Master Thesis

CPG-inspired control of curved beam hoppers

Fall Term 2013

Supervised by:
Xiaoxiang Yu
Fumiya Iida

Author:
Christos Lataniotis
# Contents

Abstract iii

1 Preliminaries 1
  1.1 Introduction to curved beam hoppers 1
  1.2 Abstract models for vertical hopping 2
  1.3 A quick introduction to CPGs 4
  1.4 Goal Description 4
  1.5 The Hopf Oscillator 5

2 Testing the concept 9
  2.1 Hopping height Stabilization 10
  2.2 Hopping height Control 12

3 Case of Uncoupled Dynamics 16
  3.1 Hopping height Stabilization 17
  3.2 Hopping height Control 20
  3.3 Controller Implementation 22

4 Case of Coupled Dynamics 27

5 Conclusions and Outlook 31

A Equations of motion of new curved beam vertical hopper's abstract model 33

Bibliography 35
A feedback control strategy for a family of curved beam robots that are hopping vertically is explored. A specific kind of oscillator is used, the adaptive frequency Hopf oscillator, as the main component of the feedback controller. This kind of oscillator is based on Central Pattern Generators (CPGs) paradigms. Such oscillators have the property that they learn the frequency of a periodic input signal without any external optimization process fixed. A novel way to perform hopping height stabilization and hopping height control has been found, for certain conceptual models of the curved beam robots, based on the selection of proper input signals to be sent to the oscillator. Based on whether the motor that is used for actuating the curved beam robots has a high gear ratio gearbox or not, the effectiveness of the proposed feedback control strategy is investigated.
Chapter 1

Preliminairies

1.1 Introduction to curved beam hoppers

Many aspects of legged robot locomotion are being extensively investigated during the past decades. One of these aspects that still poses a great challenge is the energy efficiency of legged robot locomotion. So far, a typical energy efficient legged robot is a passive dynamic walker; a bipedal walker design (introduced in [1] ) that can achieve walking on a shallow slope without active control or any type of energy input comparable to human walking in terms of energy efficiency and that it makes for mechanical simplicity. Further extensions have been proposed, based on the passive dynamics, so that walking on level ground can also be achieved using small active power sources([2],[3]). The most important limitations of passive-dynamic walkers so far is the minimal robustness against disturbances and the limited diversity of their behaviours due to their minimal control characteristic.

Many researchers try to introduce and exploit compliant elements to the robot structure because compliant dynamics can greatly reduce energy dissipation, mainly due to ground impacts or deceleration([4],[5], [6]) and additionally improve the system’s stability ([7],[8]). However, substantial improvements in terms of energy efficiency are still not possible. One of the challenges lies in the improvement of the efficiency of vertical hopping or jumping because such behaviors play major role in locomotion.

Curved beam robots, were introduced as a possible approach for achieving energy efficient hopping behavior([9],[10],[11]). A rotating mass (energized by an electric motor) forces a compliant beam structure to vibrate. Such vibrations can induce hopping behavior. Curved beam robots are easy to manufacture, low cost and light-weight. Additionally, only one actuator is needed to induce a periodic oscillation of forces on the elastic beam as compared to conventional position control of hip joints. The dynamics of such robots have not yet been fully clarified, although some abstract models have already been proposed and will be discussed in the following section([12]).
1.2 Abstract models for vertical hopping

As proposed in [12] three minimalistic models could be of use in order to describe the hopping behaviour of a curved beam vertical hopper. For this type of robot a reasonable assumption is that most of the mass is located on the upper body, where the payload is (Figure 1.1). In that sense, probably, the most abstract model of the robot could be the single spring-mass or SLIP (Spring Loaded Inverted Pendulum) with damping. However, using such model the ground impacts are ideal which makes it hard to use such abstraction to fully capture the robot’s behaviour. The vertical force that is applied to the mass is the vertical component of the centrifugal force due to the mass rotation.

Based on the motor that we use the dynamics and, as a result, the robot’s behaviour varies depending on whether a gearbox is used or not. When a (high gear ratio) gearbox is used the body’s vibrations do not have effect on the dynamics of the rotating mass as opposed to the case of not having a gearbox (or using a gearbox with low gear ratio).

Figure 1.2: Abstract models for vertical hopping. From left to right: single, double and triple mass model.

Keeping that distinction in mind, the double mass model is expected to better
describe the case of using a high gear ratio gearbox. This case will also be refered to as case of *uncoupled dynamics*, in order to emphasize the fact that although the rotating mass’s dynamics affect the behavior of the curved beam the opposite is not true. As in the simple SLIP case the force that is applied on \( m_2 \) is the vertical component of the centrifugal force due to the mass rotation. Additionally the mass \( m_1 \) is the base mass; the existence of the base mass helps up model different scenarios for ground impact. In the current thesis the case of fully plastic impacts is assumed, i.e. the entire velocity of the bottom mass is becoming zero on touchdown.

The case when the gear ratio of the motor’s gearbox is small (or no gearbox exists) will be refered to as case of *coupled dynamics*. This naming is used because the dynamics of the rotating mass are coupled with the dynamics of the curved beam.

The triple mass model was proposed in that case as a simplistic model that can capture that behaviour. The idea of considering the rotating mass as an additional spring mass on top of the rest of the system is based on the observation that is shown in Figure 1.3.

Figure 1.3: An explanatory schematic for the equivalence between rotating mass and spring mass under certain conditions. Courtesy of [12]

The centrifugal force, \( F_c \) applied to \( m_3 \) is

\[
F_c = m_3\omega^2r 
\]

thus the vertical component is

\[
F_{cy} = m_3\omega^2r\sin\phi \tag{1.1}
\]

If the vertical position of the mass (since we are interested in vertical hopping this one dimensional assumption is intuitive) is treated as if the mass was connected to a spring then the spring force, \( F_{sp} \), would be

\[
F_{sp} = k\sin\phi \tag{1.2}
\]

since the vertical displacement of the mass (with respect to \( m_2 \) is equal to \( r\sin\phi \).

