AN ACTUATED FLEXIBLE SPINAL MECHANISM FOR A BOUNDING QUADRUPEDAL ROBOT

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Utku Çulha

January, 2012

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Uluç Saranlı(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Melih Çakmakçı

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Selim Aksoy

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

AN ACTUATED FLEXIBLE SPINAL MECHANISM FOR A BOUNDING QUADRUPEDAL ROBOT

Utku Çulha
M.S. in Computer Engineering
Supervisor: Assist. Prof. Dr. Uluç Saranlı
January, 2012

Evolution and experience based learning have given animals body structures and motion capabilities to survive in the nature by achieving many complicated tasks. Among these animals, legged vertebrates use their musculoskeletal bodies up to the limits to achieve actions involving high speeds and agile maneuvers. Moreover the flexible spine plays a very important role in providing auxiliary power and dexterity for such dynamic behaviors. Robotics research tries to imitate such dynamic abilities on mechanical platforms. However, most existing robots performing these dynamic motions does not include such a flexible spine architecture. In this thesis we investigate how quadrupedal bounding can be achieved with the help of an actuated flexible spine. Depending upon biological correspondences we first present a novel quadruped robot model with an actuated spine and relate it with our proposed new bounding gait controller model. By optimizing our model and a standard stiff backed model via repetitive parametric methods, we investigate the role of spinal actuation on the performance enhancements of the flexible model. By achieving higher ground speeds and hopping heights we discuss the relations between flexible body structure and stride properties of a dynamic bounding gait. Furthermore, we present an analytical model of the proposed robot structure along with the spinal architecture and analyze the dynamics and active forces on the overall system. By gathering simulation results we question how such a flexible spine system can be improved to achieve higher performances during dynamic gaits.

Keywords: Bio-inspired robotics, Legged robots, Dynamic Locomotion, Quadrupedal bounding, Spinal actuation, Gait optimization.

ÖZET

SIÇRAYARAK KOŞAN DÖRT BACAKLI BİR ROBOT İÇİN KONTROL EDİLEBİLİR ESNEK BEL OMURGA MEKANİZMASI

Utku Çulha
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Assist. Prof. Dr. Uluç Saranlı
Ocak, 2012

Evrim ve tecrübe tabanlı öğrenme süreci hayvanlara doğada hayatta kalabilmeleri için birçok karmaşık görevi yerine getirebilen vücut yapıları ve hareket kabiliyetleri vermiştir. Bu hayvanlar arasında özellikle bacaklı omurgalılar, yüksek hızlara erişmek ve keskin manevralar yapabilmek için kaslı iskelet vücut yapılarını kullanırlar. Buna benzer dinamik davranışlarda esnek omurga, havyana fazladan güç ve esneklik desteği vererek önemli bir rol oynamaktadır. Robotik araştırmaları buna benzer dinamik becerileri mekanik sistemlerde gösterebilmeyi amaçlar. Buna rağmen günümüzde benzeri dinamik hareketleri yapabilen robotlar esnek omurga sistemlerine sahip değillerdir. Biz bu tez ile kontrol edilebilir omurga sisteminin dört bacaklı robotlarda sıçrayarak koşma hareketi üzerindeki etkilerini araştırıyoruz. Biyolojik kaynaklara dayanarak, ilk olarak yeni bir kontrol edilebilir omurgaya sahip dört bacaklı robot modeli sunuyor ve bu modeli yine yeni olarak sunduğumuz sıçrayarak koşma kontrolü modeli ile bağdaştırıyoruz. Çok tekrarlı değişkenli yöntemler kullanarak, sunduğumuz esnek robot ve standart robot modelini eniyileştiriyor ve bununla beraber omurga kontrolünün sunduğumuz esnek robot modeli üzerindeki başarım etkilerini araştırıyoruz. Daha yüksek hız ve zıplama yükseklikleri elde ederek esnek vücut yapısı ile dinamik sıçrayarak koşma hareketinin adım özellikleri arasındaki bağı inceliyoruz. Ayrıca tezimizde, sunduğumuz robot modelinin ve robottaki omurga sisteminin analitik incelemesini yaparak, bütün sistemdeki dinamikleri ve kuvvetleri araştırıyoruz. Simülasyon sonuçlarına bakarak esnek bir omurga sisteminin, robotlardan daha yüksek verim alabilmek için ne dereceye kadar geliştirilebileceğini sorguluyoruz.

Anahtar sözcükler: Biyolojiden esinlenmiş robotlar, Bacaklı robotlar, Dinamik hareket, Dört bacakla sıçrayarak koşma, Omurga kontrolü, Yürüyüş eniyileştirme.

Acknowledgement

I would like to thank my supervisor Uluç Saranlı for guiding me through my bachelor and graduate years in my university and giving me his vision, insight and understanding of academics with his utmost energy for this 7 year period. I can honestly say his provision has shaped my career in robotics and possibly the rest of it and I am gladly thankful for all of his efforts so far and so will I be in the rest of my career for he played a very important initiative role.

I wish to acknowledge Melih Çakmakçı and Hartmut Geyer in providing creative ideas and support during the initiation of this project and showing me possible directions about how my research can be extended for many future paths.

I would also like to thank Fırat İlhan for his guidance and help on academic decisions in my career. His intellectual and technical knowledge and friendship have been very important in both my social and professional life, for which I am gladly thankful.

I want to thank all my colleagues in Bilkent Dexterous Robotics and Locomotion (BDRL) group because of their never ending motivation and energy for keeping the research alive and fruitful.

Most importantly, I am grateful to my family who supported me anytime and believed in me since the beginning. It would be impossible without their support to complete any of my duties and fulfilling my ideas including this research. Therefore it will be my pleasure to dedicate this research to my lovely family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Dexterous Robotics and Locomotion

As a result advances in materials science and manufacturing and production of actuation technologies, it has become easier to design and build robots that are more mobile and efficient and that can be used in many different areas in daily life. Currently, mobile and dexterous robots have been used in various tasks such as search and rescue, rehabilitation, surgical operations, exploration and many others where human intervention could be risky or unnecessary [29]. The need for robots increases in parallel with improvements in robotics technology. While robots become more complex and their sets of skills increase, the desired actions that they are asked to complete are also becoming more complex.

However, in almost all of these domains, locomotion, or the action of moving the subject body from one point to another, is one of the most important parts of the whole task. The method used for locomotion also depends on the desired duty of the robot. While many robotic platforms prefer to use wheels due to a large body of knowledge and experience on the design of their structure and control, a relatively new research area focuses on the usage of legs in robots, in manners similar to how they are used in nature.

From this perspective, inspiration from nature plays a very important role in designing leg structures for robots in order to succeed in performing locomotion tasks that might occur frequently in robotic applications. When high speed and maneuverability are required, there are many striking examples in nature which successfully demonstrate the agility of leg morphologies [15]. Animals from the cat family such as leopards and cheetahs and other mammals such as gazelles and goats demonstrate very high performance while running with high speeds, or avoiding obstacles during locomotion. These animals also use their leg and muscle morphologies in the most efficient way to minimize energy consumption, resulting in efficiency in catching their preys or running away from their predators on complex and unstructured outdoor environments [33]. When locomotion patterns and behaviors of these animals are investigated, it is found that the agility of such maneuvers depends on how they maintain balance during their motion. Compared to static balance requirements, where the center of mass of the subject body must stay inside a triangle formed by the group of legs touching the ground at the same time, dynamic balance does not impose such constraints. For different locomotion gaits performed by these animals, there are phases of the gait when none of the legs are touching the ground, but the animal is still in balance. As a result, these animals can run with high speeds and overcome obstacles by jumping high in the air. For these reasons, many researchers in the robotics field have been inspired from these animals and their legged morphologies to create their own bio-inspired multi-legged robots capable of performing gaits based on dynamic balance [29].

Quadrupedal animals in nature adapt different dynamic gaits due to the requirements of different tasks and actions. Considering locomotion in general, gaits are classified with respect to the periods of each leg's event of touching the ground. As these events happen periodically, the whole gait can be represented by only one period, showing the event times in one stride. While some quadrupedal gaits like amble, canter and gallop have asymmetrical patterns with respect to leg events, some gaits like trot, pace, bound and pronk have symmetrical leg patterns [1]. Due to differing advantages and disadvantages of each of these gaits, animals prefer to switch between them during their locomotion. For example, while galloping lets the animal achieve high speeds, bounding and pronking decrease the speed but enable jumping over high obstacles.

Among these dynamic locomotion gaits, bounding stands out with its symmetrical pattern and its maneuverability. In nature, most quadrupedal animals switch to the bounding gait in order to achieve obstacle avoidance in moderate speeds [6]. The symmetrical pattern of the gait has also caught the attention of most robotics researchers because it allows easy implementation with complex mechanisms and control systems. Furthermore, by reducing this gait into simple phases, many quadruped robots have performed bounding by using simple control strategies and leg structures [13, 17, 24, 31].

When all current quadruped robots capable of dynamic locomotion gaits are investigated, it can be seen that nearly all approaches focus on the structure and design of the legs, the complexity and robustness of the control systems or the type of actuation used in the mechanism. Even though these robots are inspired from natural examples, one of the major mechanisms in animals seems to be missing in the implementations: the flexible spine. Some of the fastest and most agile land running mammals use their flexible spine to enhance their performance and energy efficiency during high speed locomotion. The musculoskeletal spine acts as a compliant mechanism to increase the body flexibility, as well as an intermittent unit to transfer energy from front muscles to back or vice versa [3]. It is also used like a spring to give additional thrust to the subject body for jumping tasks [20].

By observing the efficiency of spinal flexibility in nature and locomotion performance of mentioned robotic applications, the question of how such a compliant mechanism can be implemented for quadrupedal robots. The literature in this domain contains few attempts to answer this question by implementing an actuated or passive spine structure to enhance dynamic locomotion gaits used [18, 32]. Even though the idea of implementing a spine mechanism similar to animals sounds trivial, the researchers have focused on solving more basic problems in locomotion such as leg designs and controller systems. However, the state-of-the-art robotic systems can now provide a large range of solutions to these problems and the literature in this field has a big volume. Therefore, by depending on the brought solutions to previous problems, it is now easier to focus on the question of spinal actuation and its effects on dynamic locomotion.

1.2 Contributions

Our main contributions in this thesis are to first introduce a proof of concept model for a spinal actuation mechanism for a planar quadruped robot, running with a bounding gait. This model addresses the question of how a flexible spine could enhance the locomotion performance compared to a stiff trunk. Along with this concept, a new bounding controller that fits this flexible spine model has also been proposed. The comparison of a state-of-the-art stiff backed quadruped model and a newly proposed flexible spine model, running according to a widely used bounding model and the proposed new bounding model has been presented [9]. It is shown that the proposed structure and controller for a flexible spine increases locomotion performance for horizontal body speed and jumping heights both of which are found to be the results of increased stride length.

This thesis expands this concept's contribution by also introducing a detailed mathematical model of a planar quadruped robot model with a flexible spine. We derive the equations of motion for this new spinal mechanism and find various forces acting on the spine and legs. This derivation also enables us to investigate spinal thrust and necessary compliant mechanisms to enhance the locomotion performance as well as decreasing the power costs.

As an overview this thesis proves the positive effects of using a flexible spine mechanism as in a manner similar to how it is used in many mammals, by comparing simulation results of a stiff backed quadruped robot with those of the proposed flexible backed robot.

1.3 Thesis Organization

In Chapter 2 we briefly give background information related to the scope of this thesis. The chapter explains dynamic locomotion gaits used in animals and bio-inspired robots. Then, we focus on bounding gait and present its widely used and accepted control models. This chapter also gives examples from robotic applications on bounding gait and the usage of flexible spine in quadrupedal locomotion. We also explain major gait optimization techniques and focus on the Nelder and Mead optimization algorithm which we used to optimize the parameter set of our quadrupedal models.

In Chapter 3, we describe our proof of concept for a quadruped robot with an actuated spine. In this chapter, the structural model of the robot, the newly proposed bounding gait controller model, and the comparative simulation experiments done are presented and evaluated.

Chapter 4 presents analytical derivation of equations of motion of flexible quadruped model which gives information about dynamics and forces acting on legs and spine. Additional features such as leg retraction and ground friction are also explained. Simulations done with this model and their results are explained at the end of this chapter, along with the discussion of the overall mechanism.

In the last Chapter 5 we give a summary of what we have presented in the thesis and suggest further expansions our research.

Chapter 2

Background

2.1 Locomotion Gaits

2.1.1 Major Gait Types

Animals have adapted different locomotion types through evolution and learning through experience. Even though animals differ greatly in size, functionality and form, a big portion of them use common locomotion gaits. These gait types can be classified with respect to the duration of feet contact with the ground and the symmetry of events within a single stride [1].

Alexander describes a single stride in the whole gait as a complete cycle of leg events starting from the touching down of one particular leg until its next touchdown and all other legs touching the ground only for once during this period [1]. He also uses a descriptor called $dutyfactor$ of a foot β, which denotes the fraction of a whole stride period for which the foot is on the ground. Considering this factor, he selects a reference foot, which is generally the fore left foot, and describes different types of gaits in quadrupedal mammals as in Figure 2.1.

It can be seen from the figure that amble, canter and gallops show asymmetric patterns as each foot spends a different duration on the ground. However, other

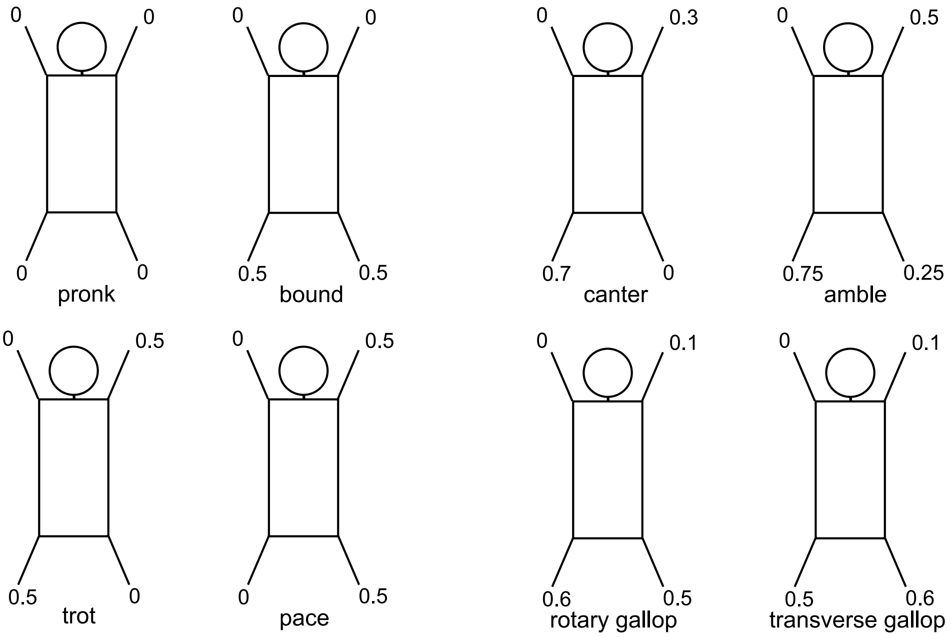

Figure 2.1: Quadrupedal gait types inspired from Alexander's work. The group on the left shows symmetrical patterns, while the four gait types on the right have asymmetric periods for each leg.

gait types represent symmetrical patterns. Differences in each gait also brings different advantages. While animals prefer symmetric gaits for slow speeds but complex maneuvers, asymmetric gaits are generally preferred to achieve higher speeds on dynamically stable tracks. Picturing a horse running at high speeds with a rotary gallop and an antelope jumping over high obstacles using the pronking gait can give the reader an idea about the usage of these gaits.

In robotics, symmetric gaits are generally more preferred than asymmetric gaits. The reason for such a choice lies within the simplicity of representing a single stride with less complexity. This low complexity also helps researchers to identify gait controllers on reduced space dimensions since groups of legs having identical event durations and triggers can be represented as a single leg. For example, as all of the legs in the pronking gait act identically within the stride, all 4 legs can be represented as a single leg [2]. Also, legs on the same diagonal line in the trot gait or legs on the same side of the body in the pace gait can reduce control complexity in the overall system [27].

2.1.2 A Gait Model For Bounding

Compared to the other dynamic locomotion gaits, bounding is the most implemented on robotic platforms. Due to its symmetrical pattern and coupled event maps for front and back legs, bounding reduces control complexity. However, in order to understand the advantages of this gait, one must analyze the mechanics of phases within a single stride of bounding.

First of all, the most appealing features of this gait is the coupled use of the front and back legs [1]. In quadrupedal bounding, we see that both of the front legs have exactly the same periods of motion within the overall stride. In other words, they touch the ground at the same time, and leave the ground at the same time. This is also true for the two back legs. When this symmetry is considered, quadrupedal bounding becomes very easy to be restrict to the sagittal plane for analysis.

The planarization of quadrupedal bounding reduces the dimension of the problem from a 3D world into a planar, 2D world [26]. In a planar world, a pair of legs can be represented by only a single leg, which is a very efficient way of modeling bounding. As bounding has the leg symmetry, the front legs and the back legs of the quadruped can be summarized with a single leg for each corresponding pair. This also reduces the number of controllers for each leg, as only one controller will be needed to control each leg pair, which are now combined into a single leg on the sagittal plane.

Considering this planarization, bounding has been represented by two different approaches so far. Both of the approaches are widely used by many robotics platforms due to their low level of complexity. Both of these methods represent a single stride of bounding in four consecutive phases. Moreover, each method relies on the detection of each leg contact event and change their controller parameters and strategies accordingly. Major events that need to be detected are the *touchdown* and *liftoff* of each leg pair. By only depending on these leg events, none of these bounding gait controllers need to consider the state of the body explicitly. Detecting these two events for each leg, both models achieve successful dynamic locomotion via bounding.

The first method [2] assumes that there can be only one leg touching the ground at a time. Consequently, every time a leg touches the ground and leaves it afterwards in the half of the stride period, the other leg is controlled for touchdown for the rest of the stride. On the left part of Figure 2.2, this controller's state machine can be seen. Although this gait controller has 4 visible phases, it has two switching controller states. Whenever the current controlled leg leaves the ground after the liftoff event, the controller switches to the other leg to complete another cycle. By using this strategy, the same controller with different parameters can be used for each leg by only detecting the touchdown and liftoff events.

The alternate method, [23], includes a double stance phase, hence differing from the first method. As it can be seen from Figure 2.2, the state machine shown on the right shows a double stance phase which is triggered by the touchdown event of back leg, which happens before the liftoff event of the fore leg.

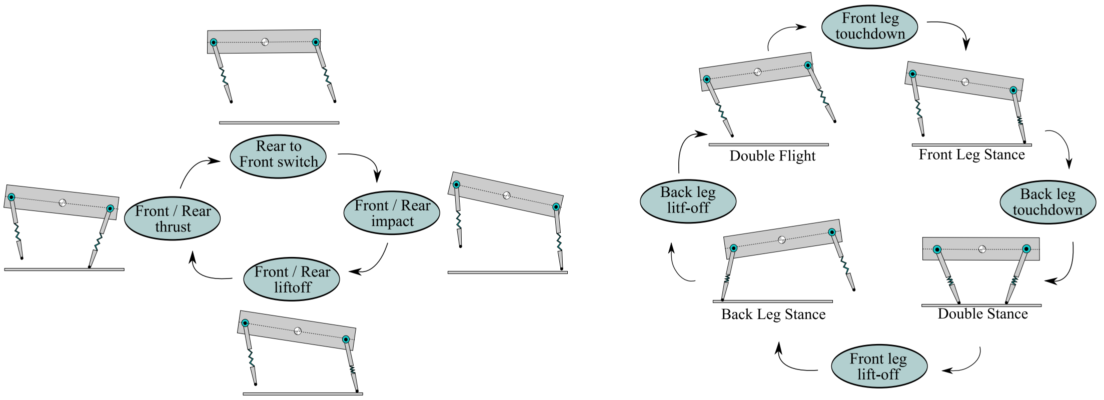

Figure 2.2: Two different planarized bounding models. (left) The first model assumes that there could be at most one leg on the ground at a time. (right) The second model has a double stance phase.

2.1.3 Legged Robots Capable of Dynamic Bounding

Some of the first contributions to bio-inspired legged robot research were done by Raibert et al. [26], based on a dynamic model of a single leg. The Spring-Loaded Inverted Pendulum (SLIP) model, was found to be very close in performance compared to animal legs when active forces and dynamics are investigated. As implied by the name, the SLIP model consists of a point mass attached to a spring, which touches the ground on the other end. Very similar to the dynamics of an inverted pendulum, the overall body acts as an elastic pendulum on the ground. When the tip of the spring touches the ground, the kinetic energy on the point mass starts to be converted into potential energy, which is stored in the compliant leg mechanism during stance. While the spring gets compressed due to active forces, the point mass follows a sinusoid-like trajectory. When the spring is in the full compression, the body mass reaches its lowest height on the overall trajectory. After that point, the potential energy stored on the spring mechanism starts to be transferred into kinetic energy with the leg entering its thrust phase. Figure 2.3 shows an example trajectory of this model throughout a single stride.

