = 0.2; % [N s/m] estimated damping from Xiao Xiang's paper ICRA 2013

lp = mp*lp^2 + Im;

vsAmp = 3;
%parameters vector:
% p = [g kb kt R L mp lp mb ks r0 lp];

%% Initial Conditions
vs0 = vsAmp;
```

45
QDot = 0;
phi = 0;
phiDot = 0;
r = r0 - (mb+mp)/ks*g; %precompressed
rDot = 0;

% State vector
T = 0;
q0 = [QDot, phi, phiDot, r, rDot,vs0]';
traj = []; % state trajectory
trajT = 0; % time trajectory
tspan = [0,1]; % maximum time of a cycle
steps = 30; % number of hops (stance-phases) before stopping simulation

% Options for matlab's ode solvers
options = odeset('Mass',@massMatrix);
optionsStance = odeset(options, 'Events', @takeOff);
optionsFlight = odeset(options, 'Events', @touchDown);

% Simulation
for iter = 1:steps
    %% Check Flight Phase
    if (q0(4) > r0)
        [T,q] = ode45(@fFlight, tspan,q0,optionsFlight);
        q0 = q(end,:);
        a = getDer(T,q,@fFlight); % calculating accelerations
        yForce = mb*a(:,5) + mb*g; % calculating vertical force ...% pendulum on body
        yForce2 = -mp*( a(:,5) - lp*sin(q(:,2)).*a(:,3) - ...
                        lp*cos(q(:,2)).*q(:,3).^2 ) + mp*g +sin(phi)*lp*kt*q(1)); % The second calculation is simply used to verify
        q = [q(:,1:3) q(:,4:5) a(:,5) q(:,6:end) yForce yForce2]; % Saving extended state vector
        traj = [traj; [q zeros(size(q,1),1)]];
        trajT = [trajT; T+trajT(end)];
    end
    %% Stance Phase
    tStep = [tStep; T(end)];
    [T,q] = ode45(@fStance, tspan,q0,optionsStance);
    q0 = q(end,:);
    a = getDer(T,q,@fStance); % calculates state accelerations
    % calculating vertical force exerted by pendulum on body
    yForce = mb*a(:,5) + mb*g - ks*(r0-q(:,4)) + b*(q(:,5));
    yForce2 = -(mp*( a(:,5) - lp*sin(q(:,2)).*a(:,3) - ...
                          lp*cos(q(:,2)).*q(:,3).^2 ) + mp*g +sin(q(:,2))*lp*kt*q(1));
    subplot(steps,1,iter);
    plot(T,yForce,T,yForce2);
    legend('force from body', 'force from pendulum');
    title('Stance Forces');
    q = [q(:,1:3) a(:,3) q(:,4:5) a(:,5) q(:,6:end) yForce yForce2];
    traj = [traj; [q ones(size(q,1),1)]];
    trajT = [trajT; T+trajT(end)];
```matlab
display( 'q = [1:QDot, 2:phi, 3:phiDot, 4:phiDotDot, 5:r, 6:rDot, ... 

% Modifying angular position to stay between 0 and 2*pi.
q(:,2) = mod(q(:,2),2*pi);

function dqdt = fStance(t,q)
    % q = [qDot phi phiDot r rDot vs]';
    % Unboxing parameters
    vs = vsAmp;%(2*pi*t).*vsAmp;
    if(q(5) > 0 && mod(q(2),2*pi) < pi/2)
        vs = vsAmp;
    elseif (q(5) < 0 && (mod(q(2),2*pi) > pi && mod(q(2),2*pi) ... 
        <3*pi/4))
        vs = vsAmp;
    else
        vs = 0;
    end

dqdt = zeros(size(q));

dqdt(1) = -R/L*q(1) - kb/L*q(3) + vs/L;

dqdt(2) = q(3);

dqdt(3) = kt*q(1) + mp*sin(q(2))*lp*g;

dqdt(4) = q(5);

dqdt(5) = mp*lp*cos(q(2)))*q(3)^2 + ks*(r0 - q(4)) - (mp+mb)*g - ... 
        b*q(5);

dqdt(6) = 0; %constant
end

function dqdt = fFlight(t,q)
    if(q(5) > 0 && mod(q(2),2*pi) < pi/2)
        vs = vsAmp;
    elseif (q(5) < 0 && (mod(q(2),2*pi) > pi && mod(q(2),2*pi) ... 
        <3*pi/4))
        vs = vsAmp;
    else
        vs = 0;
    end

dqdt = zeros(size(q));

dqdt(1) = -R/L*q(1) - kb/L*q(3) + vs/L;

dqdt(2) = q(3);

dqdt(3) = kt*q(1) + mp*sin(q(2))*lp*g;

dqdt(4) = q(5);

dqdt(5) = mp*lp*cos(q(2)))*q(3)^2 - (mp+mb)*g;