By using equations 1.1, 1.2 we can quickly conclude that equivalence is true between the two models when

\[
k = m_3\omega^2 \tag{1.3}
\]

In other words, in equation 1.3 the fact that a spring of specific spring constant can give the same response of the vertical displacement of the rotating mass as in the standard case where the mass is actually rotating. It should be noted however, that in the general case the spring would be non-linear since the spring constant is a function of the angular velocity, \( \omega \).
1.3 A quick introduction to CPGs

Central Pattern Generators are neural networks that can endogenously (i.e. without rhythmic sensory on central input) produce rhythmic patterned outputs. The first modern evidence that rhythmic motor patterns are centrally generated in animals, was the demonstration that the locust nervous system, when isolated from the animal, could produce rhythmic output resembling that observed during flight ([13]). Subsequent work (e.g. [14]) showed that, in a wide variety of invertebrate and vertebrate animals, nervous systems isolated from sensory feedback, could produce rhythmic outputs resembling those observed during rhythmic motor pattern production. This work has further shown that rhythmic pattern generation does not depend on the nervous system acting as a whole, but that CPGs are instead relatively small and autonomous neural networks.

Interesting properties of CPGs like distributed control, ability to deal with redundancies, fast control loops and allowing modulation of motor control by simple control signals make them interesting building blocks for locomotion controllers in robots. Various types of mathematical models have been proposed in order to study biological CPGs, as well as the types of animal locomotion that have been modelled (An overview can be found in [17]). Generally speaking, depending on the phenomena under study, CPG models have been designed at several levels of abstraction

- **Detailed biological models** (outside the scope of this thesis).
- **Connectionist models.** Simplified neuron models are used, such as leaky-integrator neurons or integrate-and-fire neurons.
- **Oscillator models.** They are based on mathematical models of coupled nonlinear oscillators to study population dynamics. In this case an oscillator represents the activity of a complete oscillatory center (instead of a single neuron or a small circuit).

Oscillator models, traditionally, are not used in order to explain rhythmogenesis but to study how inter-oscillator couplings and differences of intrinsic frequencies affect the synchronization and the phase lags within a population of oscillatory centers. The motivation for this type of modelling comes from the fact that the dynamics of populations of oscillatory centers depends mainly on the type and topology of couplings rather than on the local mechanism of rhythm generation, something that is well established in dynamical systems theory. For example, locomotion based on different types of topology and couplings of oscillators different gait patterns can be produced with some nice features that CPGs can offered as mentioned in the previous paragraph ([15], [16]).

In this thesis the use of a specific type of oscillator model is going to be used, namely the Hopf Oscillator in a different sense than usually, motivated by the work in [18]. Instead of studying the coupling between oscillators the coupling between an oscillator and a robot is assumed. One main distinction in this case is that the coupling between the oscillator and robot cannot be considered weak. More details about this will be given after the oscillator of choice will be discussed.

1.4 Goal Description

The main idea that is going to be investigated in the current work is whether a controller based on some concept related to Central Pattern Generators, namely an Adaptive Hopf Oscillator, can be employed. An acceptable controller should have the following features
• Stabilize the controller-robot system. For our case this is interpreted as stabilizing the hopping height.

• Control the robot system. For our case this is interpreted as control of the hopping height.

The general framework of the approach is illustrated in Figure 1.4. Some abstract models have already been proposed for the platform of choice and they are going to be used in order to experiment conceptually on this approach. The ultimate goal would be to implement such controller on the actual platform.

For now we focus on the model; what we are looking for is first, a proper way to interconnect the model and the oscillator and then observe whether/when such pair of controller-robot can give satisfactory results based on the criteria defined above. As a starting point, the simple SLIP model is going to be used in order to test this concept and then we are going to build up on more complex ones. The output of the oscillator is supposed to be directly the actuation of the system. So for example, in the SLIP case the oscillator’s output is the vertical force that is applied on the mass, $F(t)$. The sensory input that the oscillator needs in order to produce this force is not known. Next, the Hopf oscillator will be discussed because of the importance of some remarks that have to be made before moving on towards the goal.

### 1.5 The Hopf Oscillator

The dynamics of the Hopf oscillator is governed by the following differential equations, in polar coordinates:

1. $\dot{r} = \gamma (\mu - r^2) r 
2. \dot{\phi} = \omega 

The system’s output, $y$ is

$$ y = r \cos \phi $$

The property of interest for such system is the fact that it will always eventually produce a sinusoidal output (Figure 1.5), with amplitude $\sqrt{\mu}$ and frequency $\omega$ irrespective of the initial conditions. The speed of the convergence is governed by the parameter $\gamma$. Such system can become more interesting if some sort of sensory input
Figure 1.5: An example of the output of a Hopf oscillator.

is introduced in order to make its frequency adaptive. In order to do so, equations 1.4, 1.5 are rewritten as follows

\[ \dot{r} = \gamma (\mu - r^2) r + P_{\text{teach}} \cos \phi \]  
(1.7)

\[ \dot{\phi} = \omega - \frac{1}{r} P_{\text{teach}} \sin \phi \]  
(1.8)

and additional dynamics are introduced to the frequency, \( \omega \) (adaptation rule):

\[ \dot{\omega} = -\epsilon P_{\text{teach}} \sin \phi \]  
(1.9)

The system’s output remains as in equation 1.6. Now a teaching signal, \( P_{\text{teach}} \), is fed into the oscillator. The additional parameter \( \epsilon \) determines the effect of the teaching signal to the oscillator’s dynamics and is usually referred to as learning rate. Simple example is going to be discussed in order to briefly explain how this system is supposed to work.

Figure 1.6: An example of the output of an Adaptive Hopf oscillator, learning a simple sinusoidal signal.