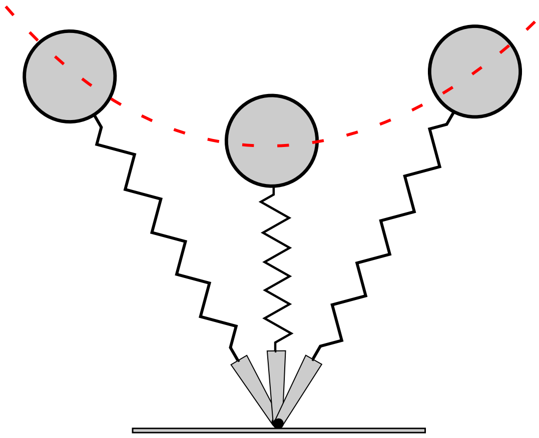

Figure 2.3: The Spring-Loaded Inverted Pendulum (SLIP) model, showing an example trajectory of the point mass during stance.

Despite being an approximation, the SLIP model has been and is still being used as the basis for most hip actuated robot designs. One of the most important aspects of this model is its simplicity. The leg itself is composed of a compliant spring, so the contraction and retraction phases of the leg on the ground can be maintained by the nature of spring dynamics without any external controls. Leaving the ground dynamics of the leg to the passive spring, the only control needed for the whole leg is for the hip joint which connects the leg rod onto the body. This hip actuator's job is to control the position of the leg with respect to the body and adjust an angle differing with respect to the phases of a dynamic gait. Raibert's quadruped robots use this basic design to achieve different dynamic gaits such as trotting, pacing and bounding [27]. He incorporates two levels of control for quadrupedal bounding. The high-level controller for gait phases decides on desired angles for legs for touchdown and liftoff events. This controller is independent of body state, but only acts upon detected leg events. In contrast, the low-level controller consists of a local PID feedback loops to maintain angles selected by the high-level controller.

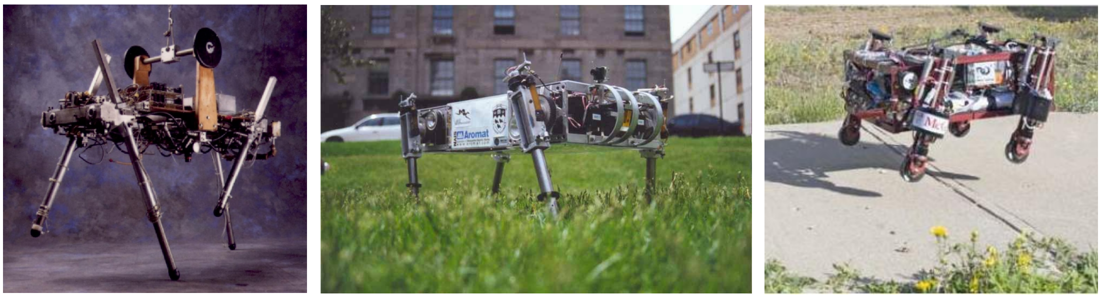

Figure 2.4: Raibert's quadruped (left), SCOUT II platform (middle), PAW robot (right)

Based on the same SLIP model, Buehler et al. introduced his quadruped robot platforms SCOUT and SCOUT II [4, 22]. Similar to Raibert's quadruped robots, SCOUT platforms only include hip actuators to control the angles of legs and base their dynamic locomotion gaits to compliant leg dynamics. With reduced complexity in the robot design, SCOUT platforms achieved various dynamic gaits successfully and further supported the use of simple controllers for dynamic legged locomotion gaits.

RHex, a hexapod robot designed by Saranli et al [28], is also a bio-inspired multi-legged robot. Trying to mimic a cockroach's effective locomotion behaviors over unstructured terrain, RHex has been successful by achieving stable locomotion on different grounds. Apart from adaptive gaits with respect to the terrain, RHex platform was also used for bounding experiments [5]. By deactivating middle legs of the hexapod, quadrupedal bounding gait was still possible even with RHex's different leg designs.

Somewhat differently than SCOUT platforms, Smith et al. developed a hybrid leg system by placing a lockable wheel at the tip of each leg [30]. By using such a system, the PAW robot managed to overcome slippage problems encountered during the ground phases of the legs. By virtue of being lockable, these wheels could be adjusted to be fixed or rolling with respect to the phase of the gait or the terrain conditions in means of roughness. One important aspect of this design is that it incorporates a leg with a wheel to benefit from dynamic properties of both structures.

Apart from the usage of passive dynamics on the legs, there have also been other approaches on robotic designs to achieve bounding and other gaits. While previous robot designs depended on passive dynamics of spring mechanisms of the legs, these robots have multiple degrees of freedom (DOF) leg designs [13, 17, 25]. The multi-linked leg designs enabled a closer look into natural leg forms and created more accurate results compared to SLIP approximations. However, all these multi-links need additional actuation controls and sensors which increase the level of complexity in the overall gait control designs.

2.2 Existing Work on Flexible Spine Structures

All robots performing dynamic locomotion behaviors explained so far adopt different approaches for control strategies and leg structures. However, one of the main properties they have in common is their adaptation of a rigid trunk. Even though bio-mechanics research shows the importance of spinal structures on a flexible body [3, 10, 20] , there are few inquiries into their role in robotics research.

The most detailed research done so far is Karl F. Leeser's planar quadruped and dynamic locomotion experiments done on this robot, presented in his thesis [18]. Using the planarization method of Raibert, this quadruped was developed on a reduced dimensional state space. The main motivation behind this platform was to investigate the role of an articulated spine on the thrust given by the back legs.

In order to understand the effects of the spine, Leeser designed a robot with

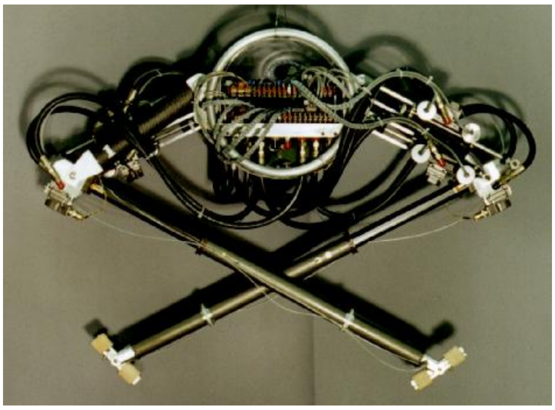

Figure 2.5: Leeser's planar quadruped with an articulated spine. All actuation is done through hydraulically-powered motors. (Figure adapted from [18])

two prismatic legs and an articulated spine mechanism, consisting of three parts controlled by hydraulically-powered actuators. As the robot has only two legs, the robot was attached to a planarizing boom in order to maintain its balance in the real world experiments, which is basically a rotating rod connecting the robot body to a fixed point in the center of a circular test track. With this rod, the robot was able to move on the circumference of the track as if it was running on a 2D plane.

In his thesis, Leeser experimented on this robotic platform to achieve a bounding gait. The design of this robot enabled the adjustment of spinal stiffness so he was able to compare a stiff configured robot with a flexible configured robot. Moreover he proposed two different control methods to analyze the spinal thrust on the back of the body. In his first approach, he controlled the spinal joints to maintain open loop static positions during various parts of the locomotion to adjust the rate of compression of leg springs. In his second approach, he positioned these joints so that they could act as a vertical spring during the take off phase of the gait cycle.

His research shows that, compared with a stiff backed system, an articulated spine mechanism can give additional power and thrust to the robot in dynamic gaits. It can be seen from his results that with both of his control techniques a flexible spine slightly increases the hopping height and robot speed. However his second control approach, in which he positions the spine to give vertical thrust, the compression rate on the back legs drops while the robot maintains a higher jumping height. This shows that the spine takes on some part of the total thrust activity and gives additional power to the system.

Even though Leeser's work has not been improved since then, his initial research gives an idea about how spinal flexibility can be achieved and under which conditions it can be effective. His analytical and experimental work reveals an undiscovered method in order to improve the dynamic behaviors that legged robots can achieve.

Additional research done on spinal flexibility by Takuma et al., shows its role on the yaw direction during quadrupedal locomotion [32]. By constructing a simple quadruped robot, Takuma et al. investigates the role of spinal compliance. Unlike previous robots, their robot does not maintain dynamic locomotion and moves much more slowly. However, by means of showing a direction towards using compliant bodies instead of rigid ones, this new research carries an importance in the robotics field.

2.3 Gait Optimization Techniques

2.3.1 General Approaches

Locomotion plays a very important role in the robotics field and all legged robots are required to complete their tasks by accommodating complex mechanics of running or walking gaits. Although the output of gait control looks very simple, a large set of parameters exist in the background. In order to get the best performance from the legged gaits, these parameters need to be optimized for the specific robot structure, desired output and terrain properties.

As stated before, animals have optimized their gaits through evolution and learning by experience. It is clear that the evolution process cannot be fully applied to physical robots in a long time period, but numerical models and simulations can help to solve the problem of finding an optimal set of parameters by trying and approximately evaluating a larger number of candidates. To this end, a number of different methods have been proposed using on machine learning techniques or adaptive control strategies.

The main idea in machine learning methods is to start with an initial set of parameters that define gait properties and iteratively evaluate different sets and try to find an optimal set to yield the best performance. Generally the performance criteria is chosen to be the speed or the stability in the gait. For this purpose, a cost function is generated by evaluating the results of an experiment done with a particular parameter set. In order to get to the best parameter set, different algorithms and functions may be used.

Genetic algorithms randomly choose a pair of parameter sets and crosses them over to generate a new seed parameter set [16]. If this new seed set outperforms the selected parent sets, then this new seed is placed over the parents. The evaluation of the artificial evolution method is done with experiments run on a robot trying to achieve a task given the generated gait parameters. At the end of each gait, the results are given to the main algorithm to check whether the genetically produced seed is better than the parents [8]. In another method named *Gaussian Process Regression*, the elements in the parameter vector are considered to be the randomly observed values of a non-observable larger function. Then these values are given to a Gaussian function to generate an approximation of this hidden larger function. By changing the values of these elements through an evaluation process, the algorithm is expected to produce a function which produces a Gaussian representation of an optimum parameter set [19]. In a similar fashion, by using a policy gradient method the gradients of the cost functions of each parameter set are explored. The best gradient yielding the optimum results is selected to form another set. By repeating this process multiple times, a global minimum is expected to reach [12].

In a different approach, controller parameters are changed online, while the robot is in motion, getting sensory feedback and adapting locomotion parameters with respect to changing conditions. This method is called adaptive control looks similar to adaptation in natural locomotion done by animals such as changing the stiffness of leg muscles due to the roughness of the terrain. In general, adaptive control uses an additional observer system in a closed feedback loop, which changes the desired gait controller parameters that are given to the local controller by looking at the results of the gait. With this double loop, the observer maintains stability and the desired performance of the gait itself, while the local controller adjusts local parameters to perform the strides in the gait [34].

2.3.2 Nelder-Mead Optimization Algorithm

In this thesis, we will be using an optimization method by proposed by Nelder and Mead [21]. The main idea in this method is to minimize a function with n parameters, by evaluating it on (n+1) vertices of a simplex in the parameter space. New parameters are generated depending on the values of these vertices, replacing or displacing them as appropriate.

The algorithm starts by initializing parameter points P_i at user selected n+1 vertices. The iterative process starts by finding P_h and P_l, the highest and lowest cost generating points. The algorithm then finds the centroid of the n points, with P_h removed, called as $\bar{P}$. Following that, each P_h is compared with the results of three different points generated by three methods: *reflection, contraction* and *expansion*. The reflection phase finds a point called P^* that is defined as

$$P^* = (1 + \alpha)\bar{P} - \alpha P_h$$

where α is a positive constant called *reflection coefficient*. The cost value of this point, y^* is calculated and compared with y_l. If $y^* < y_l$, expansion phase starts and another point P^{**} is calculated

$$P^{**} = (1 + \gamma)P^* - \gamma\bar{P}$$

where γ is the positive *expansion coefficient*. Similarly, the cost value of this point, y^{**} is found and compared with y_l. If $y^{**} < y_l$ the point found in expansion is replaced with P_h, however if case fails, point found in reflection is replaced with P_h. On the other hand, if the comparison case fails in the reflection phase, the algorithm enters the contraction state. If y^* is larger than all y_i except y_h, then P_h is replaced with P^*, but if it is also larger than y_h, then contraction phase finds another point P^{**} as

$$P^{**} = \beta P_h + (1 - \beta)\bar{P}$$

where β is the positive *contraction coefficient*. In case of y^{**} of this point is greater than either of y_h or y^*, all points in the simplex is replaced with $(P_i{+}P_l)/2$. On the opposite case, P_h is replaced with P^{**} and algorithm ends its round by checking if the best point has been found. These methods are used iteratively each round until the convergence to the best point. This method proposed by Nelder and Mead can be considered as a simplex that is manipulated to get to its smallest size by extruding or pushing in the vertices. The detailed algorithm is given in A.1.

Chapter 3

Bounding with Flexible Spine

In this chapter, we will present two basic structural models we use to compare bounding with a stiff-spine robot to bounding with our new flexible-spine robot. Moreover, we give a novel bounding gait controller suitable for use with our spine actuated robot.

3.1 Planar Robot Models

3.1.1 Standard Model with a Stiff Back

In Chapter 2, we described a number of robotic platforms that use stiff trunks but different leg structures and controller systems. In this section, we will focus on robots that use the SLIP model as a basis for their leg designs and a rigid body for their trunks. The model that will be presented here is widely used in many robotic platforms which is why we use it as a reference. Based on this model, we will focus on mainly three performance criteria: hopping height, horizontal speed and power consumption.

Figure 3.1 shows a planar quadruped model with a stiff spinal structure. The robot consists of three main parts: two passive spring legs and a stiff body trunk.

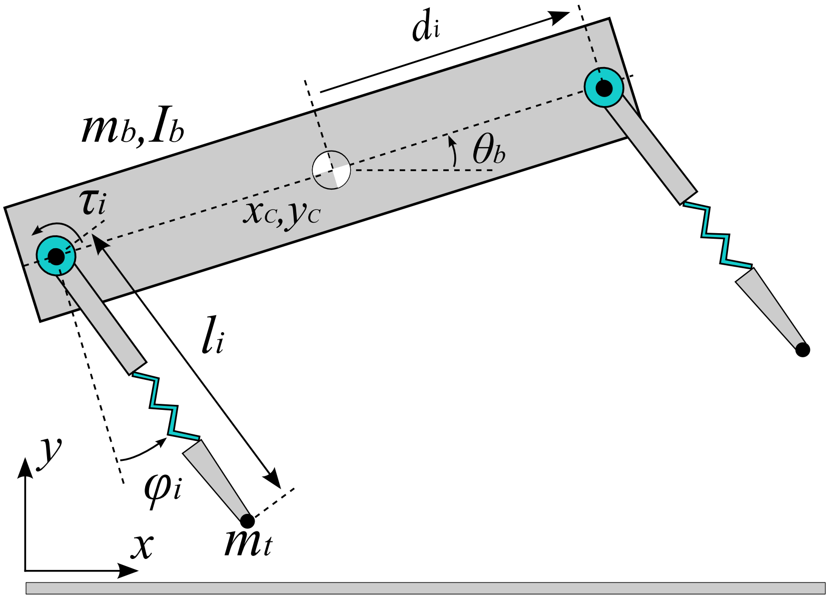

Figure 3.1: The planar quadruped robot model with passively compliant legs and a stiff body structure.

The body has mass m_b and inertia I_b, and its center of mass (COM) is located at point (x_C, y_C) in the inertial world frame W. It has a pitch angle θ_b, defined in the counter-clockwise direction from the x axis of W.

Two identical legs with spring-damper systems with spring constant k and damping constant b are attached to the body at d_i away from the COM. For the sake of simplicity, the leg attachment points are assumed to be vertically aligned with the COM. Both legs have a rest length of l_i, and have a toe mass of m_t at the tip of their toe. This toe mass is very small compared to the robot body mass in order to reduce inertial effects during the flight phase of the leg. Despite the fact that the legs are massless, the toe mass is nonzero to implement meaningful flight dynamics.

Each hip is controlled with DC motor, producing an input torque τ_i. Legs can be in either one of two phases: stance or flight. The equations of motion for this model are derived in many other references in the literature [22, 23, 26]. In this chapter, we will be using Working Model 2D to numerically simulate its dynamics.

3.1.2 New Model with a Flexible Spine

In this section, we propose a new planar robot model with an actuated spine mechanism. This robot model is one of the major contributions of our thesis. Differing from the standard, stiff-backed model explained in the Section 3.1.1, our model has flexible body structure similar to land mammals described in Chapter 2. This extension is expected to increase dynamic gait performances compared to the rigid body robots. Again, our performance criteria will be hopping height, horizontal speed and power consumption for this model.

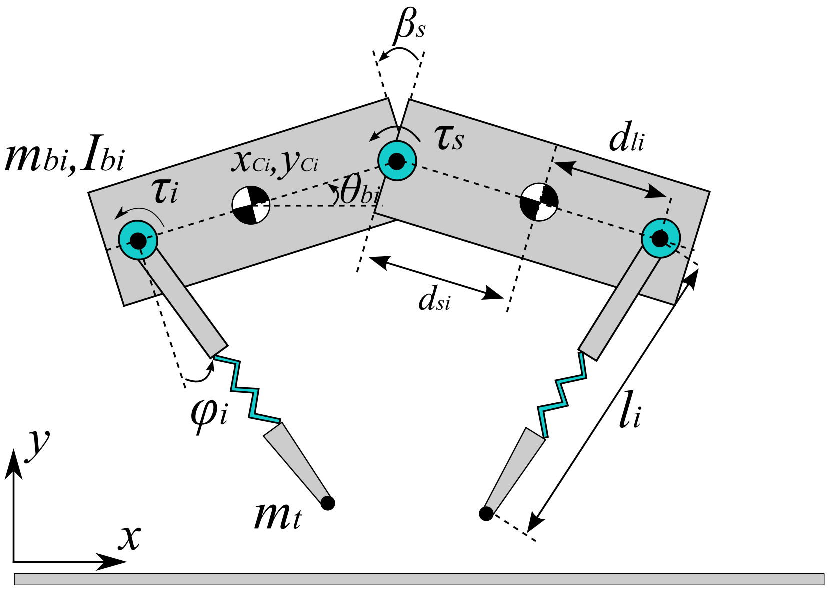

Figure 3.2: The proposed planar quadruped model with an actuated spine joint and hip actuated compliant legs attached. The model is inspired from nature in order to increase the dynamic gait performance of the robot.

Figure 3.2, depicts our planar quadruped model with a flexible spine. Unlike the previous model, our robot consists of four main parts: two rigid body elements connected to each other with a pin joint and two spring-damper compliant legs attached to each body part. The main difference of this model is the spinal joint connecting both of the body parts in the middle. Each body i has mass of m_{bi} and inertia I_{bi}, and its COM is located at (x_{Ci}, y_{Ci}) in W. The associated body pitch angle θ_{bi} is defined similar to the previous model, in the counter-clockwise direction from the x axis of W.

The main focus of this model is on the spinal joint, which is the merging point of the two body parts. Placing an electric motor on this joint, we are able to produce a spinal torque τ_s between the bodies. The spinal joint is vertically aligned and d_{s_i} away from the COM of each body. The angle between two body parts, the spinal angle, is represented with β_s.

Similar to the stiff backed model, we use two identical spring-damper legs attached to each body segment. The rest length of each massless leg is denoted with l_i and each leg is attached to the body through an actuated hip joint, controlled with individual DC motors that produce a torque τ_i with respect to leg angles φ_i, defined in the counter-clockwise direction from the body horizontal. At the tip of each leg, there is a very small toe mass m_t, assumed to be negligible for their inertial effects on the body.

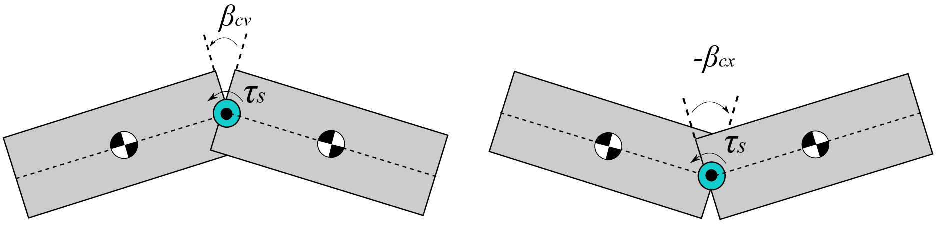

Figure 3.3: Two phases of the flexible spine; concave (left) and convex (right)

Like the standard stiff back model, each leg can be in one of two phases: stance and flight. In addition to these, our gait controller can give body two separate poses. As shown if Figure 3.3, the body can take a concave form or a convex pose defined by the changes in the spine angle β_s. By introducing these two body poses, we claim that the robot will behave similar to its natural correspondences which are explained in Section 1. Our gait controller bends and stretches the body structure by changing the spinal angle, which is expected to increase the stride length of the robot. In other words, the flexibility of the body will give the legs a longer activity area which will eventually increase the length of a stride in the cycle of the whole gait.

3.1.3 The Working Model 2D Environment

Before a mathematical analysis of our model with, we compare the standard bounding model with our new model in a dynamic simulation tool. For this reason we have selected the Working Model 2D [11] simulation environment to run our simulations and produce initial results for our model.

Working Model 2D is a simulation environment which solves dynamic equations and constraints in a planar world. The environment uses fixed but configurable time steps or variable time steps to integrate differential equation solutions associated with a dynamical system. Either the Euler or the Kutta-Merson integrator can be chosen to solve the dynamics. Moreover, the user can configure the resolution of overlap and integration errors.