dqdt(6) = 0;
end

function [impact,terminate,direction] = touchDown(˜,q)
    % This function detects a zero crossing, therefore the impact
    % variable needs to be zero when the touchdown condition is fulfilled.
    impact = r0 - q(4);
    % The terminate switch is used to stop the simulation. If set to zero
    % the crossing point is recorded but the integration continues.
    terminate = 1;
    % This variable indicates the direction of the zero crossing that will
```
% be detected:
%  1 -> negative -> positive
% -1 -> positive -> negative
%  0 -> both
% direction = 1;
end

function [impact, terminate, direction] = takeOff(~, q)
impact = r0 - q(4);
terminate = 1;
direction = -1;
end

function M = massMatrix(~, q)
M = [
    1  0  0  0  0  0;
    0  1  0  0  0  0;
    0  0  mp*lp^2  0  -mp*lp*sin(q(2))  0;
    0  0  0  1  0  0;
    0  0  -mp*lp*sin(q(2))  0  (mb+mp)  0;
    0  0  0  0  0  1;
];
end

function acc = getDer(T, q, fHandle)
acc = [];
for iter2 = 1:numel(T)
    mass = massMatrix(T(iter2), q(iter2,:));
    f = fHandle(T(iter2), q(iter2,:));
    acc = [acc; (mass \\ f')'];
end
end

end

data

function [xNew] = forwardEuler(x0, fHandle, h)
% forward euler solution for autonomous function
xNew = x0 + h*fHandle(0,x0);
end

function [xNew, iter] = implicitEuler(x0, fHandle, DfHandle, h, tol, maxIt)
% fHandle are the system dynamics
% DfHandle is the jacobian, d fHandle / d x
% bogus value for time (to be compatible with Matlab odes, all functions
% accept t, but don't use it)
t = 1;
xkl=x0; % first estimate for x(k+1)
% deltaX = 1;
for iter=1:maxIt
    deltaX = -DfHandle(t,xkl) \ fHandle(t,xkl);
    if(deltaX < tol)
        break
    end
    xkl = xkl + deltaX;
    if(iter == maxIt)
        display('max iteration reached');
    end
end
xNew = x0 + h*fHandle(t,x0);
end

% Write and Read to a NI USB-6008 DAQ device
clear all
close all
clc

%% Digital input enables ESCON
dio = digitalio('nidaq', 'Dev1');
hline = addline(dio, 0:11, 'out');
putvalue(dio, [8 8 8 8 8 8 8 8 8 8 8 8])

%% Initialization
ai = analoginput('nidaq', 'Dev1'); % Analog Input
ao = analogoutput('nidaq', 'Dev1'); % Analog Output
ao0 = addchannel(ao, 0); % Add desired channel for output
ai0 = addchannel(ai, [0,1]); % Add desired channel for input
% (See also NI SCB 68 PIN positions)
timelength = 5; % total motor control time (s)
freq = 3.5 % Signal frequency (Hz)
A = 3.5; % Current amplitude
rc = 0.95/10; % Current to voltage ratio. e.g. 0.95/10
% -> 0.95 Ampere if the analog output is 10V
ai_value = [0,0]; % Initialize Input value
ao_array = 1; % Initialize Output value
time_array = 0; % Initialize time
offset = 0; % offset to shift sinusoid zero crossing
tic % Start time

%% Start Control loop
disp('Start of loop')
while time_array(end)<timelength

%%%%% Define trajectory of control input (current) here:
ao_value = A*sign(sin(2*pi*freq*time_array(end)) + offset );
ao_value = max(-0.1,ao_value);
ao_array=[ao_array,ao_value*rc]; % Write current in vector
time_array=[time_array,tic]; % Write time in vector
tmp=getsample(ai); % Get motor winding 1 Voltage
ai_value = [ai_value,tmp]; % Write motor winding 1 Voltage in vector
end
ao_value = 0;
putsample(ao, ao_value)
disp('End of loop')

%% Disable ESCON
dio = digitalio('nidaq', 'Dev1');
hline = addline(dio, 0:11, 'out');
putvalue(dio, [0 0 0 0 0 0 0 0 0 0 0])
delete(ai)
delete(ao)

%% Transform Voltage
R2 = 22.2; % Voltage divider resistance 2 (kOhm)
R1 = 98; % Voltage divider resistance 1 (kOhm)
K = R2/(R1+R2); % Voltage reduction constant
VoltIn= -ai_value(:,1)/K;

%% Filter Results
n_p = 5.8;  % Filter band (percent)
av = mean(VoltIn);  % Average voltage
av_ub = av*(1 + n_p);  % Voltage upper bound
av_lb = av*(1 - n_p);  % Voltage lower bound
V_Filt = VoltIn;

for i = 2:length(VoltIn)
    if VoltIn(i) >= av_ub || VoltIn(i) <= av_lb
        V_Filt(i) = V_Filt(i-1);
    end
end

va = 15;  %Variance
V_Filt_G = V_Filt;
V_sum = 0;
G_sum = 0;
for i = 1:length(VoltIn)
    for j = 1:length(VoltIn)
        G_x = 1/(sqrt(2*pi)*va)*exp(-((j-i)^2/(2*va^2)));
        G_sum = G_x + G_sum;
        V_sum = G_x*V_Filt(j) + V_sum;
    end
    V_Filt_G(i) = V_sum/G_sum;
    V_sum = 0;
    G_sum = 0;
end

%% Plot results
% plot(time_array,VoltIn)
hold on
plot(time_array,a_o_array,'r')
plot(time_array,V_Filt,'k')
plot(time_array,V_Filt_G,'b')
legend('Current [A]','Voltage [V]');
xlabel('Time [s]')
ylabel('V/A')

%% Calculate expended energy
E_o = 0;
for i = 1:length(V_Filt_G)-1
    dt = time_array(i+1)-time_array(i);
    E_n = E_o + 2*abs(V_Filt_G(i))*abs(a_o_array(i))*dt;
    E_o = E_n;
end
E_exp = E_n
V = V_Filt_G(end)*2

%% Calculate Power
P = [];
for i = 1:length(V_Filt_G)
    P(i) = 2*abs(V_Filt_G(i))*abs(a_o_array(i));
end
P = P';