Assume that the teaching signal simply a sinusoid, for example \( P_{\text{teach}} = 5 \sin(15t) \). Additionally assume that the initial value of \( \omega \) is \( \omega_0 = 0 \). In Figure 1.6 the output of the Hopf oscillator and frequency \( \omega \) over time is illustrated. After a few seconds Hopf output has synchronised with the teaching signal in terms of frequency (and phase). Notice that in such approach the amplitude does not adapt to the input; although simple extention of the current oscillator can give such behavior ([21]).
More generally it is known that when the oscillator has its intrinsic frequency $\omega$, close to one frequency component of a periodic input, in will phase-lock (also known as entrainment)\[19\]. The maximum distance between the intrinsic frequency of the oscillator and the periodic input that still permits phase-locking depends on the coupling strength $\epsilon$. The stronger the coupling, the larger the entrainment basin. Outside this basin the oscillator is still influenced by the coupling but does not synchronize.

If a periodic input has more than one frequency components, then several entrainment basins will appear. Phase-locking will be possible with each frequency component if inside the respective entrainment basin and outside the basins the oscillator will have the tendency to accelerate or decelerate (according to the term $P_{\text{teach}} \sin \phi$).

In order to understand why such learning takes place in an Adaptive Hopf Oscillator, some basics regarding dynamic Hebbian learning in frequency oscillators are going to be discussed ([20]). Notice that such behavior is not true only in case of a Hopf oscillator but in a family of frequency oscillators that have the general form in cartesian coordinates:

$$\dot{x} = f(x, y, \omega) + \epsilon P_{\text{teach}}$$
$$\dot{y} = g(x, y, \omega)$$

For such type of perturbation from an external signal the same equations in polar form would be written as (as an abbreviation $f$ is used instead of $f(x, y, \omega)$ and $g$ instead of $g(x, y, \omega)$) \[\text{1}\]

$$\dot{r} = f \cos \phi + g \sin \phi + \epsilon P_{\text{teach}} \cos \phi$$
$$\dot{\phi} = -\frac{1}{r} [f \sin \phi - g \cos \phi] - \frac{\epsilon}{r} P_{\text{teach}} \sin \phi$$

In order to understand how to come up with an adaptation rule (like the one in 1.9) some analysis has to be made about the effects of perturbations on a limit cycle system, i.e. an oscillator ([20]). Due to the stability properties of a limit cycle system a perturbation can in the long term only affect the phase of the oscillator. Take as an example equations 1.4, 1.5 where it is clear that perturbations to the radius will finally fade away but perturbations to the phase will remain unchanged. Especially in a small neighborhood of the limit cycle a small perturbation can only affect the phase strongly if it perturbs the oscillator in the direction tangential to the limit cycle (Figure 1.7). By introducing a coordinate system with its origin on the phase point, the first base vector $\vec{e}_r$ is chosen perpendicular to the limit cycle and the second base vector $\vec{e}_\phi$ tangential to the limit cycle. Thus this coordinate system rotates with the phase point along the limit cycle. Next we introduce a perturbation $\vec{P}$. The influence of the perturbation to the phase will be the projection of $\vec{P}$ to $\vec{e}_\phi$, thus:

$$p_{\phi} = \vec{P} \cdot \vec{e}_\phi$$

\[\text{1}\] It is straightforward to obtain these equations by using the relations between polar and cartesian coordinates:

$$x = r \cos \phi$$
$$y = r \sin \phi$$

For the sake of completeness the functions $f, g$ for a Hopf oscillator are given:

$$f(x, y, \omega) = (\mu - r^2) x - \omega y$$
$$g(x, y, \omega) = (\mu - r^2) y + \omega x$$

where $r = \sqrt{x^2 + y^2}$. 
From equation 1.12 it can be seen that depending on the external perturbation and the state of the oscillator the perturbation either accelerates or decelerates the phase point. If the perturbation is a periodic signal it would result in an average acceleration or deceleration, depending on the frequency difference. In other words, the influence of the perturbation to the phase carries the information needed to adjust to the frequency of the external perturbation. Therefore, if we take this same effect to tune the frequency of the oscillator (on a slower time scale) the frequency should evolve towards the frequency of the perturbation! So, if we have an adaptation rule of the form

$$\dot{\omega} = h(\omega, r, \phi, P_{teach})$$

the effect of $h(\omega, r, \phi, P_{teach})$ on $\omega$ has to be the same as the effect of the perturbation on the phase, thus on average driving $\omega$ towards the frequency of the perturbation.

In case of oscillators of the general form, given in 1.10, 1.11, it is clear that the effect of the perturbation of the teaching signal to the phase is

$$p\phi = -\frac{\epsilon}{r}P_{teach}\sin\phi$$  \hspace{1cm} (1.13)

thus we choose the adaptation rule of equation 1.9.

It was shown that by using a certain adaptation rule for $\omega$ the oscillator will eventually adapt to the frequency of a periodic input signal. However, if the teaching signal has more than one frequency components further analysis must be made in order to see why different basins of attraction are formed for each of the harmonics (for details refer to [20]). One important finding is that the amplitudes of each frequency component of a periodic input signal will influence the convergence of the oscillator, in the sense that the more intensity a frequency component has, the more it will attract $\omega$ over time. It must also be noted that the zero frequency, i.e. the mean of the periodic signal, can also influence the convergence because of its amplitude in the frequency spectrum; if it has a stronger influence than the other frequency components. In other words, if the teaching signal has a non-zero mean, the oscillator’s frequency can eventually converge to 0.
Chapter 2

Testing the concept

The concept of feedback control of a damped vertical hopper using adaptive Hopf oscillator will be discussed. The physical system’s dynamics are well-known. It is a simple hybrid dynamical system with two phases, the fight phase described by

\[ \ddot{x} = -g + F \quad (2.1) \]

and the stance phase described by

\[ \ddot{x} = -\frac{1}{m} [-c\dot{x} - k(x - L_0) + F] \quad (2.2) \]

Touch down (transition from flight to stance phase) occurs when \( x = L_0 \) and take off (transition from stance to flight phase) when \( x = L_0 \) and \( \dot{x} > 0 \). It is assumed that the force \( F \) is acting in both phases; this assumption is motivated by the fact that the rotating mass is constantly applying force to the rest of the robot during both phases. Additionally \( F \) is considered to be periodic of the form:

\[ F(t) = ar(t)\cos\phi(t) \quad (2.3) \]

where \( r(t) \) is the radius, \( \phi(t) \) is the phase of the oscillator and \( a \) a fixed amplitude. The parameter set that was used for the simulation results that will be shown next can be found in Table 2.1.