In order to create a simulation world, the user can place geometric body parts and define joints between these. Spring-damper systems, external forces and torques, linear and rotational actuators and other types of gear systems can be added into the simulation as constraints. However, two constraints cannot be attached to each other without having a physical body unit in the middle. The same rule applies to body parts as well: there must be a constraint defined between two adjacent body parts. Mass, inertia, elasticity, static and kinetic friction constants and electric charge of a body can be configured as well.

After defining the relations between these constraints and body parts, the selected integrator solves dynamic equations with the selected time step. However, the integrator can only solve up to 32 seconds of simulation. The user can also define input and output monitors for every kind of constraint and state in the system. As such, forces acting on different components can be tracked throughout the whole simulation.

There is also a scripting feature of the Working Model 2D. It is similar to the Basic programming language, and with it user can create, modify, configure, control and track every possible body and constraint in the simulation environment. By using this tool, data output can also be obtained for further analysis of simulation results.

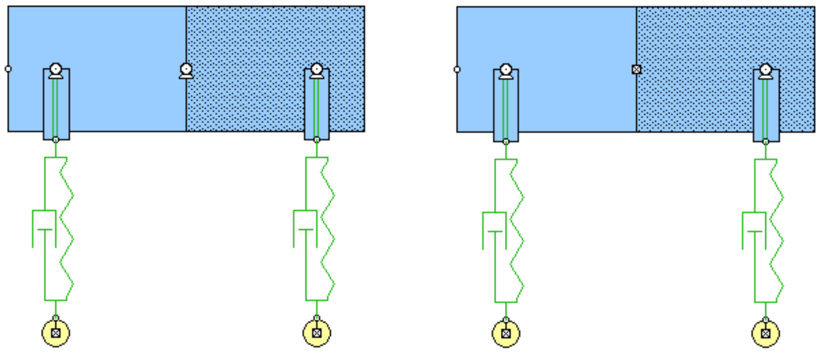

Figure 3.4: Flexible back (left) and stiff back (right) planar robot models created in Working Model 2D environment. The stiff-backed model has an anchor constraint between two body segments.

By using this tool, we created two robot models as shown in Figure 3.4. Based upon the conceptual models shown in Section 3.1.1 and Section 3.1.2, the flexible robot model has two body segments with a rotary actuator in the middle. For different spinal angles, these body segments can assure different poses as shown in Figure 3.3.

Because of the body-constraint-body rule, the legs are defined as a combination of bodies and constraints. As seen in the figure, the upper limb of the leg is a physical body connected to a spring-damper constraint system. This limb part is attached to the corresponding robot body part with a rotary actuator constraint on the hip joint. The tip of the spring-damper constraint is attached to a circular mass, which represents the toe. There is also a virtual vertical slider between the upper limb of the leg and the toe to avoid the toe and the spring system to bend in directions other than the radial during the compression phase. In other words, this virtual slider lets the spring compress in the radial leg direction only. The model on the right is the stiff backed robot. By converting the spine joint into a fixed joint, we created a single rigid body trunk. The remaining body parts and their physical properties are identical to those of the same with the flexible back robot.

3.2 Gait Controllers

3.2.1 A New Bounding Gait Model

In this section, we present our novel bounding gait model. The standard bounding gait models explained in Chapter 2 can only be used for stiff-backed robots. For a quadruped robot with an actuated spine, a new bounding gait controller with flexible body phases needs to be designed. Consequently, relying on the planarization method explained by Raibert in [26], we extend on the standard bounding gait model to work with our new flexible robot system.

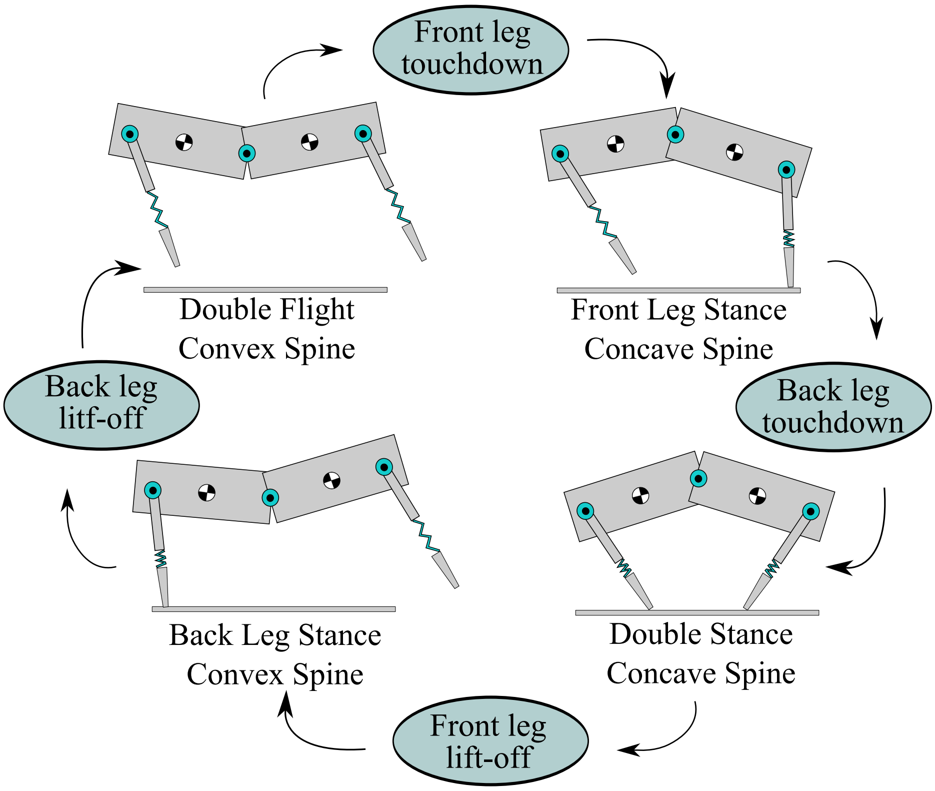

Figure 3.5: The new bounding gait model with different poses of the flexible spine in our new model.

Figure 3.5 shows the state machine associated with the new bounding gait model. It can be seen that a single stride in the bounding is represented with four consecutive phases, each having a unique set of properties and triggering events. Selecting the double flight phase as our reference, we can track the rest of the gait very easily. In the double flight phase, the spine takes up a *convex* pose bending upwards, to extend the reach of front legs. In this phase, both of the legs are positioned to their desired touchdown angles which are fixed parameters of the gait controller. The touchdown angle is assumed to be non-negative to increase the range of the legs and give them a longer stance period. Together with the convex spine pose and the front segment's extended angle, the range of the front leg increases more than the stiff backed robot model. The double flight phase ends with the touchdown event of the front leg. When the front leg stance phase is initiated, the spine starts to bend in the opposite direction to form a *concave* pose. The idea behind this inner bending is to increase the reach of the back leg before it touches down. A similar approach cannot be applied to a stiff back robot running for the bounding gait. After the front leg touches the ground, it starts to swing back towards the center of the body, trying to maintain the liftoff angle for that leg.

The front leg stance phase ends with the touchdown event of the back leg and the double leg stance phase starts. In this manner our bounding gait looks similar to Poulakakis's model [23]. During this phase, the spine starts to bend in the opposite direction again to its convex pose. The change of body the pose in the double stance phase is designed to give additional thrust to the back leg in its compression period. In addition to this, it also gives the front leg sufficient space to lift off, without getting stuck on the ground. This choice enables the robot to achieve higher speeds without losing its balance. While the front and back legs are both controlled to maintain their liftoff angles, the front leg leaves the ground triggering the entrance to the third phase of the gait model: back stance. In this phase, only the back leg stays on the ground and the front leg starts to swing forward to its touchdown angle while the spine continues to bend to reach its maximum convex pose angle. This phase ends when the back leg also leaves the ground and starts to swing forward similar to the front leg. A single stride cycle in the bounding gait hence ends with this event and the robot re-enters its double flight phase.

Compared to a standard bounding gait model, our new model uses the flexibility of the spine and changes the body pose to exploit the abilities of a flexible body. First of all, the changes in the body pose give both legs an increased stride length, which is useful to increase the horizontal speed of the robot. Moreover, the bending of the front body segment outwards just after the double stance phase pulls the front leg spring forward so that it can be lifted off from the ground easier and reduces the risk of falling. Finally, this bending strategy gives additional thrust to the body during the double stance phase as the spinal joint acts like an additional spring.

3.2.2 Design of Bounding Controllers

In this section, we give detailed descriptions of bounding gait controller parameters used in both models we have presented: the standard gait model with double stance phase and our proposed flexible spine controller. In both of these models we only depend on leg touchdown and liftoff events, which are assumed to be detected using pressure sensors at the toes of each leg. Apart from that, the only sensors we use are encoders attached to hip joints and the spine joint in our new model. Our controllers hence depend on angular position data from the encoders and contact states of the legs. We also use the position encoders to track the angular velocity of the legs during swinging to maintain a constant speed for the legs.

From now on, we will be referring to gait controllers as the high level controller, and the PID controllers that adjust leg and spine angles as the low level, or local controller.

3.2.2.1 Stiff Backed Gait Controller

Table 3.1 shows different states of the state machine that controls the bounding gait in the stiff-backed model, which was also used in other robotic platforms and experiments [7, 24, 31]. The first column in the table shows the name of the current phase in the gait cycle, whereas the last column shows the event that initiates the start of this phase. The middle column shows target angles for each

Table 3.1: High-level state machine for stiff-backed bounding.

State	**Target Angles**	**Trigger Event**
Double Flight	$(\varphi_{b_{td}}, \varphi_{f_{td}})$	Back leg lift-off
Front Leg Stance	$(\varphi_{b_{td}}, \varphi_{f_{lo}})$	Front leg touchdown
Double Stance	$(\varphi_{b_{lo}}, \varphi_{f_{lo}})$	Back leg touchdown
Back Leg Stance	$(\varphi_{b_{lo}}, \varphi_{f_{td}})$	Front leg lift-off

leg in the corresponding phase. The first row shows that the double flight phase starts when back leg liftoff event is detected. In this phase, the back and front legs are commanded to maintain their desired touch down angles $\varphi_{b_{td}}$ and $\varphi_{f_{td}}$ respectively. The second row shows that front leg stance phase starts with the detection of front leg touchdown event. In this phase, the back leg continues to maintain its touchdown angle as before and the front leg is given a new desired angle $\varphi_{f_{lo}}$ as its liftoff angle. The double stance phase starts after the touchdown event of the back leg, following which back leg is given its new liftoff target angle $\varphi_{b_{lo}}$. Finally the last phase starts when the front leg leaves the ground and is given the desired touchdown angle as in the first double flight phase. In all of these phases, the back and front legs are controlled to maintain an angular velocity of $\dot{\varphi}_b$ and $\dot{\varphi}_f$ respectively.

All of these parameters are given to local PID controllers as explained in Section 3.2.3. To reduce the number of parameters, we used the same PID gains for both legs. So, along with the parameters given above, a particular instance of the bounding gait can be represented with the following parameter vector

$$\mathbf{p}_{sb} := [\varphi_{b_{td}}, \varphi_{b_{lo}}, \dot{\varphi}_b, \varphi_{f_{td}}, \varphi_{f_{lo}}, \dot{\varphi}_f, K_p, K_i, K_d]^T , \tag{3.1}$$

where K_p, K_i and K_d are PID controller gains for proportional, integral and derivative terms.

3.2.2.2 Flexible Backed Gait Controller

Table 3.2 shows the state machine for flexible backed bounding. In this table, the details of the controller are very similar to those of stiff backed bounding.

So, by reusing the same events and sensors, we managed to extend the bounding gait controller by including the spinal joint. Once again, the first column shows the entered state, the middle shows target angles and the last column includes triggering events.

Table 3.2: High-level state machine for flexible backed bounding.

State	**Target Angles**	**Trigger Event**
Double Flight	$(\varphi_{b_{td}}, \varphi_{f_{td}}, \beta_{cx})$	Back leg lift-off
Front Leg Stance	$(\varphi_{b_{td}}, \varphi_{f_{lo}}, \beta_{cv})$	Front leg touchdown
Double Stance	$(\varphi_{b_{lo}}, \varphi_{f_{lo}}, \beta_{cx})$	Back leg touchdown
Back Leg Stance	$(\varphi_{b_{lo}}, \varphi_{f_{td}}, \beta_{cx})$	Front leg lift-off

We previously mentioned that this flexible gait controller is very similar to the stiff backed controller, so we will only explain different events in each phase. In the double stance phase, the spine joint is commanded to bend outwards to its convex spinal angle denoted with β_{cx}. When the front leg touches the ground, the spine is then commanded to bend inwards in the opposite direction, and position itself to a concave spine angle β_{cv}. When the double stance phase starts, the spine bends outwards again to β_{cx} in order to yield additional space for the front leg to lift off. Entering the back leg stance phase does not change the desired angle for the spine, which continues to maintain its convex angle to provide auxiliary thrust to jump up higher.

Apart from the desired angular velocities for the legs, we now have a third desired angular velocity for the spine motor; $\dot{\beta}_s$. Moreover, additional PID controller gains are also required for the spine. With these new parameters added to existing ones, a single parameter set for a flexible bounding gait can be represented with the following vector:

$$\begin{aligned}\mathbf{p}_{fb} \;:=\; & [\varphi_{b_{td}}, \varphi_{b_{lo}}, \dot{\varphi}_b, \varphi_{f_{td}}, \varphi_{f_{lo}}, \dot{\varphi}_f, \beta_{cx}, \beta_{cv}, \dot{\beta}_s, \\ & K_p, K_i, K_d, K_{ps}, K_{is}, K_{ds}]^T, \end{aligned} \tag{3.2}$$

where K_{ps}, K_{is} and K_{ds} are the low level proportional, integral and derivative controller gains for the spine actuator.

3.2.3 Local Controllers

In all phases of the bounding controllers in Section 3.2.2, we used local PID controllers for each actuator to determine associated torque commands. Torque commands for legs in both models and the spinal joint are computed as

$$\tau_j = K_p e_j(t) + K_j \int_0^t e_j(t)dt + K_d \frac{de_j(t)}{dt}, \tag{3.3}$$

where j represents either the leg number or the body joint. Leg and body tracking errors are respectively defined as $e_i(t) := \varphi_i^*(t) - \varphi_i(t)$, and $e_b(t) := \beta_s^*(t) - \beta_s(t)$.

The high level controller determines the desired angles for both legs and the spine in all phases of the bounding gait. These angles, along with low level controller gains and other parameters are collected in the parameter vectors $\mathbf{p}_{sb}$ and $\mathbf{p}_{fb}$ for the stiff-backed and flexible-spine models, respectively. The computation of the desired angle at a single time instance involves the usage of both the desired angle and angular velocity parameters given in the state vector. For both models, the computation of the desired angles for the legs depend on whether they are individually in stance or flight. During stance, we have

$$\varphi_i^*(t) = \begin{cases} \varphi_i(t_{td}) + \dot{\varphi}_i(t - t_{td}) & \text{if } t - t_{td} < \frac{\varphi_{i_{lo}} - \varphi_{i_{td}}}{\dot{\varphi}_i} \\ \varphi_{i_{lo}} & \text{otherwise .} \end{cases} \tag{3.4}$$

Similarly, during flight, we have

$$\varphi_i^*(t) = \begin{cases} \varphi_i(t_{lo}) - \dot{\varphi}_i(t - t_{lo}) & \text{if } t - t_{lo} < \frac{\varphi_{i_{td}} - \varphi_{i_{lo}}}{\dot{\varphi}_i} \\ \varphi_{i_{td}} & \text{otherwise .} \end{cases} \tag{3.5}$$

In the equations given above, i represents the leg index; back or front more specifically. We can also see the details of the angular velocity control in equations (3.4) and (3.5). For example, for the stance phase at a given time t, the low level controller controls the time required to reach the desired liftoff angle. In order to do that, the local controller first computes the time passed since the it has positioned itself in the touchdown angle. This time interval is shown as $t - t_{td}$ in the formula above. Using the angular velocity parameter $\dot{\varphi}_i$ and the computed time, the controller updates the desired angle $\varphi_i^*(t)$. With this method, the desired angle of a leg with respect to the state of the gait changes according to a desired angular velocity. The change in the desired angle can be represented by a graph as shown Figure 3.6.

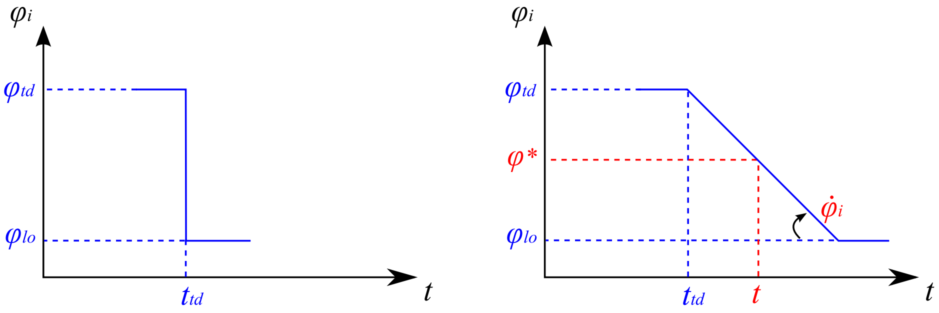

Figure 3.6: Trajectory generation system shown for an angle control in a stance phase of a leg. Compared to step change (left), trajectory tracking (right) updates $\varphi_i^*(t)$ in each time step with respect to slope or $\dot{\varphi}_i$.

So instead of changing the desired angle to its full value in a single time step like it is shown on the left graph, we are using the angular velocity as a slope to reach to the desired value. By using this method we can also regulate the amount of torque produced at the actuators. With trajectory tracking, we also eliminate peak torques that can be produced at step changes due to big amounts of difference between current and desired angles.

The spine actuator is controlled in a similar fashion with its concave and convex poses using the target angles determined by the parameter vector in 3.2. The desired body angle for the concave body pose is computed as

$$\beta_s^*(t) = \begin{cases} \beta_s(t_{ftd}) + \dot{\beta}_s(t - t_{ftd}) & \text{if } t - t_{ftd} < \frac{\beta_{cv} - \beta_{ftd}}{\dot{\beta}_s} \\ \beta_{cv} & \text{otherwise ,} \end{cases}$$

whereas for the convex pose it takes the form

$$\beta_s^*(t) = \begin{cases} \beta_s(t_{flo}) - \dot{\beta}_s(t - t_{flo}) & \text{if } t - t_{flo} < \frac{\beta_{cx} - \beta_{flo}}{\dot{\beta}_s} \\ \beta_{cx} & \text{otherwise .} \end{cases}$$

3.3 Simulation Results

In this section we present simulation experiments comparing our flexible model with the standard stiff-backed model with the bounding gait. In order to make a fair comparison we use optimization methods to find the best performing gait parameters for both models. Subsequently we evaluate the results and compare them to discuss the effects of spinal actuation on the bounding gait.

3.3.1 Configuration and Initialization

We implemented the models described in Section 3.1.3 using the Working Model 2D simulation environment. We used the Kutta-Merson integrator with a fixed time step of $10^{-3}s$. To increase the accuracy of the system, the integrator, assembly and overlap error tolerances were chosen to be less than $5 \times 10^{-3}m$. Due to the limitations of the simulation environment, every test run lasted up to a maximum of 32 seconds.

Table 3.3: System parameters for both bounding models.

Param.	**Value**	**Param.**	**Value**
m_{bi}	$10\ kg$	m_b	$20\ kg$
I_{bi}	$1.3\ kg\text{-}m^2$	I_b	$3.85\ kg\text{-}m^2$
d_i	$0.365\ m$	d_{si}	$0.25\ m$
k	$3500\ N/m$	b	$55\ Nm/s$
l	$0.8\ m$	τ_{max}	$200\ Nm$
μ_s	0.9	μ_k	0.8
$i \in \{f, b\}$, f: front, b: back μ_s, μ_k : Static and kinetic friction			

Physical properties were used for the robot models were inspired from the morphology of a cheetah [10]. It is important to note that both robot models had the same values for these parameters to enable a fair comparison. As the aim of this thesis is to investigate the effects of spinal flexibility on dynamic bounding gaits, we wanted to eliminate other differences between the two robot models. Table 3.3 shows values of each system parameter used in the simulations. The values of these parameters can be adjusted when a practical application of this research is to be implemented.

The mass of each robot body segment, m_{bi}, was 10 kg which makes both models weigh 20 kg in total. The length of each body segment was 0.5 m, making the robot 1 m in total length. As the flexible back robot model consists of two body parts, each part has a separate moment I_{bi}. However, the total inertia of these two robot body parts is equal to the moment of the stiff back robot, I_b. The distance between the center of mass of each body part and the spinal joint is represented as d_{si} and it is the half the length of a body segment. The hip joints are attached to the body parts 0.115 m away from the COM. While this distance is represented as d_{li} in the flexible robot model, it is depicted as d_i in the stiff backed robot. Although the representations are different in each model, the total distance between the legs is the same in both models assuming the spine in the flexible model stays in its neutral angle making front and back body segments parallel to each other.

Legs have an initial rest length l_i of 0.8 m and the constants for spring, k, and damping, b, are 3500 N/m and 55 Nm/s respectively. Toes attached to these legs have only $5 \times 10^{-3} kg$ of mass. The static and kinetic friction constants given in the table to mimic contact relation between rubber on concrete. Moreover, we use a saturation limit, denoted with τ_{max}, on the DC motors to make the simulation more realistic.