Next the solution that was found for hopping height stabilization will be discussed, followed by the solution for hopping height control.
Chapter 2. Testing the concept

Table 2.1: The parameter set for the simulation of the single mass vertical hopper.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>0.3 kg</td>
</tr>
<tr>
<td>$k$</td>
<td>300 N/m</td>
</tr>
<tr>
<td>$L_0$</td>
<td>0.5 m</td>
</tr>
<tr>
<td>$c$</td>
<td>0.3 Ns/m</td>
</tr>
<tr>
<td>$a$</td>
<td>0.5</td>
</tr>
<tr>
<td>$x(0)$</td>
<td>0.7 m</td>
</tr>
<tr>
<td>$\dot{x}(0)$</td>
<td>0 m/sec</td>
</tr>
</tbody>
</table>

2.1 Hopping height Stabilization

After testing of different scenarios of sensory signals to use as teaching signals for the oscillator, asymptotically stable solutions for the hopping height could be obtained when the velocity of the mass was used ($\dot{x}$). Interestingly the actual value of the velocity seemed not necessary since identical results was possible to obtain by just using

$$P_{teach} = sgn(\dot{x})$$  \hspace{1cm} (2.4)

Some typical simulation results are shown in Figure 2.2.

Figure 2.2: Hopping height response when $P_{teach} = sgn(\dot{x})$. On top the response for the first 50 seconds is shown and on bottom the response for 250 seconds.

Initially the force is not synchronised with the hopper and as a result the hopping
height gradually decreases (Figure 2.3). This is also evident by noticing that in many time intervals the power consumption is negative which means that the force is acting against the movement of the mass. After some time where the oscillator and the mass have synchronised the behavior radically changes and the mass is hopping continuously to larger heights asymptotically reaching the maximum possible height that can be achieved from the current force amplitude. This synchronization can be seen in Figure 2.4. There we can also notice that the Power always remains positive which means that the force never acts against the natural movement of the hopper.

![Figure 2.3: Top: Hopping height and force over time for the first few seconds of the run. Bottom: Power over time for the first few seconds of the run.](image)

The behavior of the frequency of the oscillator over time (Figure 2.5) can be intuitively interpreted. Initially, since the force was actually pushing against the natural movement of the hopper and the hopping height was gradually decreasing the frequency of changes of the sign of the velocity was continuously increasing until the point when the force starts synchronising properly with the hopper. Then since the hopping height gradually increases the frequency of changes of the sign of the velocity decreases and this is also reflected on the frequency of the oscillator. The nice feature this kind feedback has is that the frequency of the force can adapt over time based on the height that the mass hops.

The problem with this kind of feedback is that although hopping height seems to be able to get asymptotically stabilized, controlling it is not easy since small changes to the force amplitude lead to large changes on the resulting height. Additionally, this conceptual test gave us the insight that choosing an appropriate teaching signal is of utter importance.
2.2 Hopping height Control

Being energy efficient, i.e. keeping the power always positive, is in general a good thing but if we would like to actively control the hopping height this cannot always be true, since sooner or later we would have to pump energy out of the system in order to stop it from getting values higher than the desired one. This kind of reasoning can be incorporated into an altered version of a teaching signal, but based on the one in 2.4.

We know that a teaching signal like $\text{sgn}(\dot{x})$, when learned, will result to a force that always follows the motion of the mass. This is something desirable up to the point when the height has reached the desired apex height. Afterwards pumping energy out of the system is inevitable in order to maintain the desired value. First define the system’s total energy, $E(t)$, as

$$E(t) = \frac{1}{2} m \dot{x}^2 + \frac{1}{2} k (x - L_0)^2 + mg x^2$$

(2.5)
where the term $\frac{1}{2}k(x - L_0)^2$ is only present in stance phase. Given a desired apex height $h^\text{Apex}_d$ the corresponding desired energy, $E_d$, that we want the system to reach is

$$E_d = mgx^\text{Apex}_d$$  \hspace{1cm} (2.6)

Notice that with this model there is a one-to-one mapping between apex height and energy; this is a nice feature but in general not true in more complex models. In order to reflect the observations that we made to the teaching signal, redefine it as follows:

$$P_{\text{teach}}(t) = \tanh \left[ A \left( \frac{E(t)}{E_d} - 1 \right) \right] \text{sgn}(\dot{x})$$  \hspace{1cm} (2.7)

This new version of the teaching signal has the same effect as before when $E_d < E(t)$. When the opposite is true, $E_d > E(t)$, the teaching signal is guiding the force to work fully against the mass’s movement. This is something not desirable for relatively small deviations of $E(t)$ around $E_d$. That is the reason why a smooth sigmoidal function like the hyperbolic tangent is used. Parameter $A$, is optional and can be used in order to tune how sharp or wide the ‘S’ of the sigmoidal function will be.

Some simulation results using such teaching signal are shown next. Using such teaching signal a desired apex height value can be maintained (Figure 2.7). In Figure 2.8 we can see the transition from $h^\text{Apex}_d = 0.8 \text{ m}$ to $h^\text{Apex}_d = 0.75 \text{ m}$. The force selectively desynchronises when the current energy level gets higher than the desired one. It should be noticed that this entire range of apex heights was achieved using a fixed amplitude of the force. In order to achieve smooth transitions when the desired apex height is reduced small gradual decreases usually work better; otherwise the force can get completely desynchronised with the mass and a behavior similar to the one in Figure 2.3 can be observed, i.e. the hopper can get pushed to do very small hops until synchronisation takes place and the desired value can then be reached.