Given these static parameters and control parameters determined by the high level controller, each simulation started with an initial forward speed of 1 m/s and a height of 0.75 m. The pitch angle of the robots were chosen as 0 rad and in order to make this, the spinal angle β_s of the flexible robot was also chosen as 0 rad. Throughout the whole simulation, a stability check was performed. A simulation is considered stable if the norm of its state vectors at successive apex points of the robot stay within a threshold value of 10^{-1} of their average for at least 5 strides. If the run is successful, in other words the robot managed to stay in the bounding gait without losing its balance, a cost function was calculated at the end of the simulation. Being inspired from the widely used performance criteria of *specific resistance* [14], we have defined our cost function as

$$\epsilon := P/mgv^3,$$

where P denotes either the *instantaneous* or *absolute* power spent by all the actuators on the robot and v is the average horizontal speed of the robot. If the gait was found to be stable with respect to the criteria above, this cost function takes the power and velocity values from the last 5 strides of the run. Otherwise, to emphasize the high cost of unstable running, these values are taken from the beginning of the simulation until the end. This last choice could remain the same with stable running as the last 5 strides of an unstable gait would also yield larger costs than a stable gait.

3.3.2 Nelder-Mead Optimization on Gait Parameters

Considering the details given in Section 3.3.1, our simulations on both models are based on implementing the Nelder-Mead optimization algorithm explained in Section 2.3.2. By giving an initial set of parameters for each robot model, the algorithm runs a modified sets of parameters and finds their cost values at the end. These cost values are then considered and the set of parameters are updated until convergence is achieved. The convergence criteria C is defined as

$$C = \sqrt{\sum_i (\epsilon_i - \bar{\epsilon})^2/(D+1)},$$

where ϵ_i denotes the cost value of each run, $\bar{\epsilon}$ is the mean of all cost values and D the dimension of the parameter set. We have selected a threshold value of 10^{-3} and all optimization rounds continue until the result of the convergence function falls below this level.

Regarding the number of parameters in the state vectors defined by the high level controllers, the dimension of the parameter set for a stiff backed robot is $D = 9$ and for a flexible backed model is $D = 15$. With respect to the requirements of the Nelder-Mead algorithm, $(n + 1)$ number of vertices for an n parameter problem must be defined. The selection of these initial parameters for

$(n + 1)$ vertices can impact the time required for convergence to the optimum set. The range of each parameter in a set should be defined properly to give the algorithm a large variety of choices.

3.3.3 Results of Gait Optimizations

With given initial parameter sets and the convergence threshold, the Nelder-Mead optimization algorithm found two optimal sets for both the stiff-backed and the flexible-spine models. Figure 3.7 shows a snapshot from the convergence of cost functions of the two robot models to the given threshold values.

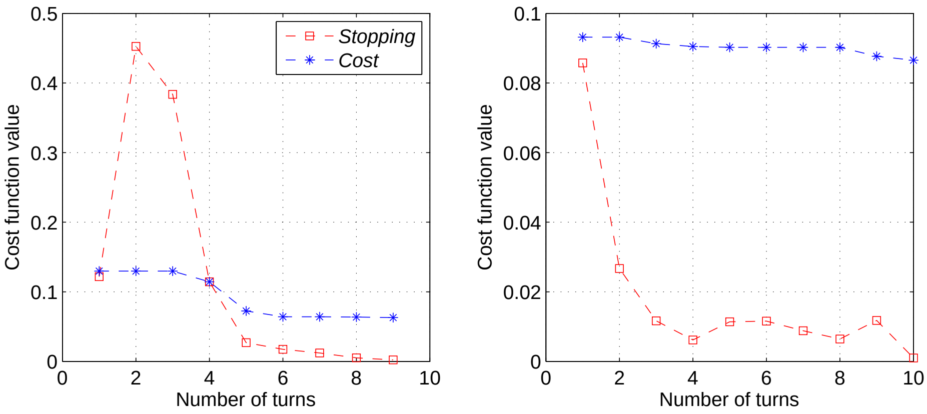

Figure 3.7: Progression of the Nelder-Mead optimization for stiff backed (left) and actuated spine (right) models. Red squares plot the stopping criteria function C, whereas blue stars represent the best vertex cost values for each simplex. Each "turn" corresponds to five Nelder-Mead iterations.

The graph on the left shows the cost convergence for the stiff backed robot, whereas the graph on the right shows the flexible backed model. Each data point on the graph gives information about 5 consecutive Nelder-Mead cycles. In this case, the optimum set for the stiff backed robot has been found in 45 cycles. It took 5 more cycles to find the optimum set for the flexible backed robot model. Although the trend in the graphs is descending towards the threshold value, at some point the cost value seems to increase. For the flexible-backed model, these particular cases are due to the failed contraction phase of the algorithm where the only solution to find the global minimum is to expand the size of the vertices, which eventually increases the cost function at each vertex.

Once the algorithm converge to the optimal minimum, two sets of parameters are produced for bounding gait parameters. The resulting parameter sets for both models are shown in Table 3.4 and 3.5.

Table 3.4: Optimal gait parameters for Stiff Backed Bounding

Parameter	**Value**
K_p, K_i, K_d	(338.2, 0.08, 6.7)
$\varphi_{td_f}, \varphi_{lo_f}, \dot{\varphi}_f$	(0.4 rad, -0.03 rad, 4.25 rad/s)
$\varphi_{td_b}, \varphi_{lo_b}, \dot{\varphi}_b$	(0.27 rad, -0.11 rad, 4.5 rad/s)

Table 3.4 shows the optimum parameter values found for stiff-backed bounding. It can be seen that the saturation limit of the motors and the angular velocity control has limited PID gains and they have not exceeded realistic values. When target angles for the legs are compared, it can be seen that the touchdown angle for the front leg is larger than the back leg. We can reason from this difference that the front leg aims to increase the range of its reach to both increase the stride length and add stability to the system. The wider the front leg's touchdown angle, the more the amount of counteracting ground force on the body in the opposite direction to the action. This would work as a breaking system for the whole robot and it would keep the horizontal speed within a stable region. Finding an optimum angle between two ends will ensure that the front leg will both be keeping the robot in balance and increase its stride length as much as it can.

Similarly, we see that the swing velocity of the back leg is slightly higher than that of the front leg. This can be a result of the back leg's role in providing thrust to the system. The angular velocities for both legs should be adjusted reasonably to guarantee that the back leg will be thrusting the robot up to a stability limit and the front leg will keep the robot in balance without crashing onto the ground. We can also reach this conclusion by looking at the target liftoff angles. It can be seen that the back leg is swung backwards more than the front leg. By doing so, the duration of the stance phase of the back leg increases as well as the spring forces acting on the body. A higher ground reaction force due to the compliant legs will be created in the same direction with the locomotion of the robot if the back leg is swung backwards more.

Table 3.5: Optimal gait parameters for Actuated Spine Bounding

Parameter	**Value**
K_p, K_i, K_d	(524.7, 0.11, 6.9)
K_{ps}, K_{is}, K_{ds}	(1737, 0.03, 330.3)
$\varphi_{td_f}, \varphi_{lo_f}, \dot{\varphi}_f$	(0.3 rad, 0.17 rad, 3.43 rad/s)
$\varphi_{td_b}, \varphi_{lo_b}, \dot{\varphi}_b$	(0.3 rad, -0.12 rad, 4.93 rad/s)
$\beta_{cx}, \beta_{cv}, \dot{\beta}_s$	(0.1 rad, -0.22 rad, 23.05 rad/s)

Table 3.5 shows optimal parameters found for the flexible back bounding gait. By looking at the first two rows, we can see that, the gains for the spine joint are larger than those of the hip joints. This can be the result of the need for a stronger actuation mechanism for the spine system. We can also reason that the spine joint deals with larger amounts of forces during locomotion when compared to the legs. A valid reason for this result is that the weights and inertial forces of two heavy robot bodies can put up a large amount of torque on the spinal joint and that is why a stiffer actuator is needed.

If we look at the target angles for the legs, we can see that the roles of the legs follow a similar fashion in the stiff backed bounding. Although the touchdown target angles for both legs are found to be the same, in this model the spine angle also changes the pose of the body as well as the touchdown angles of the legs. We know from our bounding gait model that when the robot is in the double fight phase, the body is in its convex pose. As this angle, β_{cx} is found to be 0.1 rad, this offset needs to be added to the front leg and subtracted from the back leg. So eventually, we will be seeing a difference between the touchdown angles of both legs, confirming the role of legs as the front leg in stride length increase and stability and back leg in thrusting. Liftoff angles also show the same difference, however in this model, only the back leg is swung behind the vertical of its attached body. This can be explained with the concave body pose due to the negative concave spine angle. By including this offset created by the spine angle, as we did for the touchdown angles, we can find changing liftoff angles which will support the requirements of leg roles. Moreover, the swinging velocities differ from each other around 1.5 rad/s favoring the back leg. The swinging velocity for the spine has been found to be much larger than both of the legs which tells us the need for a faster spine actuation to react to the changing torques on the spinal joint during locomotion.

3.3.4 Simulation Results

By using the optimal parameter values given in Table 3.4 and 3.5, we have run both models with the bounding gait to gather data and compare their results. We have used the same configurations and initial settings we explained in Section 3.3.1 for our robots. We ran the robots up to 32 seconds and collected data from different parts by monitoring corresponding outputs. In this section, we will present results we have obtained from these simulations.

3.3.4.1 Stiff Backed Bounding Results

The optimum parameter set for the stiff backed bounding resulted in a stable and successful bounding gait of whose snapshots can be seen in Figure 3.8. The figure shows a complete cycle in a single strike of the stiff backed bounding. We can see that the execution of phases in a stride cycle follows the phases of the standard stiff backed bounding we have shown in Chapter 2.

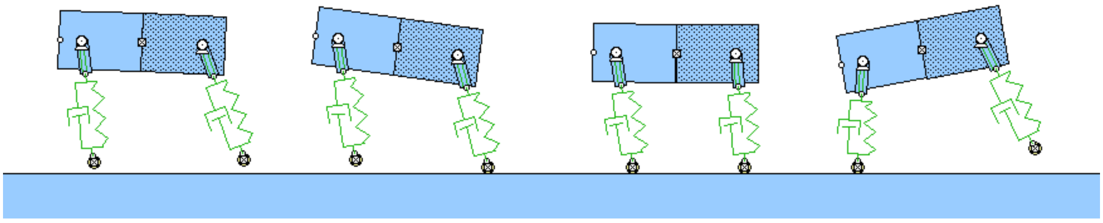

Figure 3.8: Snapshots of stiff backed bounding model during Working Model 2D simulations.

Figure 3.9 shows the compression rate of the leg springs and the pitch angle of the body during the gait. The graph on the top shows the leg lengths during the changing phases in the last 5 strides. It can be seen from the figure that the back leg compresses more than the front leg. The back leg compresses down to 0.71 m while the front leg compresses down to 0.73 m. The graph on the bottom shows the pitch angle of the stiff robot body. We can see that the pitch angle of the body oscillates between -0.1160 rad and 0.1390 rad.

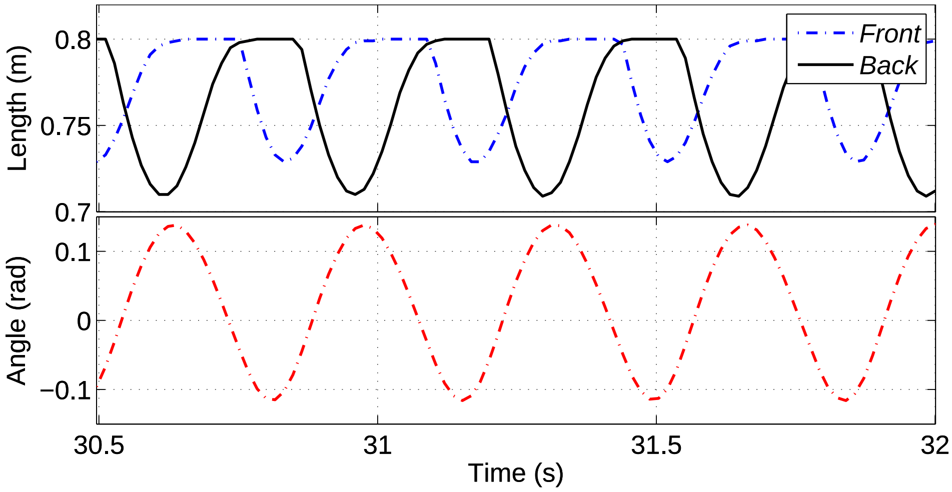

Figure 3.9: Leg lengths (top) and body pitch angle (down) for the stiff backed bounding model.

The next figure, Figure 3.10, is a compilation of different performance criteria for the stiff backed bounding. Each of these graphs shows the data acquired from the last 5 strides of the stiff backed robot during bounding. The topmost graph shows the hopping height of the robot, taking the center of mass point as a reference. It can be seen that the robot body oscillates between 0.735 m and 0.755 m in each stride. The green dashed line in the middle of the graph shows the average height of the robot which is 0.7441 m. The next graph in the middle of the figure shows the horizontal speed of the robot. With respect to the nature of the bounding gait the horizontal speed changes due to the states in a cycle of a single stride. The maximum speed is reached in the double stance phase and the minimum speed is reached in the double flight phase. With the optimum parameters, the stiff backed model can reach up to 2.004 m/s of horizontal velocity. However for a fair comparison, the average velocity is taken into consideration which is 1.747 m/s which makes 1.74 $bodylengths/s$.

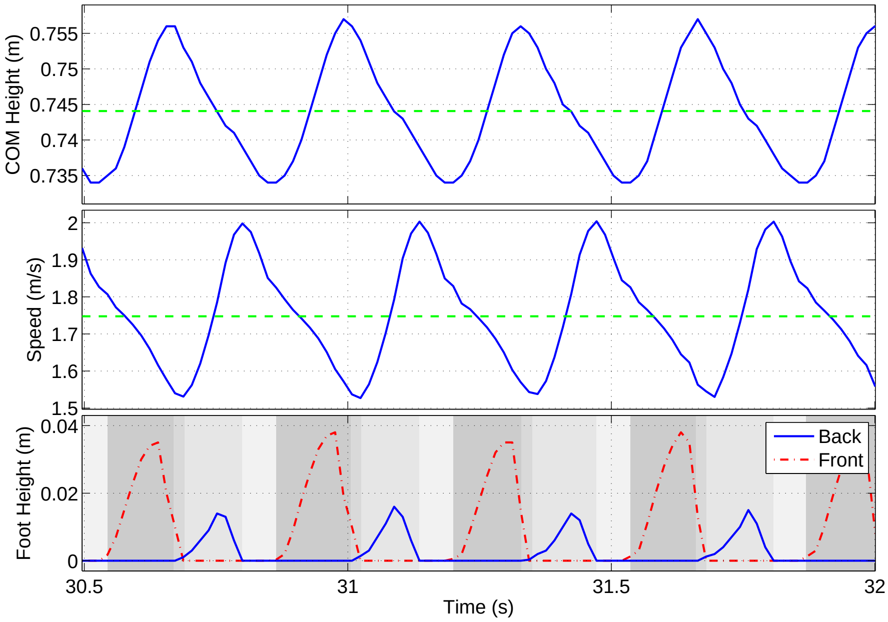

Figure 3.10: Body height (top), horizontal velocity (middle) and foot clearance (bottom) trajectories for bounding with the stiff-backed model. Green dashed lines in the top two plots indicate the average horizontal speed and heights. Shaded regions in the bottom indicate different controller phases.

The graph on the bottom includes data showing the foot clearance of the legs and the duration of each phase in a single stride. By looking at the dashed and solid lines in the graph, we can see that front foot has a ground clearance of 0.0380 m and back feet has 0.0160 m. The shaded regions on the background of the graph show the duration of each of the four phases of the stiff bounding gait. A single stride lasts approximately 0.32 s, and within this stride the list of phases in the descending order of duration is back leg stance (0.13 s), front leg stance (0.11 s), double stance (0.06 s) and double flight (0.02 s). Regarding these results the stiff backed robot in the bounding gait spends only 1/16th of the whole gait flying.

Figure 3.11 shows the trajectories of leg angles maintained by the hip actuators governing the PID controllers. It can be seen that both of the legs reach their corresponding target angles in stance and flight. This graph proves that the local controller achieves the target angles presented in Table 3.4 for the stiff backed model.

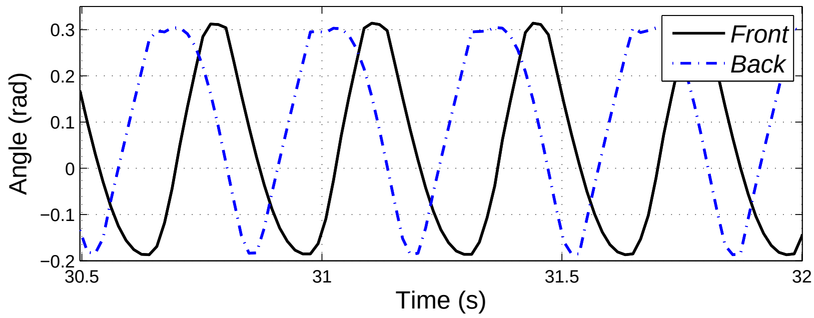

Figure 3.11: Leg angles of the back leg (dashed blue line) and the front leg (solid black line) during bounding motion.

Figure 3.12 consists of two graphs showing the power consumption and the torque outputs of the hip actuators. By looking at the power consumption graph, we see that neither of the hip actuators use more power than 400 W instantaneously. It can be seen that the graph shows also negative power values which indicate that the average power consumption values for the hips will be much lower. The graph on the bottom shows the torque output of each motor on the hips. Although being limited with a saturation level of 200 Nm, none of the motors exceed 90 Nm level.

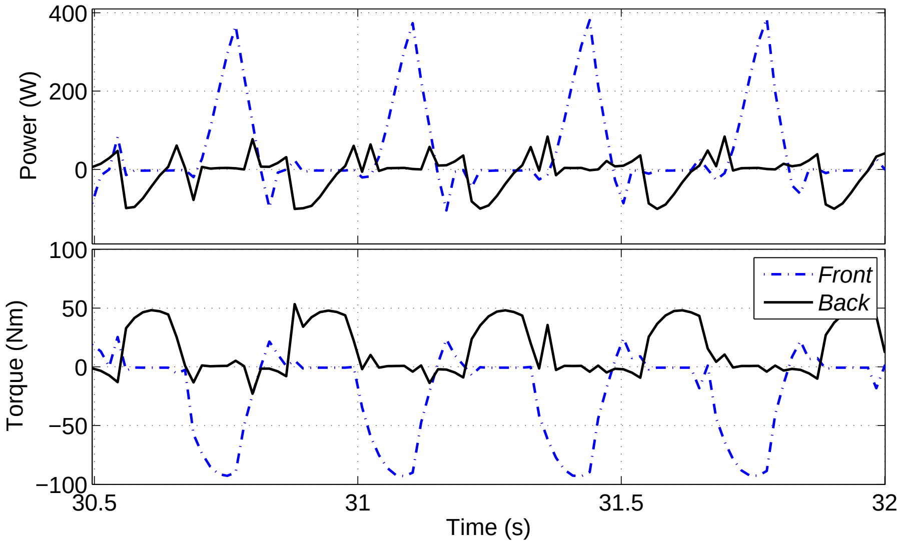

Figure 3.12: Power consumption (top) and torque output (bottom) of motors. Blue dotted line represents front hip motor and black line back hip motor.

3.3.4.2 Flexible Backed Bounding Results

We have run our flexible backed robot with the optimum parameter set found by the Nelder-Mead algorithm and observed the successful bounding gait shown in Figure 3.13. Similar to the stiff backed bounding snapshots before, this image show frames from the consecutive phases in the flexible backed bounding gait we have proposed in this research. The snapshots clearly shows that the spine angle changes with respect to the state of the gait to adjust the body pose.

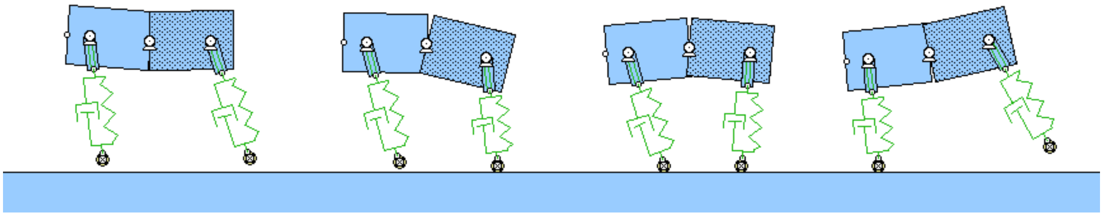

Figure 3.13: Snapshots of the flexible backed bounding model during Working Model 2D simulations.

In Figure 3.14, the changing leg lengths and pitch angles of body parts of the flexible robot model is given. On top graph, leg lengths follow a pattern similar to the stiff backed robot; the back leg compresses more than the front leg. We see that the amount of compression of the back leg is 0.105 m while it is 0.093 m for the front leg. In the bottom graph the pitch angles of two body parts of the robot is given. As the spine actuator changes the body pose, these body parts have differing pitch angles with respect to the state of the gait. We see that back body part oscillated between -0.151 rad and 0.146 rad, while front body part has a range between -0.228 rad and 0.259 rad.