Convergence of the oscillator’s frequency is satisfactory and is tightly related to the learning rate, $\epsilon$ that is used (recall equations 1.7 - 1.9). Although a large range of
values of \( \epsilon \) can give acceptable results it was observed that very small values (much smaller than 1) can result in poor synchronisation and very large values can result in higher fluctuations around the point of convergence. Lastly the teaching signal is shown in Figure 2.10. We can notice that the bigger the difference between the current energy level and the desired one the higher the amplitude of the teaching signal is.
2.2. Hopping height Control

Figure 2.9: The oscillator’s $\omega$ response during hopping height control.

Figure 2.10: The teaching signal that is sent to the oscillator during hopping height control. The plot around $t = 40\text{sec}$ when $h_d^{\text{prez}}$ changes is zoomed.
Chapter 3

Case of Uncoupled Dynamics

Figure 3.1: A schematic of the interconnection between the double mass model and an adaptive Hopf oscillator.

After making the conceptual system of interconnected single mass hopper with an adaptive Hopf Oscillator working we now move on to a model that is closely related to a curved beam that is using a motor with a gearbox and high gear ratio. This is a hybrid system with different dynamics during flight and stance phase. The equations of motion during flight phase are

\[
\ddot{x}_1 = \frac{1}{m_1} [-m_1 g + k (x_2 - x_1 - L_0) + c (\dot{x}_2 - \dot{x}_1)] \\
\ddot{x}_2 = \frac{1}{m_2} [-m_2 g - k (x_2 - x_1 - L_0) - c (\dot{x}_2 - \dot{x}_1) + F]
\]

and during stance phase

\[
\ddot{x}_1 = 0
\]

the equation of motion for \(x_2\) remains the same during both phases. The ground impacts are considered plastic, i.e. whenever \(m_1\) touches the ground \(\dot{x}_1\) becomes zero. The transition from stance to flight phase (take off) occurs when the spring force becomes higher than the weight of the bottom mass, i.e. \(k (x_2 - x_1 - L_0) > m_1 g\) and the transition from flight to stance phase (touch down) occurs when \(x_1 = 0\). The dynamics of the rotating mass can be neglected. The connection between \(F\) and the vertical component of the centrifugal force because of the rotation of the mass is straightforward to obtain (recall Equation 1.1, Figure 3.2):

\[
F_{cy} = m_3 \omega^2 r \sin \beta
\]
3.1 Hopping height Stabilization

In order to avoid confusion of the rotating mass’s angle and the phase of the Hopf oscillator the rotating mass angle is denoted by \( \beta \). The main difference of this model compared to the previous one is the addition of the foot mass which affects the entire dynamics but additionally introduces ground impacts (energy loss). The parameter set that was used during simulations is given in Table 3.1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m_1 )</td>
<td>0.1 kg</td>
</tr>
<tr>
<td>( m_2 )</td>
<td>0.55 kg</td>
</tr>
<tr>
<td>( k )</td>
<td>500 N/m</td>
</tr>
<tr>
<td>( L_0 )</td>
<td>0.5 m</td>
</tr>
<tr>
<td>( c )</td>
<td>0.1 Ns/m</td>
</tr>
<tr>
<td>( x_1(0) )</td>
<td>0 m</td>
</tr>
<tr>
<td>( x_2(0) )</td>
<td>0.7 m</td>
</tr>
<tr>
<td>( \dot{x}_1(0) )</td>
<td>0 m/sec</td>
</tr>
<tr>
<td>( \dot{x}_2(0) )</td>
<td>0.5 m/sec</td>
</tr>
</tbody>
</table>

Table 3.1: The parameter set for the simulation of the double mass vertical hopper.

3.1 Hopping height Stabilization

As in the case of the single mass hopper, the teaching signal

\[
P_{\text{teach}} = sgn(\dot{x}_2) \tag{3.4}
\]

was able to produce stable hopping behavior. Some typical simulation results are shown in Figure 3.3. In the example shown here the force amplitude is relatively small (\( a = 0.5 \)) and as a result the apex height is small, which leads to convergence of \( \omega \) in relatively high value (Figure 3.4). However it is interesting to notice that the system finally converged to a stable synchronous behaviour even though the initial value of the oscillator’s frequency was far off.

Figure 3.2: The centrifugal force due to the rotating mass.
While the force amplitude varies, the stable hopping height that is achieved varies in a different way than the case of the single mass hopper (Figure 3.5). Interestingly the synchronisation modes do not change and as an example in Figure 3.6 the synchronisation modes for two amplitude values that give very different apex heights are compared. Although in case of $a = 1.4$ the small oscillations of the bottom mass are higher (this is normal due to higher hopping height) the overall synchronisation of the masses $m_1$ and $m_2$ has a very similar trend in both cases. The important piece of information that one can obtain from this observation is that each value force amplitude is likely to be able to produce apex heights around a region, tighter than the case of the single mass hopper.
3.1. Hopping height Stabilization

Figure 3.5: The stable hopping height that is achieved, using $P_{teach} = \text{sgn}(\dot{x}_2)$ for different values of force amplitude $a$.

Figure 3.6: Comparison of synchronisation modes between $x_1(t)$ and $x_2(t)$ for two amplitude values that give very different stable hopping height value.
3.2 Hopping height Control

In order to extend our approach to control the hopping height we cannot follow the procedure that was described in section 2.2, simply because the one-to-one mapping between desired apex height and system total energy is not true now. A workaround for this problem is to define the function $x_{Apex}^2(t)$ as shown in Figure 3.7.

![Figure 3.7: Representation of the function $x_{Apex}^2(t)$ with respect to a response $x(t)$.](image)

This function performs a zero order hold for each new apex value of response $x(t)$. Then the teaching signal that we are going to use is given by

$$P_{teach}(t) = \tanh \left[ A \left( \frac{x_{Apex}^2(t)}{x_{Apex}^2_{2d}} - 1 \right) \right] \text{sgn}(\dot{x}_2) \quad (3.5)$$

where $x_{Apex}^2_{2d}$ is the desired apex value of $m_2$. In order to illustrate the effectiveness of such choice of teaching signal some simulation results can be seen in Figure 3.8. One possible explanation about the reason why the apex height is stabilized slower, compared to the single mass hopper, is because the update on the teaching signal takes place once in each hop whereas in the single mass hopper the system’s energy is updated in each time step. Additionally we can see the convergence of the oscillator frequency in Figure 3.9 and the teaching singal that produced this behavior in Figure 3.10.