The Figure 3.15 includes three graphs showing the body height, horizontal speed and the feet clearance of the flexible robot. In the first graph on the top, we see the trajectory of the center of mass of the robot during bounding. The center of mass of the whole system has a range of 0.08 m. As the whole robot consists of two body parts attached together on the spine joint, the trajectory of the center of mass point diverges from a smooth sinusoid curve. On the middle graph we observe the instantaneous and average horizontal speed of the robot.

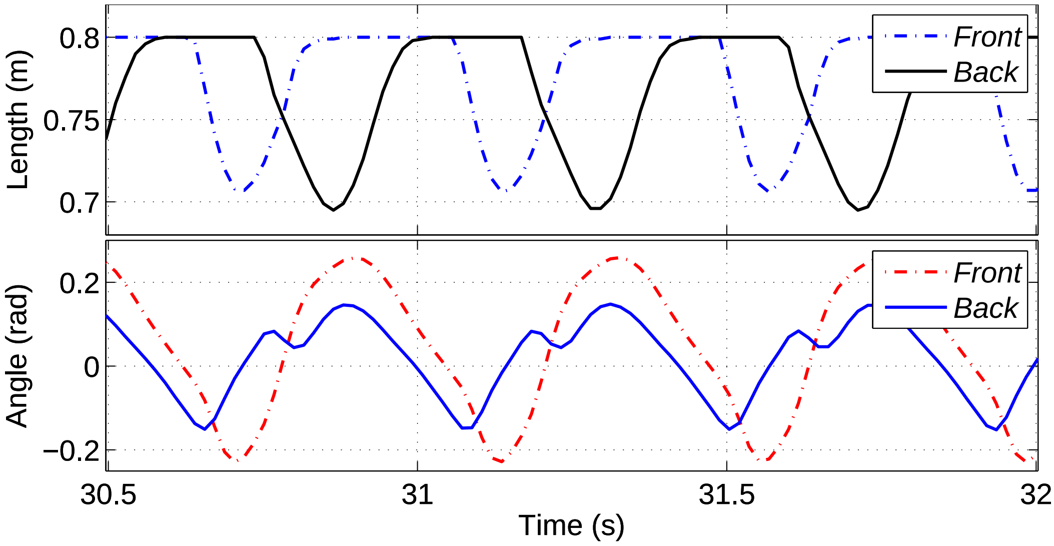

Figure 3.14: Leg lengths (top) and body part pitch angles (bottom) of flexible bounding gait model.

As it can be seen the robot can reach up to 2.23 m/s velocity, the average speed of the system is 2 $m/$.

Similar to the stiff backed bounding graphs, the last graph includes the data of feet clearance and the system states together. We see that in flexible bounding the front legs are off the ground up to 0.177 m where back legs jump up to 0.046 m. We also see an extended bounding stride in the figure. A single stride takes approximately 0.44 seconds and each phase has an increased duration compared to the stiff backed model. In flexible bounding; back leg stance lasts 0.18 s, double flight 0.10 s, front leg stance 0.11 s and double stance phase 0.05 s. We observe that in the flexible bounding the double flight phase is nearly 1/4th of the whole stride.

Figure 3.16 shows the angular trajectories of both legs and the spine during locomotion. We can see that actuators placed on the corresponding joints manage to achieve the target angles, which can also be seen from Table 3.5, found by the optimization method. It can be seen that spine actuator bends the body in convex and concave poses by looking at the bottom graph in the figure.

In Figure 3.17, the power consumption and torque outputs of each motor on

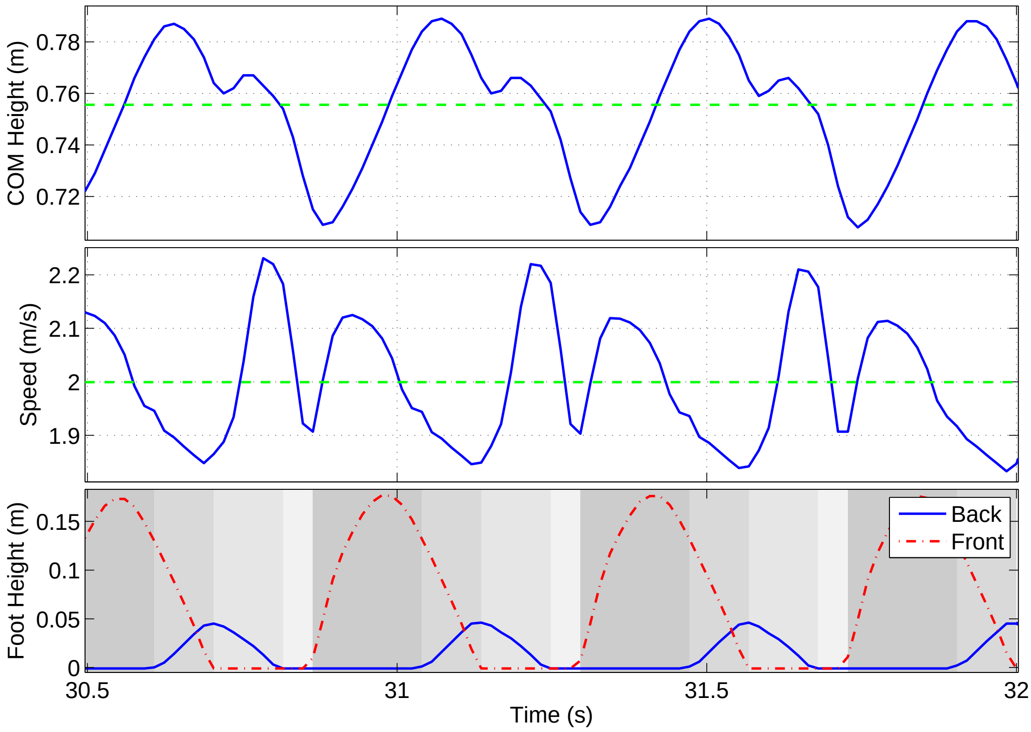

Figure 3.15: Body height (top), horizontal velocity (middle) and foot clearance (bottom) trajectories for bounding with the actuated spine model. Green dashed lines in the top two plots indicate the average horizontal speed and heights. Shaded regions in the bottom indicate different controller phases.

the robot model are shown. On the top graph we see the power consumption of each motor in the system. Except from the spine motor, the hip motors consume a similar amount of power with the stiff backed robot by not exceeding 500 W. However, the spine motor reaches up to a level of 1200 W during locomotion which shows the amount of work done by that particular actuator. In a similar fashion we observe the torque outputs of each motor to be increased such that spine motor saturates at the maximum torque level. Although the spine motor is saturated at the limit, the hip joints produce torque outputs lower than 120 Nm at maximum.

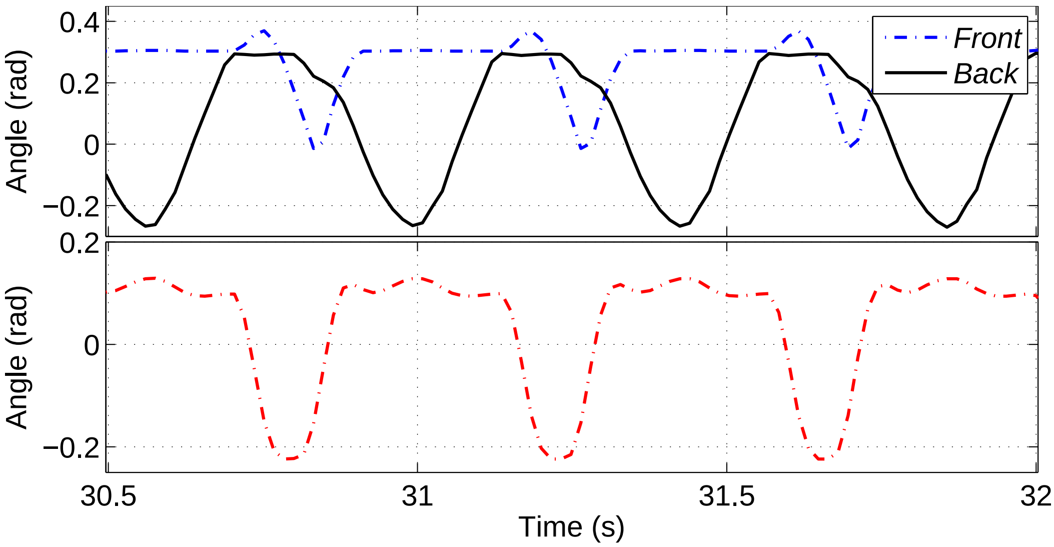

Figure 3.16: Angular trajectory of the legs (top) and the spine (bottom) during bounding motion.

3.4 Discussion

In Section 3.3, we used two different robot and bounding gait models to investigate the effects of spinal actuation on dynamic locomotion. We developed a new robot model with an actuated spine and a flexible bounding gait controller and compared these two systems with a standard stiff backed model which are widely used in other robotic platforms. For a fair comparison we used Nelder-Mead optimization algorithm to find the optimum controller parameter sets to maximize the bounding performances of each robot. Finally we ran two simulations with these optimum parameters applied to the corresponding robots to gather performance data. In this section, we are using these information to discuss the effects of spinal actuation on dynamic bounding gait.

However it is important to emphasize that, the performance results we found are only bound to the parameters we selected for the robot structure, simulation environment and calculation accuracy. With these results we are only comparing the impact of spinal actuation on two robots with same structural background. Therefore, we are not claiming that our robot models can outperform similar robot structures running in dynamic gaits.

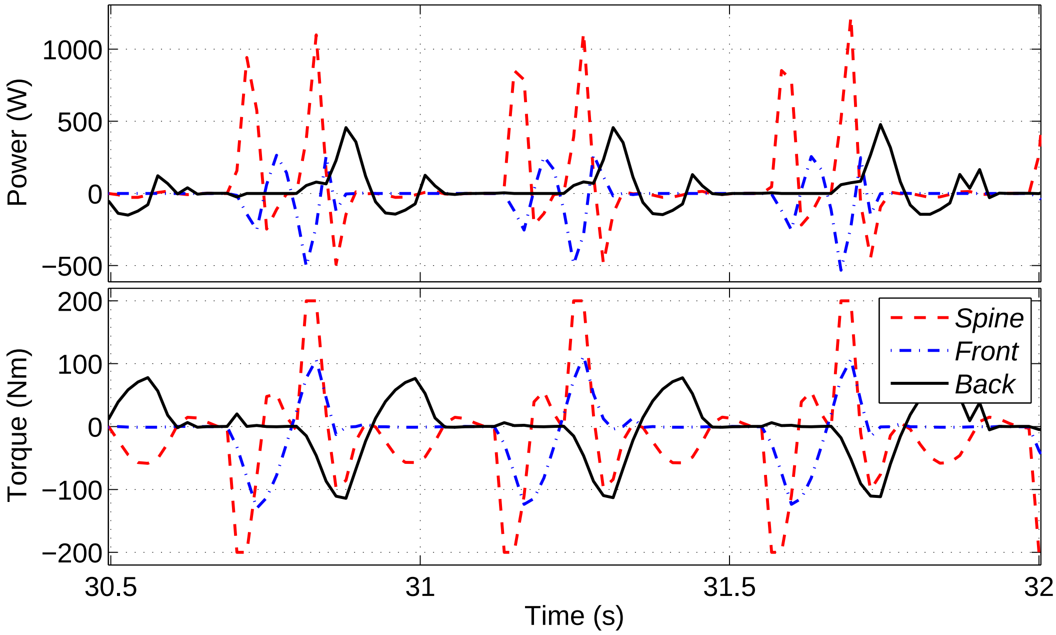

Figure 3.17: Power consumption (top) and torque output (bottom) of hip and spine actuators. Red dashed line represents the spine motor, blue dotted line front hip motor and black line back hip motor.

3.4.1 Stride Length and Speed

Horizontal speed is our most important performance criteria, which is why we used this parameter in our cost function which was defined as $\epsilon := P/mgv^3$, where v is the average horizontal speed. If we look at the results, we see that stiff backed robot can achieve an average speed of 1.74 m/s. On the other hand, the flexible robot can run with an average speed of 2.02 $m/$ which shows a %17 of increase within a stable region during bounding.

The reason of the increase in the speed does not lay behind the used cost function only. By using a flexible body structure we managed to expand the reach of legs before touchdown events as well as bending the body to give an additional thrust similar to the natural mechanisms [20]. As a result of this behavior we succeeded in increasing the stride length of the robot which can also be seen in Figure 3.18. Figure shows that for the stiff backed model, the average stride length is approximately 0.58 m. In our model, this length is increased %48 and became 0.86 m.

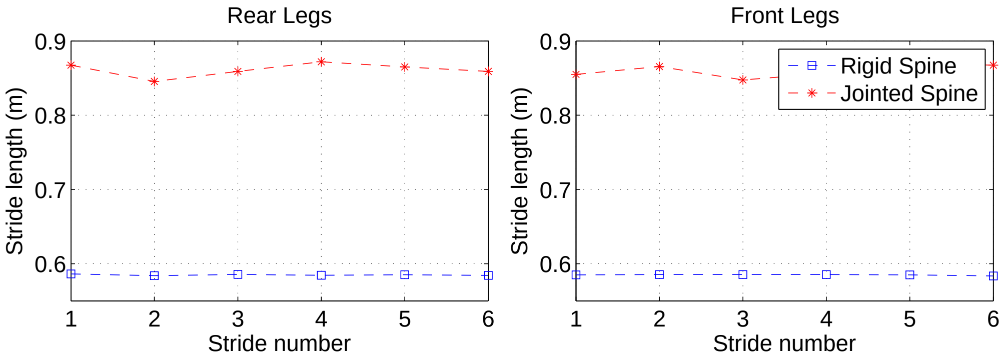

Figure 3.18: Stride lengths for the back (left plot) and front (right plot) legs for the last six bounding steps. Red stars illustrate stride lengths for the actuated spine model whereas the blue squares correspond to the stiff backed model.

One interesting notice to add here could be the change in the stride frequency. A careful eye can see the difference between our stiff back and flexible back models by comparing the duration and number of strides in Figure 3.10 and Figure 3.15. In these figures we see that the the duration of a single stride increases while the stepping frequency decreases in the flexible model. These reasonings naturally are coupled with the explanation of the increase of the stride length.

3.4.2 Hopping Height and Feet Clearance

Increase in the hopping height was another claim of our research on spinal actuation specifically on the bounding gait. When we compare the results in the previous section, we see that the maximum hopping height of the stiff backed robot is 0.755 m. The flexible model has an increase of %6 in the maximum hopping with the ability to reach up to 0.8 m. Although this rate may seem small, the important difference is between the feet clearance results. We observe that there is a %150 of increase for the back leg which has changed from 0.02 m to 0.05 m. A larger increase happens for the front leg, at a rate of %350 which enabled the robot to raise the leg from 0.04 m to 0.18 m.

The basic idea behind this increase is again related with the flexible body poses of the robot which is adjusted by the actuated spine. In the flexible bounding gait controller, we are bending our robot body outwards in order to take the convex pose. While the back leg is still on the ground in the back stance phase, the front leg is controlled to its touchdown angle position. Adding the spinal angle in the convex pose to the touchdown angle of the front leg, we obtain a larger space under the toe. Same logic increases the clearance of the back leg too. In addition to the auxiliary power provided to the system from the spine motor and the increase of leg clearances, the flexible robot jumps higher than the stiff backed robot. So as a result, the spinal actuation indirectly increases the hopping height by enabling a higher lift for the legs.

3.4.3 Additional Thrust

A question that needs to be asked is whether spinal actuation gives additional thrust to the system during stance phases. The observations which raises this question is the poses of the body changing with respect to the spinal angle. In the flexible bounding gait, there is an important transition between body poses when the robot is on the ground. When the double flight phase ends with the front leg touchdown event, the spine angle is controlled to position itself to concave angle β_{cv} which bends the body inwards to the ground direction. When the back leg touches the ground, the spine changes its angle towards the opposite direction to form a concave pose. While the body pose is changing the front leg leaves the ground leaving the back leg in its compression phase. According to the geometry of the body at this state, the torques produced by the spine motor increases the amount of force acting on the back leg towards the ground. This additional force on the leg spring increases the amount of compression of the legs, which acts like an auxiliary thrust for the whole mechanism.

Figure 3.19 shows the amount of reaction forces acting on the ground due to the compression of the legs. The first graph compares these amounts in the front legs of stiff backed and flexible backed models. We see that there is a small increase in the amount around %11. However this change is more striking in the back legs; while the maximum force is 282.5 Nm in the stiff backed model, it is

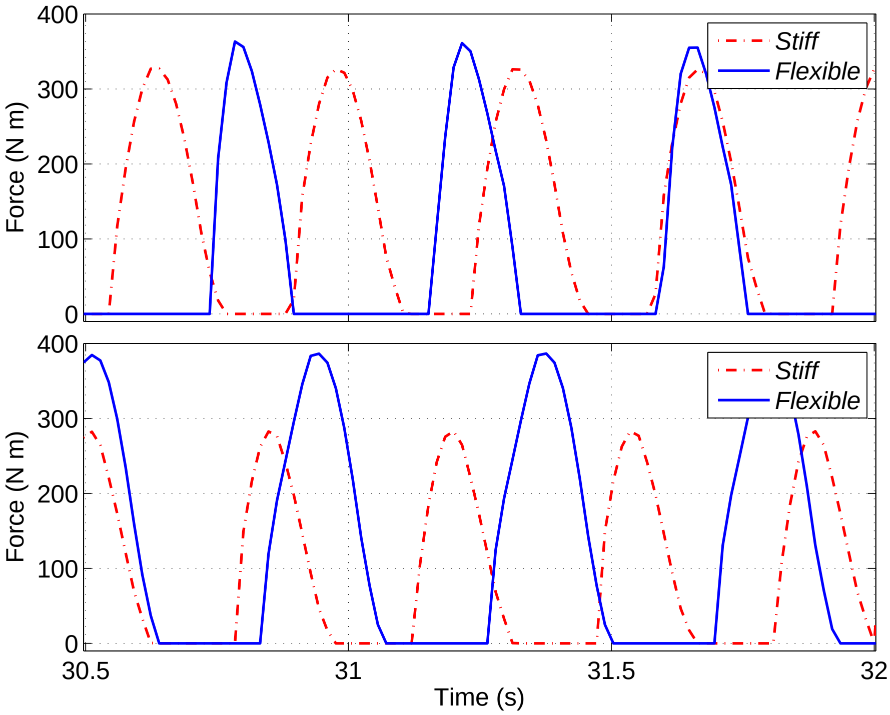

Figure 3.19: Leg springs' reaction forces on the ground comparing the stiff backed and flexible backed model. The front legs (top) and back legs (bottom) are compared.

386.3 Nm in the flexible model. Although we observe an increase of % 36.7 in the maximum forces due the spring compression, it is not clear whether this increase is due to only spinal actuation. In the previous section where we discussed the hopping height, we have observed an increase in the maximum jumping height of the robot. Regarding this increase, we cannot conclude that the changes in the spring compression is solely because of spinal thrust. However, we can reason that, both of these results are the evidence for an additional power provided by the spine joint and the auxiliary thrust and higher jumping heights are coupled observations that support each other.

3.4.4 Torque Output

Another dimension of the performance comparison is the amount of torque produced by the actuators we have used. If we look at the results of the stiff backed robot model, we see that neither of the hip joints exceeds the absolute level of 90 Nm. However, the hip joints of the flexible model produced up to 110 Nm of torque. If compare the results by leg pairs, we see that there is an increase of %22 for the front leg motors and %100 for the back leg motors in the flexible robot. This could be the result of both the additional thrust provided by spinal actuation and the increased hopping height which results in greater forces on the legs.

Also another interesting observation shows us that in both of the models, the torque produced in front legs are more than the one produced in back legs. Due to the nature of the bounding gait, the double flight phase ends with the front leg touchdown event. So the front leg will be compensating the impact forces during first touch down in a stride. Considering the dynamics and the inertia of the robot body during flight, these impact forces will be affecting the front hip joint as well as the hip actuator which will be trying to adjust the leg angle. This front motor facing a larger resistive force compared to the back motor is naturally producing more torque to maintain its task without losing the balance of the robot.

If we look at the spine motor's torque output, we observe that it is saturated at the predefined limit. In tests where we varied this saturation level, we observed that spine motor can be used to produce up to 800 Nm to enhance the performance of the bounding gait. Along with this higher limit, we achieved faster horizontal speeds close to 3.2 m/s however, none of these saturation limits were practical; therefore we selected 200 Nm as our realistic level. One important thing to not here is that our flexible robot structure and bounding gait controller enables higher horizontal speeds given the required actuation power. Even in the tests where we increased the saturation limit up to 800 Nm, the optimum parameter set for the stiff backed bounding did not change. It can also be seen from the torque output figures in the previous section that none of the legs exceed 90 Nm level which shows us that the stiff body structure cannot outperform its current speed even with higher torque limits.

3.4.5 Power Consumption and Negative Work

The best overview where we can discuss the effects of spinal actuation is shown in Table 3.6. This table shows the specific resistance values of both bounding models with different modes of the power calculated. The first column with average power is calculated as $\epsilon := P/mgv$ meaning that both positive and negative values of power consumed at motors are considered. The other type of specific resistance is calculated as $\epsilon := |P|/mgv^3$ with only the absolute values of power is used. The first row has the information about the stiff backed bounding where the other two rows belong to flexible backed bounding. We have calculated the corresponding specific resistance values of the flexible backed bounding in two ways where we included all motors and excluded the spine motor to detect its own power consumption.

Table 3.6: Specific resistance values for bounding behaviors.

Model	ϵ **with Avg. Power**	ϵ **with Avg. Abs. Power**
stiff	0.127	0.309
flexible total	0.227	0.755
flexible legs	0.024	0.371

If we compare the rows of the last column where we have the specific resistance calculated with the absolute values of the motor powers, we see that all cases of flexible bounding results are higher than the stiff backed bounding. However the most important fact here is the power consumption of the spine motor itself. When the rows of flexible backed bounding is subtracted, we get $\epsilon = 0.384$ for the spine motor, which is, by itself, is larger than the sum of both motors in both of the models. In the previous section where we showed the simulation results, spine motor was consuming up to 1200 W. This result we get from the specific resistance table also approves of this situation and shows us the role of the spine in the overall gait. We see that spine motor faces with a great amount of force during the gait and consumes a lot of power both to compensate it and give the robot auxiliary power it needs to continue the bounding.

A careful eye can notice that the peaks of spine motor power consumption happen in front leg and double stance phases. This is reasonable because the front leg stance phase is when the front leg faces the impact forces and these forces are transferred onto the spine joint through body connection points. The spine motor which is trying to maintain a specific angle, receives an external disturbing force in this state and the low level controller commands the motor to restore its position back to desired target angle. The situation in the double stance phase is different than the impact force reaction. In this phase, robot body is controlled to change its pose in the reverse direction, therefore the spine motor consumes a lot of power to turn the heavy body parts into the desired angle. Other than these states we observe the spine motor to follow a similar pattern with the legs.

The other important observation we make with this table is the amount of negative work existing in both models. If we compare the specific resistance values of average power with absolute power, we see that theses values are much higher in the absolute power based functions. The reason for that is the amount of negative work done by the motors on the robots. These actuators are asked to position the joints to very different angles where actuators need to be turned in opposite directions. As the energy needed to turn a joint in one way is not stored in any kind of passive mechanism, this lost energy cannot be used when the joint is needed to turn to its previous position.

We see that the rate of negative work in flexible bounding is more than in the stiff backed version. This is mainly because there are three motors in the former one including the spine motor which is shown to consume a large amount of power. We know that our joints do not consist of any passive compliant mechanisms like torsional springs. These results tell us the need of some passive mechanisms like these springs to store the energy of motors during actuation and release the energy back to the system when needed. If such a mechanism is used the specific resistance values will be closer to each other meaning that the system can store energy and transfer it to necessary parts when available.

Chapter 4

Mathematical Model of Flexible Spine

In this chapter, we extend the contribution of our thesis and mathematically derive the kinematics and dynamics of a flexible back quadruped robot and its bounding locomotion.

4.1 Analytical Planar Model

4.1.1 System Kinematics

We start by explaining the structure of the model and derive its kinematic equations based on specified structural parameters of the robot. Figure 4.1 shows the overall structure of our flexible backed robot model together with all parameters that determine kinematic relations.

The model is based on two main coordinate frames; W representing the inertial world frame and B representing the body frame. For ease of understanding, parameter and point definitions will be represented with these superscript letters to differentiate whether the parameter has been defined in world coordinates or

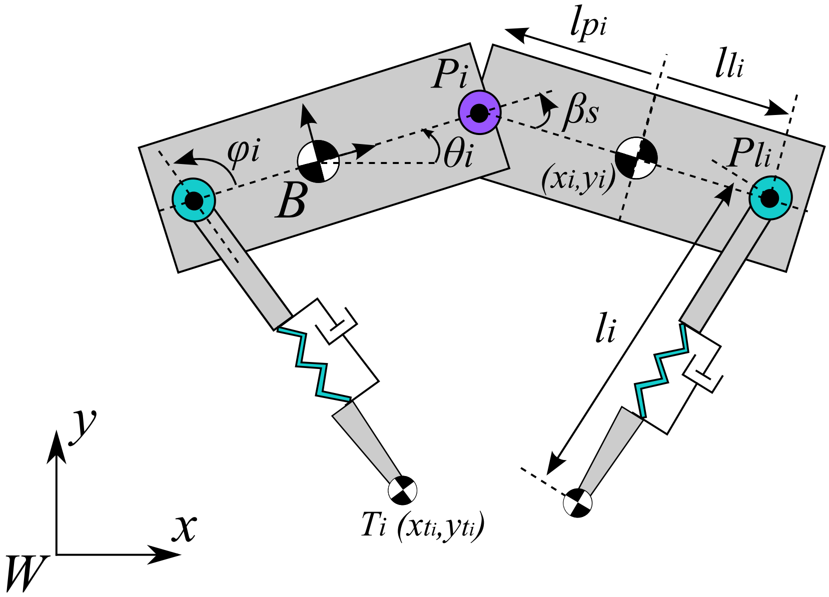

Figure 4.1: Planar quadrupedal robot model with an actuated spine joint connecting two body segments. Parameters shown on the robot define kinematic relations and properties.

body coordinates.

The robot consists of two body segments: front and back. Each of these segments are connected to a hip actuated leg on points $P_{l_i}^W$ and are connected to each other with a single spine joint denoted as P_i^W. For our kinematic formulation, this single joint is also denoted with two points; P_f^W & P_b^W, each of which is assumed to be located on corresponding body segments. Even though they are on different body segments, those points are on the same location in W, sustaining a constrained connection between body parts throughout locomotion.

Both leg joint points $P_{l_i}^B$ and spinal joint points P_i^B are vertically aligned with corresponding bodies' center of mass (COM) point, represented with the parameter pairs of (x_i, y_i). While parameters l_{l_i} and l_{p_i} represent horizontal offset between corresponding joint and COM points, l_{b_i} and h_{b_i} represent absolute values of body length and height. In body coordinates B, we represent these leg joint points using offset values shown as below:

$$P_{l_i}^B = \begin{bmatrix} l_{l_i} \\ 0 \end{bmatrix} \tag{4.1}$$

$$P_i^B = \begin{bmatrix} l_{p_i} \\ 0 \end{bmatrix} \tag{4.2}$$

As we have defined our joints vertically aligned with center of mass points, the second row of each vector defined above is 0.

The two body angles θ_i are defined as the counter-clockwise angle between the world x axis and the body x axis of $body_i$. Likewise, leg angles φ_i are defined as the counter-clockwise angles between the x axis of $body_i$ and the y axis of leg_i in B.

Table 4.1: Kinematic parameters for body segments.

Param.	**Value**
x_i	the x position of COM of body i
y_i	the y position of COM of body i
θ_i	the body angle w.r.t global x axis
l_{b_i}	the length of body i
h_{b_i}	the height of body i
l_{l_i}	the horizontal distance btw. COM_i & leg i joint $P_{l_i}^B$
l_{p_i}	the horizontal distance btw. COM_i & spinal joint P_i^B
β_s	the spinal angle formed between two body parts
$i \in \{1, 2\}$, 1: front, 2: back	

Table 4.2: Kinematic parameters for leg parts.

Param.	**Value**
x_{t_i}	the x position of COM of toe of leg i
y_{t_i}	the y position of COM of toe of leg i
φ_i	the leg angle w.r.t body i x axis
l_i	the current length of leg i
l_i^0	the rest length of leg i
$i \in \{1, 2\}$, 1: front, 2: back	

Table 4.1 and Table 4.2 show the names and descriptions of kinematic parameters used in our mathematical derivations and the equations for motion of the

flexible backed robot model. By using the parameters in these tables, a robot configuration vector q is formed as

$$q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_{t_1}, y_{t_1}, x_{t_2}, y_{t_2}] \tag{4.3}$$

Parameters in this vector define the configuration of the system at any time given during the locomotion. By using these parameters, any point on the robot and relations between these points and other parts of the robot can be derived. Depending on this vector q, system dependent variables such as $P_{l_i}^W$ and P_i^W, and further kinematic relationships such as β_s, l_i and φ_i shown in Figure 4.1 and Figure 4.2 can be found.

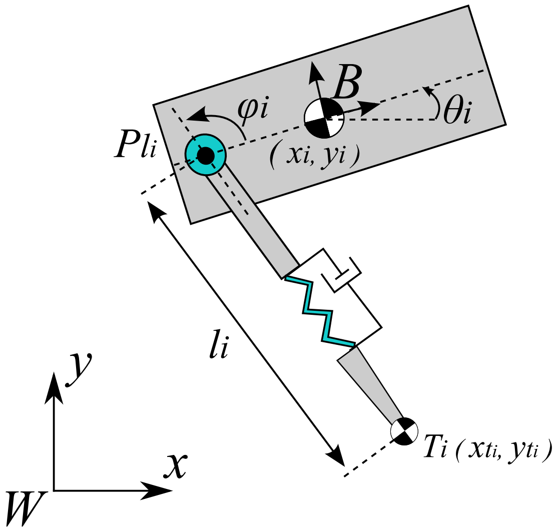

Figure 4.2: Kinematic relationships on a single leg.

We define the rotation matrix R_Θ and its first and second derivatives as;

$$R_\Theta := \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix}$$

$$\dot{R}_\Theta := \begin{bmatrix} -\sin\Theta & -\cos\Theta \\ \cos\Theta & -\sin\Theta \end{bmatrix} \dot{\Theta}$$

$$\ddot{R}_\Theta := \begin{bmatrix} -\cos\Theta & \sin\Theta \\ -\sin\Theta & -\cos\Theta \end{bmatrix} \dot{\Theta}^2 + \begin{bmatrix} -\sin\Theta & -\cos\Theta \\ \cos\Theta & -\sin\Theta \end{bmatrix} \ddot{\Theta}$$

Based on the assumptions explained above, the kinematic formulation of the planar robot model is explained in the following equations. First of all, the leg and spine joints on the robot body, $P_{l_i}^{B}$ and P_i^{B}, and their corresponding positions in the world frame are found as

$$P_{l_i}^{W} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + R_{\theta_i} P_{l_i}^{B} \qquad \dot{P}_{l_i}^{W} = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} + \left(\dot{R_{\theta_i} P_{l_i}^{B}}\right) \tag{4.4}$$

$$P_i^{W} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + R_{\theta_i} P_i^{B} \qquad \dot{P}_i^{W} = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} + (\dot{R_{\theta_i} P_i^{B}}) \quad . \tag{4.5}$$

The spinal angle β_s is defined to be the positive angle difference between the body angles of both robot body parts. The spinal angle velocity is also defined in the same manner as follows

$$\begin{aligned} \beta_s &= \theta_1 - \theta_2 & (4.6) \\ \dot{\beta}_s &= \dot{\theta}_1 - \dot{\theta}_2 \quad , & (4.7) \end{aligned}$$

and the leg length is defined as the vector formed between the toe and leg joint points on the robot as

$$\begin{aligned} l_i &= \sqrt{({P_{l_{iy}}}^{W} - y_{t_i})^2 + ({P_{l_{ix}}}^{W} - x_{t_i})^2} & (4.8) \\ \dot{l}_i &= \frac{({P_{l_{iy}}}^{W} - y_{t_i})(\dot{P_{l_{iy}}}^{W} - \dot{y_{t_i}}) + ({P_{l_{ix}}}^{W} - x_{t_i})(\dot{P_{l_{ix}}}^{W} - \dot{x_{t_i}})}{l_i} \quad . & (4.9) \end{aligned}$$

The leg angle φ_i is the positive angle defined between the horizontal world axis and the vector defined between leg joint point and toe points. The body angle of the robot part is also involved to complete the leg angle definition as follows:

$$\begin{aligned} \varphi_i &= \arctan\left(\frac{{P_{l_{iy}}}^{W} - y_{t_i}}{{P_{l_{ix}}}^{W} - x_{t_i}}\right) - \theta_i & (4.10) \\ \dot{\varphi}_i &= \frac{({P_{l_{ix}}}^{W} - x_{t_i})(\dot{P_{l_{iy}}}^{W} - \dot{y_{t_i}}) - ({P_{l_{iy}}}^{W} - y_{t_i})(\dot{P_{l_{ix}}}^{W} - \dot{x_{t_i}})}{l_i^2} - \dot{\theta}_i \quad . & (4.11) \end{aligned}$$

4.1.2 Force Equilibrium

In this section, active forces on the legs and body segments will be presented. Based on the assumption that the legs have negligible mass compared to the body, the leg forces differ in stance and flight phases. Consequently, total leg forces during flight since they are assumed to be massless.

In order to distinguish these two phases, a binary value called the *stance chart*, denoted by s_i, is defined. The value for this variable will be 1 if the corresponding leg is in the stance phase and 0 when it is in flight phase.

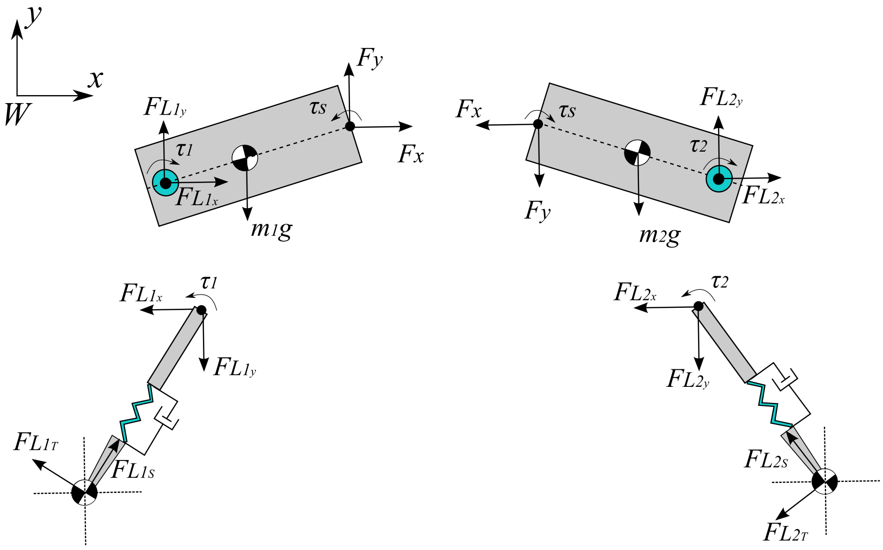

Figure 4.3: Free body diagram of the robot showing relevant forces and torques.

The model is symmetrically structured for both body segments. Therefore, derivations for a single leg and toe will generate to both legs. The leg model is an instance of the SLIP model, which expresses the whole leg-toe system as a massless spring-damper attached to a point mass at the toe. In this model, the spring-damper is still massless, but is accompanied by a small toe mass at the end of the leg.

The definition of forces in the system is shown in Figure 4.3 and their descriptions are given in Table 4.3. As stated before, the leg is defined to be a massless spring-damper system. Consequently, forces related with the spring-damper and

Table 4.3: Parameter names and definitions for forces acting on joints.

Param.	**Value**
(F_x, F_y)	the forces acting on spine joint
$(F_{l_{i_x}}, F_{l_{i_y}})$	the forces acting on hip joint i
$F_{l_{i_T}}$	the force produced by hip torque τ_i
$F_{l_{i_S}}$	the force produced by spring-damper in leg i
$i \in \{1, 2\}$, 1: front, 2: back	

the torque applied on the leg can be derived as

$$F_{l_{i_S}} = -k_i(l_i - {l_i}^0) - d_i\dot{l_i} \tag{4.12}$$

$$F_{l_{i_T}} = \tau_i / l_i \quad . \tag{4.13}$$

Depending on the assumption that the leg is massless, the leg itself does not have any dynamics. Therefore, it only acts as a means of transferring forces created by the spring-damper system and the torque applied on the leg joint. This can also be seen in Figure 4.3. The corresponding force balance equation is shown below.

$$\begin{bmatrix} F_{l_{i_x}} \\ F_{l_{i_y}} \end{bmatrix} = R_{(\theta_i + \varphi_i)} \begin{bmatrix} F_{l_{i_S}} \\ F_{l_{i_T}} \end{bmatrix} \tag{4.14}$$

4.1.3 System Dynamics

In this section, the dynamics of the entire system is investigated. Based on the system configuration vector defined in Equation 4.3, the dynamics of the system computes the second derivatives of these state variables. However, the spinal joint has an associated unknown force vector which must also be solved along with the system dynamics. The components of this vector are shown in Figure 4.3 as F_x and F_y.

The state of our system, including velocities, can be defined by the vector

$$S = [x_1, y_1, \theta_1, \dot{x_1}, \dot{y_1}, \dot{\theta_1}, x_2, y_2, \theta_2, \dot{x_2}, \dot{y_2}, \dot{\theta_2}, x_{t_1}, y_{t_1}, \dot{x_{t_1}}, \dot{y_{t_1}}, x_{t_2}, y_{t_2}, \dot{x_{t_2}}, \dot{y_{t_2}}] \quad .$$

In order to find second order differential equations in the following sections, the derivative of this state vector must be defined as

$$\dot{S} = [\dot{x_1}, \dot{y_1}, \dot{\theta_1}, \ddot{x_1}, \ddot{y_1}, \ddot{\theta_1}, \dot{x_2}, \dot{y_2}, \dot{\theta_2}, \ddot{x_2}, \ddot{y_2}, \ddot{\theta_2}, \dot{x_{t_1}}, \dot{y_{t_1}}, \ddot{x_{t_1}}, \ddot{y_{t_1}}, \dot{x_{t_2}}, \dot{y_{t_2}}, \ddot{x_{t_2}}, \ddot{y_{t_2}}] \quad .$$

The first derivatives, or the velocities, of the states given in $\dot{S}$ can be directly taken from the state vector itself. But the second derivatives, or accelerations, must be calculated through dynamical equations. However, in addition to the state vector components, the spinal joint forces F_x and F_y also need to be calculated. Including these forces, we can form the the unknown dynamics vector, denoted with U as

$$U = [\ddot{x_1}, \ddot{y_1}, \ddot{\theta_1}, \ddot{x_2}, \ddot{y_2}, \ddot{\theta_2}, \ddot{x_{t_1}}, \ddot{y_{t_1}}, \ddot{x_{t_2}}, \ddot{y_{t_2}}, F_x, F_y] \quad .$$

4.1.3.1 Toe Dynamics

At the beginning of this section, we explained that toes in our model are assumed to be fixed on the ground during stance. In order to do that, we use the stance chart variable s_i in the equations, yielding the dynamics of toes. This variable is set in stance phase and unset in flight phase, and multiplies the force parameters with either $(1 - s_i)$ or s_i. Following this method, the dynamics equations for the toes can be derived as

$$m_{t_i} \begin{bmatrix} \ddot{x_{t_i}} \\ \ddot{y_{t_i}} \end{bmatrix} = \left(R_{(\varphi_i + \theta_i)} \begin{bmatrix} -F_{l_{i_S}} \\ -F_{l_{i_T}} \end{bmatrix} - \begin{bmatrix} 0 \\ m_{t_i} g \end{bmatrix} \right) (1 - s_i) \tag{4.15}$$

$$I_{t_i} \ddot{\varphi}_i = \tau_i \quad . \tag{4.16}$$

4.1.3.2 Body Dynamics

The front and back body segments of the robot are symmetric and identical in terms of structure and control. They only differ by the direction of spine joint forces and the torque produced by the spine motor. Considering these differences, the dynamical equations of each body segment can be derived uniformly. Note that forces and torques related with hip joints are only effective on the robot while the legs are in stance phase. As mentioned before, our legs as massless and the toes have relatively small masses with respect to body. The dynamics of these particles dynamics under flight conditions do not affect the body segments as opposed to their effects in the stance phase. As such, the dynamics for the body segments are given by

$$\begin{aligned} m_{b_1}\begin{bmatrix}\ddot{x}_1\\ \ddot{y}_1\end{bmatrix} &= \begin{bmatrix}F_{l_{1x}}\\ F_{l_{1y}}\end{bmatrix}s_1+\begin{bmatrix}F_x\\ F_y-m_{b_1}g\end{bmatrix} && (4.17)\\ I_{b_1}\ddot{\theta}_1 &= -\tau_1 s_1+\tau_s+\left(\left(R_{\theta_1}P^B_{l_1}\right)\times\begin{bmatrix}F_{l_{1x}}\\ F_{l_{1y}}\end{bmatrix}\right)s_1\\ &\quad+\left(R_{\theta_1}P^B_1\right)\times\begin{bmatrix}F_x\\ F_y\end{bmatrix} && (4.18)\end{aligned}$$

$$\begin{aligned} m_{b_2}\begin{bmatrix}\ddot{x}_2\\ \ddot{y}_2\end{bmatrix} &= \begin{bmatrix}F_{l_{2x}}\\ F_{l_{2y}}\end{bmatrix}s_2+\begin{bmatrix}-F_x\\ -F_y-m_{b_2}g\end{bmatrix} && (4.19)\\ I_{b_2}\ddot{\theta}_2 &= -\tau_2 s_2-\tau_s+\left(\left(R_{\theta_2}P^B_{l_2}\right)\times\begin{bmatrix}F_{l_{2x}}\\ F_{l_{2y}}\end{bmatrix}\right)s_2\\ &\quad+\left(R_{\theta_2}P^B_2\right)\times\begin{bmatrix}-F_x\\ -F_y\end{bmatrix}\quad . && (4.20)\end{aligned}$$

4.1.3.3 Spine Joint Constraints

The equations associated with the toes and the body segments of the robot are sufficient to solve the dynamics of the system. However, because of the spinal joint and the associated unknown force vector, further constraints must be investigated and solved. As the robot model consists of two separate body segments connected with a joint in the middle, defining the constraint on this joint will suffice.