![Figure 3.8: Tracking of desired hopping height (of $m_2$) using the new teaching signal.](image)
3.2. Hopping height Control

Figure 3.9: The oscillator’s \( \omega \) response during hopping height control.

Figure 3.10: The teaching signal that is sent to the oscillator during hopping height control. The plot around \( t = 40sec \) when \( h_0^{Apex} \) changes is zoomed.
3.3 Controller Implementation

Before proceeding to the presentation of the proposed controller, the procedure that can translate the output of the oscillator to motor controller signals is going to be discussed. Recall that the output of the hopf oscillator is the force that has to be applied on the top mass, given by

\[ F(t) = ar(t)\sin(\phi(t)) \]

where \( r(t), \phi(t) \) are state variables of the oscillator. Also recall that the vertical component of the centrifugal force that is produced by the rotating mass is

\[ F_{cy} = m_3\omega^2R\sin(\beta) \]

Figure 3.11: Force produced by the oscillator (left) and centrifugal force \( F_c \) and its vertical component \( F_{cy} \) (right).

In order to achieve \( F(t) = F_{cy}(t) \) follows that

\[ \beta_d(t) = \phi(t) \quad (3.6) \]

\[ \dot{\beta}_d(t) = \sqrt{\frac{ar(t)}{m_3R}} \quad (3.7) \]

Therefore on each time step the output of the oscillator can be translated to desired angle and angular velocity of the rotating mass. The structure of the entire control structure is shown in Figure 3.12.

Using Figure 3.12 as a reference, the controller is going to be explained. The adaptive Hopf oscillator receives \( f_{out} \) as teaching signal. When our goal is to stabilize the hopping height then

\[ f_{out} = \text{sgn}(\dot{x}_2) \quad (3.8) \]

In that case velocity information or just the sign of the velocity of the top mass of the curved beam is necessary. When our goal is to control the hopping height then

\[ f_{out} = \tanh \left[ A \left( \frac{x^{Apex}}{x^{Apex}_2(t)} - 1 \right) \right] \text{sgn}(\dot{x}_2) \quad (3.9) \]

and we additionally need sensory information of the height of the top mass as well as some desired value of apex height \( x^{Apex}_{2d} \). The output of the Hopf oscillator are
Figure 3.12: The controller structure for hopping height stabilization and/or control of a curved beam hopper with a high gear ratio motor. The variables in red should be used for height control but can be neglected for height stabilization.

desired angle and angular velocity values (defined in equations 3.6, 3.7). Then a position and a velocity feedback controller is used in order to produce the appropriate torque value that the motor has to give. Notice that since DC motors are used there is a one-to-one mapping between torque and current of the motor governed by a constant usually called torque constant, which is given by the motor manufacturer.

What needs to be discussed then is how to obtain appropriate position and velocity controllers $G_{Cp}, G_{Cv}$. For simplicity assume that only velocity control is going to be implemented although a similar procedure can be followed for position control. In order to create, as much as possible, an automated procedure to obtain the desired controller the relation between the motor torque and the resulting velocity is treated as unknown, a black box. Then a standard black box system identification procedure is going to be followed in order to obtain an estimate of the transfer function that relates the motor torque with the angular velocity of the rotating mass. This procedure was implemented on a curved beam robot having a Maxon DC motor mounted (with no gearbox). The motor was connected to a Maxon EPOS Controller and the controller was connected to a PC running a Labview application via USB port. Each identification step will be accompanied with experimental results that were obtained from the experimental setup.

- **Step 1: Apply PRBS excitation signal.** Pseudo-Random Binary Sequences are a typical choice for inputs in black box identification and have properties similar to white noise. One noteworthy advantage that they offer is that they carry a rich frequency spectrum. A typical PRBS plot is shown in Figure 3.13. For more information refer to [22]. $N$ periods of the same PBRBS signal are going to be applied. In the experimental platform $N = 5$ periods were applied (desired motor current). The minimum and maximum values of the sequence are parameters (usually minimum current values between 0 - 100 mA and maximum current values between 600-800 mA would give sufficient response, i.e. the motor would keep rotating with varying velocity and not stop). Since this current sequence is random when the motor initial velocity is zero it is hard to start rotating and for that reason a constant current for about a second is applied first in order to make the motor start rotating (The amplitude and time of the startup current are also parameters). Notice in Figure 3.14 that instead of using the crisp desired current values the actual
motor current is used instead (through the EPOS controller one can only set the desired current that the motor has to reach).

Figure 3.13: A Pseudo-Random Binary Sequence.

- **Step 2: Obtain system response.** In our case the velocity on the motor shaft (measured by the motor’s encoder) is the system’s output. In Figure

Figure 3.14: The excitation signal (motor current) on top and the resulting system response (angular velocity) on bottom.

- **Step 3: Reduce signal noise.**
  In this step we cut out the first period of the input signal (motor current) and the output signal (motor velocity) in order to leave out transient effects. Then the remaining periods are averaged out in order to reduce the noise of our measured inputs and outputs. The resulting trimmed and averaged signals are shown in Figure 3.15.

- **Step 4: Obtain the transfer function estimate.** By using the trimmed and averaged input and output signals from Step 3, a discrete transfer function estimate can be obtained of the form:

\[
G(z) = \frac{B(z)}{A(z)}
\]

(3.10)

where \(A(z)\), \(B(z)\) are polynomials. The degree of each polynomial is a parameter and the user can change their values and re-calculate the estimate of \(G(z)\)
3.3. Controller Implementation

Figure 3.15: The input (motor current) and output (motor velocity) signals after performing trimming and averaging.

with a click of a button. Additionally the initial estimates of the coefficients can be manually selected although random values are assigned by default. The user interface and an example of the system response that can be produced using the transfer function estimate compared to the actual response is shown in Figure 3.16. Details about

Figure 3.16: The user interface that is used to calculate a transfer function estimate. On the right part a plot is shown with the actual and the estimated response of the system.

An optional functionality that was added to the application was the ability to identify so-called Hammerstein or Wiener systems. This means that either before or after the linear transfer function a static (not a function of time but only a function of the input) non-linear function can be used. When this function is placed before the transfer function the system is called Hammerstein and when it is placed after the linear transfer function the system is called Wiener. When such static non-linear functions exist both before and
after the linear transfer function the system is called Hammerstein-Wiener. Although certain recipes for when to use what kind of system do not exist, each variation has been proven useful in a wide area of applications. The static non-linear function can be any invertible function. Here polynomials (of even order) are used. When the polynomial degree is 1 the problem is reduced to estimating the linear transfer function only. Such approach might be proven useful if very heavy rotating masses are used. More details about Hammerstein-Wiener systems are omitted from this thesis but the interested reader can find a rich literature about the subject, e.g. in [23].

After an estimate $G(z)$ has been obtained, finding an appropriate controller is straightforward. A quick and easy way is to use Matlab PID tool. During the tests a PID controller with feedforward was chosen. After calculating the controller’s transfer function $G_C$, it can be easily used in Labview in order to implement the controller. In Figure 3.17 an example of velocity control is shown using a controller that was calculated following this procedure.

![Figure 3.17: Velocity control experimental results after following the identification and controller tuning procedure that is described in the text. Note that the sampling frequency of the controller is relatively low (approximately 17 Hz).](image)

The main advantage of using this approach is that after any change on the robot (masses, motor, etc.) the new controller can be calculated very quickly since the steps that are needed in order to obtain $G(z)$ are automated. On the other hand the main limitation of this setup is the small frequency that makes the controller performance poorer especially when heavy rotating masses are used (the sampling frequency can only go approximately up to $17 \, Hz$).
Chapter 4

Case of Coupled Dynamics

In this case although the approach of adaptive Hopf oscillator for feedback control did not perform as expected, certain points are going to be discussed to motivate further study on this case.

When the rotating mass is rotating with an almost constant angular velocity, as was discussed in section 1.2, a triple mass model can capture effectively the curved beam’s behavior. However such model is troublesome to use in our case since it is quite difficult to map the force that is produced by the output of the Hopf oscillator to the value of torque that the motor has to give. A different model is proposed here, for studying the dynamics of curved beam hoppers with no gearbox or low gear ratio on the motor (Figure 4.1). The rotating mass is modelled as is and the motor torque is directly used as actuation to the system. For derivation and equations of motion refer to Appendix A.

![Figure 4.1: A new model proposed for studying the dynamics of curved beam hoppers with no gearbox or low gear ratio on the motor.](image)

Although, it is difficult to make a statement about whether such control approach always fails or not (since it is heuristic, i.e. each time you try a teaching signal and observe the system’s behaviour) typical sensory readings were tried (positions, velocities and accelerations of masses, touch sensor) and no periodic or stable behavior could be obtained. Some typical results are shown in Figure 4.2

After many trials, the pre-existing experimental results were re-evaluated. The pre-
existing experimental setup (Figure 4.3) was used to control the robot in open loop by tuning the supply voltage $V_{sup}$. A known resistor is connected in series with the motor and its voltage is fed as an input to an Arduino microcontroller. Then the measurements that Arduino obtains are sent via RS-232 protocol to a PC running a Matlab application that obtains, visualises the data in real time and stores them. It is straightforward to see that since the resistor and the supply voltage are known, as well as the mapping of voltage to arduino reading, we can directly calculate the motor current.

![Diagram](image)

Figure 4.3: The pre-existing experimental setup that was used used to obtain motor current/torque values while the robot was controlled in open loop fashion by tuning the supply voltage $V_{sup}$.

Some representative pre-existing results are shown in Figure 4.4. A constant supply voltage is applied and then by applying an external push to the rotating mass it starts rotating and some stable hopping behavior can be observed for a few seconds. While this stable hopping behavior takes place the motor current measurements look quite like a sinusoidal. This observation motivated the usage of sinusoidal pattern on the motor current and even better an adaptive sinusoidal, what a Hopf oscillator actually produces.

While experimenting with this setup it was noticed that the real-time motor current
readings that were obtained via Matlab were actually not realtime but there was a delay that could become larger than 10 seconds. This can be easily observed by changing the supply voltage and waiting to see the changes to the motor current. Two issues were particularly noticed:

1. The Arduino microprocessor was continuously sending the sensory readings to the serial port. The application on Matlab was obtaining data from the serial port in a much slower rate and as a result the readings were far from realtime.

2. The measurements that were received in Matlab were treated as realtime and for each voltage value that the Matlab application was obtaining from the serial port a time stamp was attached with the current local time in the Matlab application.

In order to overcome these issues a simple messaging protocol was implemented between the Matlab application and the Arduino microcontroller. The main idea was to stop Arduino from flooding the serial port with sensory readings and only send the current sensory reading when the Matlab application requests it. In order to roughly control the sampling frequency of the readings timed loops were implemented in the Matlab application (every 10ms a new sample is obtained; in practice the actual time is expected to be a few miliseconds higher due to the communication delay). After making these changes real time data acquisition was observed and typical motor current measurements with the same scenario as before are shown in Figure 4.5.

Before making the changes the sampling frequency was actually unknown while now we can safely go up to 100 Hz (this could increase for example by experimenting with the transmission rate of the serial port). By assuming that after the changes the requested sampling frequency is actually achieved, a possible explanation for the sunisoidal-like motor current profile that was obtained before the changes would be that before the motor current looked like a sunisoid due to its high resolution. If our assumption is true then the sampling frequency before the changes was roughly \(1 \text{kHz}\). When the motor current profile is observed for a larger time scale (a few seconds) we can see that the sunisoidal profile claim is not necessarily true.
Figure 4.5: Typical experimental results of the motor current versus time when the curved beam starts hopping with a stable hopping height after changes were made to the routines for acquisition of the data.