Although there are two definitions for the spine joint; P_1 and P_2 for front and back segments respectively, these two points actually must correspond to the same body position and therefore they must have the same dynamics for the robot to keep it connected. The definitions of these points were already given in Section 4.1.1. Following constraint is added with respect to the acceleration of these points as

$$\begin{aligned}
\ddot{P_1^W} &= \ddot{P_2^W} && (4.21)\\
\begin{bmatrix} \ddot{x}_1 \\ \ddot{y}_1 \end{bmatrix} + (\ddot{R_{\theta_1} P_1^B}) &= \begin{bmatrix} \ddot{x}_2 \\ \ddot{y}_2 \end{bmatrix} + (\ddot{R_{\theta_2} P_2^B}) && (4.22)\\
\begin{matrix} \begin{bmatrix} \ddot{x}_1 \\ \ddot{y}_1 \end{bmatrix} + \ddot{R}_{\theta_1} P_1^B \\ +2\dot{R}_{\theta_1}\dot{P}_1^B + R_{\theta_1}\ddot{P}_1^B \end{matrix} &= \begin{matrix} \begin{bmatrix} \ddot{x}_2 \\ \ddot{y}_2 \end{bmatrix} + \ddot{R}_{\theta_2} P_2^B \\ + 2\dot{R}_{\theta_2}\dot{P}_2^B + R_{\theta_2}\ddot{P}_2^B \end{matrix} \quad . && (4.23)
\end{aligned}$$

As the points P_i^B are defined on rigid bodies, both first and second derivatives of these points will yield 0. Therefore, the last equation could be simplified as

$$\begin{bmatrix} \ddot{x}_1 \\ \ddot{y}_1 \end{bmatrix} + \ddot{R}_{\theta_1} P_1^B = \begin{bmatrix} \ddot{x}_2 \\ \ddot{y}_2 \end{bmatrix} + \ddot{R}_{\theta_2} P_2^B \quad . \tag{4.24}$$

4.1.4 Overview of the Equations of Motion

With the addition of this last equation to the system dynamics, twelve second order differential equations for twelve unknowns in the vector U are obtained. The following equations summarize the solution to the whole system:

$$m_{b_1}\ddot{x}_1 = F_{l_{1_x}} s_1 + F_x \tag{4.25}$$

$$m_{b_1}\ddot{y}_1 = F_{l_{1_y}} s_1 + F_y - m_{b_1} g \tag{4.26}$$

$$\begin{aligned} I_{b_1}\ddot{\theta}_1 &= -\tau_1 s_1 + \tau_s + \left(l_{l_1}\cos\theta_1 F_{l_{1_y}} - l_{l_1}\sin\theta_1 F_{l_{1_x}}\right) s_1 \\ &\quad + \left(l_{p_1}\cos\theta_1 F_y - l_{p_1}\sin\theta_1 F_x\right) \end{aligned} \tag{4.27}$$

$$m_{b_2}\ddot{x}_2 = F_{l_{2_x}} s_2 - F_x \tag{4.28}$$

$$m_{b_2}\ddot{y}_2 = F_{l_{1_2}} s_2 - F_y - m_{b_2} g \tag{4.29}$$

$$\begin{aligned} I_{b_2}\ddot{\theta}_2 &= -\tau_2 s_2 - \tau_s + \left(l_{l_2}\cos\theta_2 F_{l_{2_y}} - l_{l_2}\sin\theta_2 F_{l_{2_x}}\right) s_2 \\ &\quad + \left(-l_{p_2}\cos\theta_2 F_y + l_{p_2}\sin\theta_2 F_x\right) \end{aligned} \tag{4.30}$$

$$\begin{aligned} m_{t_1}\ddot{x}_{t_1} &= \big(-F_{l_{1_S}}\cos(\varphi_1+\theta_1) + F_{l_{1_T}}\sin(\varphi_1+\theta_1)\big) \\ &\quad (1-s_1) \end{aligned} \tag{4.31}$$

$$\begin{aligned} m_{t_1}\ddot{y}_{t_1} &= \big(-F_{l_{1_S}}\sin(\varphi_1+\theta_1) - F_{l_{1_T}}\cos(\varphi_1+\theta_1) - m_{t_1} g\big) \\ &\quad (1-s_1) \end{aligned} \tag{4.32}$$

$$I_{t_1}\ddot{\varphi}_1 = \tau_1 \tag{4.33}$$

$$\begin{aligned} m_{t_2}\ddot{x}_{t_2} &= \big(-F_{l_{2_S}}\cos(\varphi_2+\theta_2) + F_{l_{2_T}}\sin(\varphi_2+\theta_2)\big) \\ &\quad (1-s_2) \end{aligned} \tag{4.34}$$

$$\begin{aligned} m_{t_2}\ddot{y}_{t_2} &= \big(-F_{l_{2_S}}\sin(\varphi_2+\theta_2) - F_{l_{2_T}}\cos(\varphi_2+\theta_2) - m_{t_2} g\big) \\ &\quad (1-s_2) \end{aligned} \tag{4.35}$$

$$I_{t_2}\ddot{\varphi}_2 = \tau_2 \tag{4.36}$$

$$\begin{matrix} \ddot{x}_1 + (-l_{p_1}\cos\theta_1)\dot{\theta}_1^{\,2} \\ +(-l_{p_1}\sin\theta_1)\ddot{\theta}_1 \end{matrix} = \begin{matrix} \ddot{x}_2 + (-l_{p_2}\cos\theta_2)\dot{\theta}_2^{\,2} \\ +(-l_{p_2}\sin\theta_2)\ddot{\theta}_2 \end{matrix} \tag{4.37}$$

$$\begin{matrix} \ddot{y}_1 + (-l_{p_1}\sin\theta_1)\dot{\theta}_1^{\,2} \\ +(l_{p_1}\cos\theta_1)\ddot{\theta}_1 \end{matrix} = \begin{matrix} \ddot{y}_2 + (-l_{p_2}\sin\theta_2)\dot{\theta}_2^{\,2} \\ +(l_{p_2}\cos\theta_2)\ddot{\theta}_2 \end{matrix} \quad . \tag{4.38}$$

As the main idea behind the solution is to find the system dynamics vector U, a simple linear formulation can be used. If all the equations above are rewritten so that the members of the U vector and their coefficients are on the left hand side and the remaining known values are put on the right hand side of the equations, a simple matrix equation can capture the system dynamics vector. Assuming that all the coefficients of the unknown variables are put into matrix M and the known values are put into matrix K, the following equation will be the summarization of the intended linear operation:

$$MU^T = K \text{ and } U^T = KM^{-1} \quad , \tag{4.39}$$

where, the matrices M and K take the following form:

$$K = \begin{bmatrix} F_{l_{1_x}} s_1 \\ F_{l_{1_y}} s_1 - m_{b_1} g \\ -\tau_1 s_1 + \tau_s + \Big(l_{l_1} \cos\theta_1 F_{l_{1_y}} - l_{l_1} \sin\theta_1 F_{l_{1_x}} \Big) s_1 \\ F_{l_{2_x}} s_2 \\ F_{l_{1_2}} s_2 - m_{b_2} g \\ -\tau_2 s_2 - \tau_s + \Big(l_{l_2} \cos\theta_2 F_{l_{2_y}} - l_{l_2} \sin\theta_2 F_{l_{2_x}} \Big) s_2 \\ \big(-F_{l_{1_S}} \cos(\varphi_1 + \theta_1) + F_{l_{1_T}} \sin(\varphi_1 + \theta_1) \big) (1 - s_1) \\ \big(-F_{l_{1_S}} \sin(\varphi_1 + \theta_1) - F_{l_{1_T}} \cos(\varphi_1 + \theta_1) - m_{t_1} g \big) (1 - s_1) \\ \big(-F_{l_{2_S}} \cos(\varphi_2 + \theta_2) + F_{l_{2_T}} \sin(\varphi_2 + \theta_2) \big) (1 - s_2) \\ \big(-F_{l_{2_S}} \sin(\varphi_2 + \theta_2) - F_{l_{2_T}} \cos(\varphi_2 + \theta_2) - m_{t_2} g \big) (1 - s_2) \\ (l_{p_1} \cos\theta_1) \dot{\theta_1}^2 - (l_{p_2} \cos\theta_2) \dot{\theta_2}^2 \\ (l_{p_1} \sin\theta_1) \dot{\theta_1}^2 - (l_{p_2} \sin\theta_2) \dot{\theta_2}^2 \end{bmatrix}$$

$$M = \begin{bmatrix}
m_{b_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & m_{b_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & I_{b_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & l_{p_1}\sin\theta_1 & -l_{p_1}\cos\theta_1 \\
0 & 0 & 0 & m_{b_2} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & m_{b_2} & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & I_{b_1} & 0 & 0 & 0 & 0 & -l_{p_2}\sin\theta_2 & l_{p_2}\cos\theta_2 \\
0 & 0 & 0 & 0 & 0 & 0 & m_{t_1} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & m_{t_1} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_{t_2} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_{t_2} & 0 & 0 \\
1 & 0 & -l_{p_1}\sin\theta_1 & -1 & 0 & l_{p_2}\sin\theta_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & l_{p_1}\cos\theta_1 & 0 & -1 & -l_{p_2}\cos\theta_2 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

4.2 Bounding Gait Controller

For the planar quadruped detailed in this chapter, we used the same bounding gait model we have defined in Chapter 3. As a reminder, the bounding gait we implement consists of four consecutive phases that are separated from each other by conditions on state vector components. Along with these conditions, the stance chart variable s_i plays the most important role, as it is an indicator of the state of each leg.

The event detection for leg touchdown and liftoff was assumed to be handled by pressure sensors at the tip of each toe in the previous model. When the second order dynamic equations given in Section 4.1.4 are integrated throughout a period, we can find the velocities of toe points at a time instance t. By looking at the value of $\dot{y_{t_i}}(t)$ we can understand whether the toe is in flight or stance phase. In a similar fashion, we can detect the apex point of the center of mass of the whole system, denoted as COM_S, as

$$\begin{aligned} COM_S &= \left(\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} m_{b_1} + \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} m_{b_2} \right) / (m_{b_1} + m_{b_2}) \\ \dot{COM_S} &= \left(\begin{bmatrix} \dot{x_1} \\ \dot{y_1} \end{bmatrix} m_{b_1} + \begin{bmatrix} \dot{x_2} \\ \dot{y_2} \end{bmatrix} m_{b_2} \right) / (m_{b_1} + m_{b_2}) \quad , \end{aligned}$$

by calculating the value of the second equation. The second row of $CO\dot{M}_{system}$ will yield the vertical velocity of the robot, which can be used to detect the apex height. By using leg events, we can implement the dynamic bounding gait for this model as well. The apex height can also be used for further extensions of the system.

In this chapter, we use a PD instead of a PID controller for the control of joint motor positions. Except for the integral term in the controller, the formulation and derivation of torque outputs for this model are the same with those explained in Chapter 3. We also used the same trajectory tracking method to maintain a constant angular velocity during leg swings and body pose changes.

4.3 Extensions to the Model

Recall that the Working Model 2D simulation environment was used for our models in the Chapter 3. It was able to solve the dynamic equations by itself, without revealing mathematical details to the user. In addition to the dynamics of the robot, it also solves various dynamics such as ground and air friction, elasticity and electric charge. Despite the fact that we do not know the mathematical models of the simulation tool, we were able to use it for our previously presented simulations.

For the mathematical model we present in this chapter, we needed to include some additional features as well. In this section, we present these additional features that can also be used with our robot and bounding model in order to increase the level of realism. For instance we implemented a ground friction force model and an alternative bounding gait controller and observed their effects on the bounding performance.

4.3.1 Ground Friction

Various friction models attempt to model forces generated between two bodies in contact, in the opposite direction of motion. There can be two types of force with respect to this model, static and kinetic friction forces. Generally each type of these force is generated as a function of the normal force on the contact surface and a friction coefficient related with body material. These forces can be derived as

$$|F_s| \leq \mu_s |F_N| \qquad |F_k| = \mu_k |F_N| \quad , \tag{4.40}$$

where F_N is the normal force on the surface and μ_s, μ_k are static and kinetic friction coefficients for materials used. According to this model, a moving force F must exceed the static friction force F_s in order to move a body towards a direction. If this force is lower than the static friction force, then a body will not move. Whenever static friction force is exceeded by the moving force, a constant kinetic friction force is applied to the moving bodies as long as the normal force

F_N stays same.

In order to implement such a model, an additional event for the toes must be detected. By monitoring the horizontal speed of the toe mass, $\dot{x_{t_i}}$, a switching function to generate a friction force according to the given cases must be implemented. However, despite being realistic, embedding such a function based on an additional event detection into our dynamic equations and the integration method we use has computational and implementation costs. In addition to these, the dynamics of the toe mass needs to be changed by integrating this discrete force function into the equations given in Section 4.1.3. Therefore, instead of this model, we preferred to use a simpler first order model.

In our viscous friction model, we assumed that the friction force acts upon with the velocity of the toe instead of its acceleration. By using a single type of friction force, we can relate the velocity of the toe with the force applied on it. Because of this, we do not need to detect additional events, friction force can be initiated by the detection of already used leg touchdown events. Working in the same fashion with any kind of friction force, the force exerted on the body must exceed the total friction to initiate motion. According to the horizontal force F_x, our function $f(F_x)$ produces the output shown in the figure below.

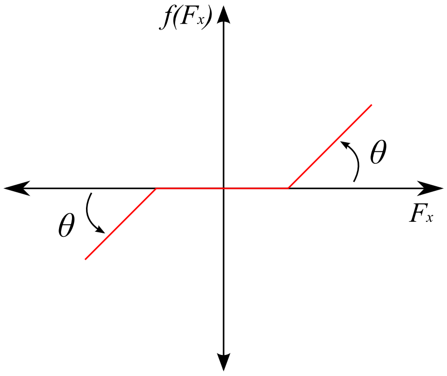

Figure 4.4: Viscous friction force as a function of total horizontal force on the toe, F_x.

In Figure 4.4, the net coming from the friction model, $f(F_x)$, is assumed to be linearly proportional to the total horizontal force F_x exerted on the toe mass. This total force F_x is defined as $F_x = F_{xt_i} - F_s$ where F_{xt_i} is the force applied on toe mass by robot dynamics, and F_s being the friction force. The function output is symmetric with respect to the vertical axis, meaning that it works for forces on positive and negative directions. It is important to note that, the net force is zero for the interval where the friction force is equal to the force applied on toe mass by system dynamics.

To integrate this viscous friction model, we need to revise the derivative of the system state vector $\dot{S}$ defined in Section 4.1.3. Since we are not using the friction force in dynamic equations deriving the accelerations, the friction function output will be applied directly on the horizontal speed of the toe masses. In order to do that, we use a switching law that depends on the state of the leg as being in flight or stance, regulating the derivative of the system state vector. While the derivation of this vector remains the same in the flight phase with the derivations above, it takes up the following form during the stance phase:

$$\dot{S} = [\dot{x_1}, \dot{y_1}, \dot{\theta}_1, \ddot{x_1}, \ddot{y_1}, \ddot{\theta}_1, \dot{x_2}, \dot{y_2}, \dot{\theta}_2, \ddot{x_2}, \ddot{y_2}, \ddot{\theta}_2, F_{x_1}K_1, 0, 0, 0, F_{x_2}K_2, 0, 0, 0] \quad .$$

The last 8 states shown in the vector belong to toe mass velocities and accelerations. We see that the first derivatives of toe masses in the horizontal direction are defined by the friction force output multiplied with a constant K_i to regulate the rate of the friction effect. We also see that the second derivatives of each toe mass is given zero to ensure that their dynamics are neglected on the ground. This type of friction model will ensure that feet on the ground will only be moving in the horizontal direction only when the maximum friction force is exceeded. The motion on vertical direction is suppressed until leg lifts off from the ground.

4.3.2 An Alternative Bounding Controller

An alternative to the flexible bounding gait controller we defined in Section 3.2.1 can be an approach similar to the stiff backed gait controller proposed by Berkemeier et al. [2], in which only one leg can be in the stance phase at a time. In order to adapt this idea to our flexible robot, four consecutive phases of the bounding can be represented as in Figure 4.5.

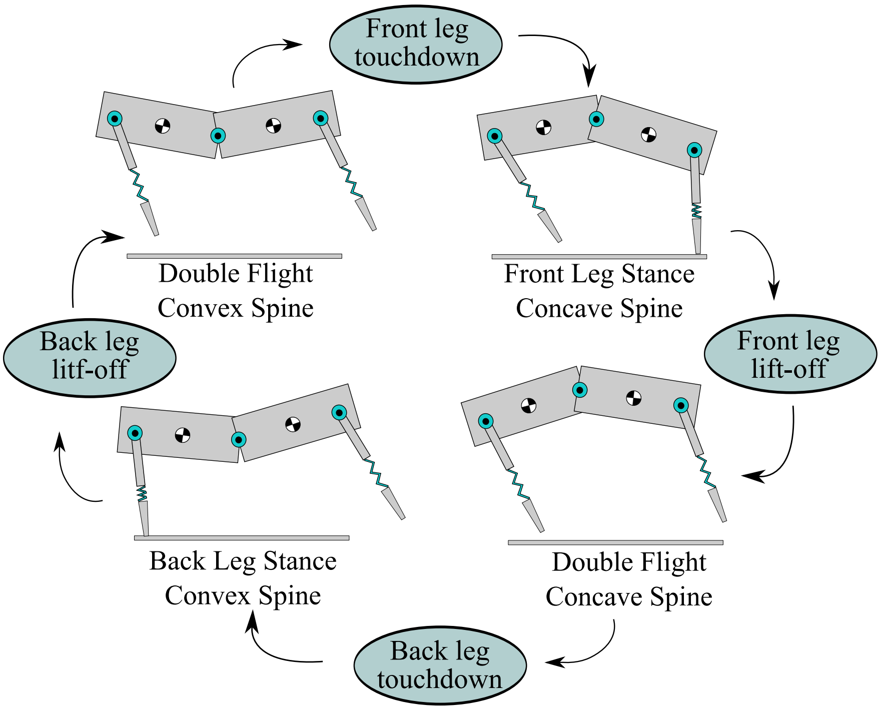

Figure 4.5: An alternative flexible bounding gait in which four consecutive phases follow each other, guaranteeing that only one leg can be in stance at a time. Therefore there are no double stance phases.

In this alternate flexible bounding gait, we can remove the double stance phase and place another double flight phase. It can also be seen from Figure 4.5 that, only one leg can be in the stance phase. The robot starts the gait in the double flight phase, where spine controls the body to form a convex shape. Similar to the previous flexible controller, this phase aims to increase the range of the front leg before it touches the ground. After the front leg touchdown occurs, the front leg is swung back until it reaches its target liftoff angle, while the spine bends inwards to form a concave pose. The event that ends this phase is one of the main differences from the previous gait controller. Recall that in the previous controller, the back leg touchdown event ends this phase and the gait enters the double stance phase. However in this alternative controller, the front leg leaves the ground before the back leg touches and the robot enters another double flight phase. But this time, the pose of the body is conserved within the concave shape in order to increase the back leg's range this time. The robot body flies in the air, while front leg is swung forward to reach its touchdown angle. Whenever the back leg touches the ground and initiates the next back leg stance phase, the spine starts to bend outward to give the robot a convex pose. The same mechanism works in this phase with the previous gait controller, as the back leg swings backwards and the spine changes the body position for additional thrust. Eventually, the back leg leaves the ground and the robot body shifts to its convex pose, completing one stride within the flexible bounding gait.

We implemented this alternative idea within our existing controller by giving our robot increased chances of maintaining a better gait. We will see in the results of our simulation that, this alternative bounding is preferred because of its additional thrust and stride length enhancements.

4.4 Simulations

4.4.1 Simulation Environment and Setup

We used MATLAB to simulate the flexible bounding gait and our mathematical model by using the dynamics equations described in Section 4.1.3. As we have defined our system with a state vector S, which involves the system configuration components and their first derivatives, we can use the associated second order differential equations to find solutions to the trajectories of the entire system spanning a given time interval.

4.4.1.1 Hybrid Dynamical Systems

Based on the nature of our system and its dynamics, we used an extension of MATLAB's *ode45* ordinary differential equation solver. This function takes in three parameters; a function in the form of $\dot{y} = f(t, y)$, an initial vector y_0 and a time span $t = \begin{bmatrix} t_0 & t_f \end{bmatrix}$. Given these three parameters, *ode45* computes the numeric integral of the function f from t_0 until t_f, using variable time steps within the given time interval. In addition to function f, user needs to specify a function describing the list of events that need to be monitored. Throughout integration, if any of these events happen, the integrator stops and finalizes the output with additional information about the event and its time of occurrence. The user can define events related to any kind of calculation within the system at a time t_i. An event can be defined with three attributes; its value, direction and terminal case. The value of the event can be a series of calculations based on system states. The direction describes the direction of zero-crossing of this value; it can be from the positive domain, negative domain or from both sides. And the terminal case defines the event as a terminal event that stops the integration.

For a hybrid dynamic system like our model, the *ode45* function needs to be invoked repeatedly as long as the events keep on happening. Therefore, we needed to modify it by adding our own controller functions. For our simulations, we used a wrapper function called hybrid dynamical system specification or *HDSS* in short, which calls *ode45* function repeatedly as events occur. This wrapper ensures that the integration continues until the absolute final time t_f, unless a critical stopping error occurs. In addition to *ode45* function parameters, we added a transition function and a system chart function to evaluate our simulation for flexible bounding.