Additionally, again by assuming that after the changes the sampling frequency is the one we want and roughly equal to 100Hz, while stable hopping was taking place, the rotating mass had an angular velocity smaller than 2 Hz. On the other hand if the motor current is assumed to be a sinusoid its frequency would be roughly equal to 12.5 Hz which is 6 times higher (at least) than the angular velocity of the rotating mass. This implies that the frequency of the current is much higher than the frequency of the robot’s masses so no synchronization actually exists between them.
Chapter 5

Conclusions and Outlook

The idea of implementing a CPG-inspired controller on curved beam robots was explored. More specifically, an adaptive Hopf oscillator was used as feedback controller directly connected to the system. The main findings can be summarized as follows:

• General Remarks.
  – In conceptual level this approach seems functional (when tested on single and double mass models) for hopping height stabilization as well as hopping height control.
  – A novel way of implementing feedback control using Hopf oscillator for hopping height stabilization and control was developed, via proper selection of teaching signals.
  – Robustness against initial conditions was observed.
  – The force that the oscillator produces is adaptive and minimal intervention is required.
  – A major drawback of this approach is that in order to find a proper teaching signal a trial and error procedure has to be followed.

• Regarding the case of uncoupled dynamics (motor with gearbox having large gear ratio).
  – Hopping height stabilization and control seem possible, using this approach.
  – For hopping height stabilization only the sign of the velocity of the top mass seems to be required (minimal sensory information required).
  – An automated identification procedure has been developed in Labview that is very useful for fast tuning of the velocity controller after changes have been made to the hardware. It is straightforward to be extended for position control.

• Regarding the case of coupled dynamics (motor without gearbox or small gear ratio on gearbox).
  – A new model has been proposed.
  – Experimental data which were acquired after revisions were made to the data acquisition routines contradicted two of the initial hypotheses:
    1. The motor current is sinusoidal.
2. The motor current is synchronised with the movement of the robot masses.

There are two directions to continue the current work. Regarding the uncoupled dynamics case, if the required hardware is acquired (faster controller, DC motor with gearbox, sensors for measuring the top mass velocity and/or height) the approach that was discussed in Chapter 3 can be tested. Regarding the case of coupled dynamics different hypotheses regarding the motor current profile can be extracted by further analysis of the experimental data of the open-loop control case after the changes that were made to the data acquisition routines. The newly proposed model can be proved useful for testing the new hypotheses.
Appendix A

Equations of motion of new curved beam vertical hopper’s abstract model

In order to derive the equations of motion, in closed form, the standard Lagrangian procedure is going to be followed. The equations of motion are given by ([24]):

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = Y_i \tag{A.1}
\]

\[
L = T - V \tag{A.2}
\]

Using the model illustrated in Figure A.1 as a reference, the generalised coordinates for this system are chosen as \( q = \{x_1, x_2, \beta\} \) and \( Y_i \) that refer to the generalised forces acting on each generalised coordinate are given by:
Appendix A. Equations of motion of new curved beam vertical hopper’s abstract model

\[ Y_1 = c(\dot{x}_2 - \dot{x}_1) \]  
\[ Y_2 = -c(\dot{x}_2 - \dot{x}_1) \]  
\[ Y_3 = \tau \]

where \( Y_1 \) refers to \( x_1 \), \( Y_2 \) to \( x_2 \) and \( Y_3 \) to \( \beta \). Notice that the damping of the rotating mass is neglected for simplicity. However introducing damping would be an interesting extension of the current model.

The system’s kinetic energy (T), is given by:

\[
T = \frac{1}{2} m_1 \dot{x}_1^2 + \frac{1}{2} m_2 \dot{x}_2^2 + \frac{1}{2} m_R \left[ (\dot{x}_2 + R\cos\beta)^2 + (\dot{x}_2 + R\sin\beta)^2 \right] \]

(A.6)

and the potential energy (V) by:

\[
V = m_1 g x_1 + m_2 g x_2 + m_R g (x_2 + R\sin\theta) + \frac{1}{2} k (x_2 - x_1 - L_0)^2
\]

(A.7)

The system is hybrid since its dynamics are different during stance and flight phase. The ground impacts are assumed to be plastic (as in [12]). By using equations A.1 - A.7 we obtain the following equations of motion for flight phase:

\[
\ddot{x}_1 = \frac{1}{m_1} \left[ -m_1 g + k (x_2 - x_1 - L_0) + c (\dot{x}_2 - \dot{x}_1) \right]
\]

(A.8)

\[
\ddot{x}_2 = \frac{1}{m_2 + m_R \sin^2 \beta} \left[ R m_R \sin \beta \dot{\beta}^2 - (m_2 + m_R \sin^2 \beta) g - k (x_2 - x_1 - L_0) - c (\dot{x}_2 - \dot{x}_1) - \frac{\cos \beta}{R} \tau \right]
\]

(A.9)

\[
\ddot{\beta} = \frac{1}{R (m_2 + m_R \sin^2 \beta)} \left[ -R m_R \cos \beta \sin \beta \dot{\beta}^2 + \cos \beta k (x_2 - x_1 - L_0) + \cos \beta c (\dot{x}_2 - \dot{x}_1) + \frac{m_2 + m_R}{R m_R} \tau \right]
\]

(A.10)

During stance phase equation A.8 is replaced by

\[ \ddot{x}_1 = 0 \]

(A.11)

Finally the event for touch-down (transition from flight to stance phase) is simply \( x_1 = 0 \) and for take-off (transition from stance to flight phase) is when the spring force becomes higher than the weight of the bottom mass, i.e. \( k (x_2 - x_1 - L_0) > m_1 g \).
Bibliography