The transition function is a system invoked after the *ode45* function terminated with respect to an event. This transition function controls the output of *ode45*, the events happened and creates a new initial vector which is basically the last state of the system before it terminated with that event. In this transition, we perform required calculations, store the system trajectory and start another integration phase with *ode45* with an initial vector created inside our transition function. With such a method, we enable our system to evaluate many events and continue integration until a given final time t_f. However, in cases when system fails with no chance of restoring, such as robot crashing on the ground or starting to run in the reverse direction, our transition function stops the integration cycle as there is no need to continue the simulation because of unrecoverable results.

The system chart function defines a vector c, which describes the discrete state of the overall system. According to our chosen bounding gait controller, a single chart of the system can be defined by the contact states of each leg. In this case, our system consists of four different charts as each leg can be in either stance or flight. This system chart function is called in many functions in the overall simulation but mainly by gait and local controllers which must decide their functionality based on the state of the legs. So we can say that our system chart function represents the touchdown sensors in a practical application of our robot.

The event function in our system finds the leg and body events required for system transitions. In total, we have five events, of which two belong to legs and the other to the body. The leg events depend on the state of each leg; if the leg is in stance the event is based on the total force on the toe, if the leg is in flight, it is based on the toe position. When the leg is in stance, the total force on the toe is generated by both the spring and the torque produced from the hip. While the normal force on the toe stays positive, it is forced to stay on the ground. Whenever the value of this normal force crosses zero from the positive domain, $F_N \leq 0$ in other words, the event is triggered. As the flight of the toe is described with respect to the vertical distance between the toe and the ground, when this distance becomes zero, the event is triggered. The body related events act similar to this idea. Two of these events check the vertical distance between the center of mass of body parts and the ground to detect whether the body hits the ground. The remaining event checks the apex time of the center of mass of the whole system. The apex is defined as $y_{\dot{COM}} = 0$ meaning that the vertical velocity of the system reaches zero.

We defined our vector field function f as the function where we calculate the results of dynamic equations shown in previous sections. As we find the system solution vector $\dot{S}$ upon the state vector S, our vector field function takes the for of $\ddot{y} = f(t, y, \dot{y})$ which makes it a second order ordinary differential equation.

4.4.1.2 Simulation Setup

By using the hybrid dynamical system defined above, we created a simulation of our robot and bounding gait in MATLAB. The physical attributes and initial conditions of the robot in this simulation are directly copied from our Working Model 2D simulations explained in the previous chapter. However, this time we did not use a model for a stiff backed robot, we only used simulations to evaluate our analytical model for the flexible backed robot. On the other hand, we used Nelder-Mead optimization method again to find an optimal set for both our model and gait controller. After finding an optimal set of parameters for the flexible bounding gait, we ran another simulation with this best parameter set and collected data to evaluate the performance.

Every bounding simulation lasts up to 30 seconds. The robot starts the simulation with a height of 0.9 m and 0.6 m/s horizontal speed. The integrator we used has an absolute maximum tolerance of 10^{-8} for overlap and integration errors.

4.4.2 Extended Optimization Set

We used the Nelder-Mead optimization method again to find the best bounding gait controller parameters for the mathematical flexible backed robot. However, unlike the previous simulations we did in Working Model 2D, we used an enlarged set for the parameters needed for the bounding gait.

$$\begin{aligned}\mathbf{p}_{fb} \;:=\; & [\varphi_{b_{td}}, \varphi_{b_{lo}}, \dot{\varphi}_b, \varphi_{f_{td}}, \varphi_{f_{lo}}, \dot{\varphi}_f, \beta_{cx}, \beta_{cv}, \dot{\beta}_{cx}, \dot{\beta}_{cv}, \\ & K_{p_b}, K_{d_b}, K_{p_f}, K_{d_f}, K_{ps}, K_{ds}]^T \end{aligned} \tag{4.41}$$

The parameter set for the flexible bounding gait is defined as $\mathbf{p}_{fb}$. It can be seen that we have enlarged this set compared to the previous model by including a separate group of PID controller gains for each leg and two angular velocity parameters for each pose of the spine. With these additional parameters, the optimization algorithm is asked to differentiate the controllers of each leg as well as the speed of spinal bending changing with respect to the body pose.

By giving an initial simplex consisting of 17 vertices, Nelder-Mead optimization algorithm found the optimum controller parameter set after 118 repetitive rounds. Table 4.4 shows the values of optimum parameters found after these rounds.

Table 4.4: Optimal gait parameters for actuated spine bounding of the mathematical model

Parameter	**Value**
K_{p_f}, K_{d_f}	(508.95, 4.77)
K_{p_b}, K_{d_b}	(479.79, 5.81)
K_{p_s}, K_{d_s}	(1631.2, 110.85)
$\varphi_{td_f}, \varphi_{lo_f}, \dot{\varphi}_f$	(0.34 rad, -0.04 rad, 5.71 rad/s)
$\varphi_{td_b}, \varphi_{lo_b}, \dot{\varphi}_b$	(0.31 rad, -0.01 rad, 3.83 rad/s)
$\beta_{cx}, \beta_{cv}, \dot{\beta}_{cx}, \dot{\beta}_{cv}$	(0.19 rad, -0.17 rad, 16.89 rad/s, 9.64rad/s)

The first two rows of the table shows the PD gains of the hip controllers. It can be seen that due to the different roles of the legs, their PD controller gains are also different. We see that front leg is slightly stronger than the back leg in means of reacting to angular error, but back leg is more sensitive to angular velocity errors. This shows us that front leg will be swung faster than the back leg, keeping its stance duration shorter. According to this behavior, front leg will be acting like a break system rather than a thrust unit. However, when we look at the back leg we see the opposite of this behavior, it is swung slower and its time in stance phase is longer. By these means, back leg acts more like a thrust spring and keeps the robot running in its balance by restoring the lost energy during the other phases of the gait. Behaviors of the legs are also supported by target angles and angular velocities. We cannot observe large differences between target angles but, the difference in angular velocities point us the same behavior.

When the parameters related with the spine motor is investigated, we observe this actuator is needed to be stronger and faster than hip motors. This is supported by the optimal PD gains shown in the third row of the table. Also we see that the angular velocity parameters differ with respect to the pose of the body. While the motor is adjusting the joint for the convex positions, it moves faster than the body is adjusted to concave pose. We know from the alternative bounding we have chosen that body changes its pose from concave to convex during back leg stance phase. This is also considered to be the thrust phase among the rest of the cycle where back leg is swung back and spine aids the compressions. With comparably a faster spinal joint in this phase, the rate and impact of the auxiliary power could be increased to fulfil the job of the spine. As the thrust impact is not necessary in front leg stance phase, observing the difference between angular velocities of the spine joint with respect to the body pose supports the idea of flexible spine's thrust role in the back leg stance phase.

4.4.3 Results

In this section we will present the data obtained after running the flexible backed robot with the optimized bounding gait controller parameters. The figures that we will present here will be extracted from the last 2 seconds of the 30 second lasting simulation, in which the robot has already maintained a stable bounding gait. These last 2 seconds also correspond to the last 5 stable strides of the gait.

Figure 4.6 shows a compilation of graphs where system's center of mass hopping height, horizontal velocity and each foot clearance along with gait phases are presented. The first graph presents the center of mass hopping height of the robot. We can see that robot's center of mass oscillates between 0.754 m and 0.788 m with a total hopping interval of 0.034 m. As the system consists of two jointed body parts, the center of mass follows a curved trajectory, however the dashed line in the graph tells us that the system has an average hopping height of 0.774 m.

The second graph in the figure shows the horizontal speed of the system center of mass. We can see that robot achieves a maximum instantaneous speed of 3.83 m/s while maintaining an average of 3.55 m/s. This average speed also corresponds to the same value of *body length / second*, which is a common measure of velocity for dynamic running robots, as the length of our robot model is 1 m.

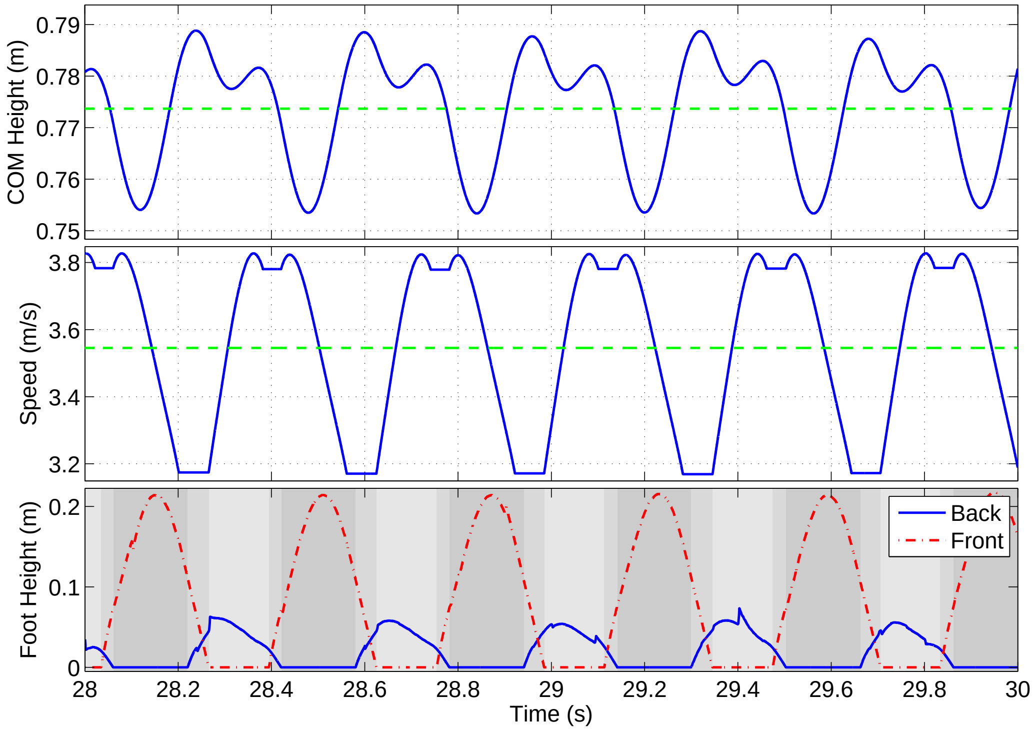

Figure 4.6: Center of mass height (top), center of mass horizontal velocity (middle) and feet clearance with gait cycle phases (bottom) of the flexible backed robot model. Green dashed lines in top two graphs show the average values.

The last graph in the figure merges the feet clearance with consecutive phases in a gait stride. The dashed lines represent the trajectory followed by the front foot and the solid line belongs to back foot. We can see that front foot can be lifted from the ground up to 0.22 m while the back foot's maximum clearance is 0.06 m. As the phases of a single stride can be defined by the phases of each foot, we can also show the gait cycle in this graph too. There are three different shaded regions on the graph which represent back leg stance, double flight and front leg stance if sorted from the darkest to the lightest grey tone. We observe that there is not double stance phase as we have defined before. Each stride lasts about 0.37 s which means that the robot has a 3 Hz running frequency. When we investigate each phase in the stride, we see that the longest phase is the back stance phase which takes approximately 0.16 s. This is followed by front leg stance phase (0.13 s), double flight in convex pose (0.05 s) and double flight in concave pose (0.03 s).

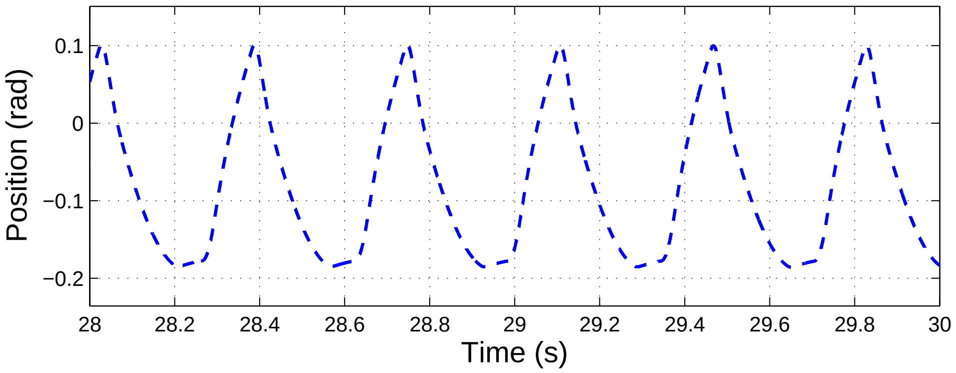

Figure 4.7: Angular trajectory of the spine during bounding motion.

Figure 4.7 shows the trajectory of the spine angle during the bounding locomotion. It can be seen that spine motor manages to adjust the spine so that the target angles for the spine, which are also shown in Table 4.4, are reached successfully.

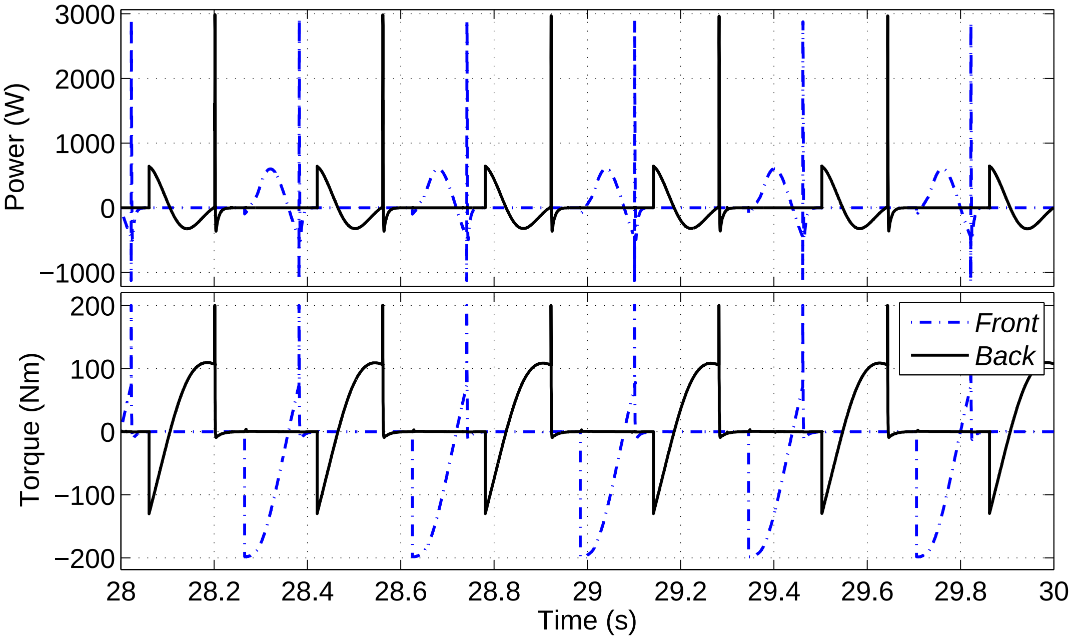

Figure 4.8: Instantaneous power consumption (top) and torque output (bottom) of hip motors during bounding gait.

Figure 4.8 shows the power consumption and torque output of hip motors used in the robot. The most striking attribute of both graphs is the torque and power peaks at particular phases of the gait. When we look at the graphs we can see that back hip motor produces a torque amount not greater than 130 Nm on its standard course. However the front hip motor produces a greater torque than the back, reaching up the levels of 198 Nm. According to these values, except from the peak levels, the maximum power consumption of back hip motor is 641 W and 601 W for the front motor. If the last graph in Figure 4.6 is investigated carefully, it can be noticed that these peaks correspond to the liftoff time instances of each leg. With respect to the fuzzy state where ground friction is disabled, the trajectory tracking system fails and PD controllers produce a larger torque than they should have. Because of that, the hips produce a peak torque saturated at 200 Nm which causes the motors to consume instantaneous power at levels of 3000 W. These peaks can be neglected as the duration of them are smaller than 0.001 s

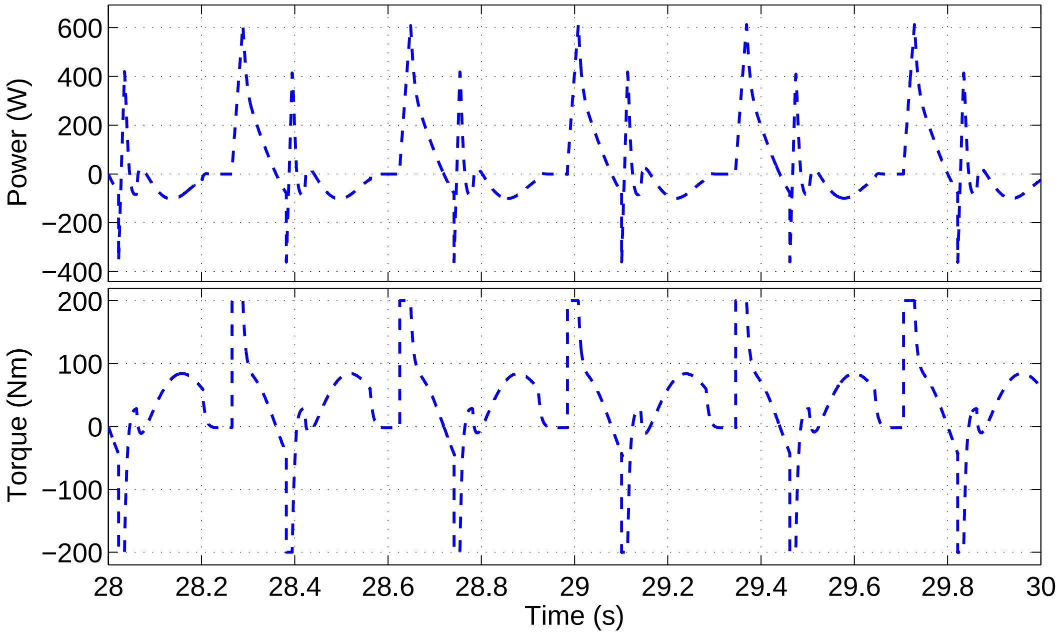

Figure 4.9: Instantaneous power consumption (top) and torque output (bottom) of spine motor during bounding gait.

Figure 4.9 presents power consumption and torque output of the spine motor during the flexible bounding gait. We can observe that spine motor saturates at certain phase of the gait by reaching the top level of 200 Nm. Except from the saturations, motor produces a maximum torque of 85 Nm. Naturally, the power consumption values are closely related with the torque output; during the saturation period spine motor consumes a maximum level of nearly 610 W. However it should be noted that the peaks observed in the hip motors do not exist in spine motor because of the continuous state flow of the PD controller.

4.5 Discussion

When we look at the results of the simulations of this model, we can clearly say that the conceptual idea presented in Chapter 3 is mathematically proved to be working and showing sound evidences of flexible spine architecture and its auxiliary role in the dynamic bounding gait. By following this conceptual idea, we have derived the dynamics and force equilibriums in the system and showed analytical inspection on the overall idea. As we have derived all the possible systems within the whole, we managed to cover a larger part of the details that we could not in the previous simulations. In addition to the required dynamic equations, we also analyzed ground friction forces and trajectory tracking system with more detail.

4.5.1 Hopping Height and Speed

If we look at the results of the simulation on hopping height and foot clearance, we do not see very important changes compared to the previous simulations. However, a very important change observed in this part is the great increase in the horizontal velocity. We managed to achieve an average speed of 3.55 m/s for our 20 kg weighing robot during a dynamic bounding gait. If the body length of the robot is concerned, this value corresponds to 3.55 $bodylength/second$ which is an important measure for the system.

There are some obvious reasons which produce such a result. The most important reason is the change in the amount of leg spring compression in this simulation. Although we have used the same spring constants and other attributes used in the previous simulations, the maximum compression rate of the leg springs were let up to the total length of the spring, that is 0.8 m. It is important to remind that, in the other simulations explained in the previous chapter, the rest length of the spring was also 0.8 m, however it was bound to compress up to 0.5 m only. For this reason, we observed a stiffer spring system in our model this time.

4.5.2 Gait Frequency

We also observe a higher frequency in the stride this time. If we look at the feet clearance graphs, we can see that there are 3 complete cycles in 1 second, which means that robot runs with 3 Hz. The reason for this is closely related with the natural frequency of springs. The damped natural frequency of a spring is described as follows:

$$\begin{aligned} f_n &= \frac{1}{2\pi}\sqrt{\frac{k}{m}} \\ \zeta &= \frac{c}{2\sqrt{km}} \\ f_d &= \sqrt{1-\zeta^2} f_n \end{aligned}$$

where given mass m, spring constant k and damping constant c,f_n is the natural frequency free of damping, ζ is the damping ratio and f_d is the damped natural frequency of a spring-damper system.

When our system is considered, the damped natural frequency of a single leg is found to be approximately 2.95. Having defined that, we see that our robot moves very close to this frequency meaning the system is stabled within the leg natural frequency. Along with this compliance, the body parts move faster than the previous robot, resulting in a faster gait. It is also for this reason, the robot does not have a double stance phase which requires softer springs and slower gaits.

4.5.3 Ground Friction and Trajectory Tracking

In our system we implemented a ground friction system which is slightly different than the mechanics of the Coulomb friction. In our which we defined in Section 4.3.1, we used a discrete state functionality where the ground friction force was applied on the horizontal velocity of the toe mass instead of its acceleration. By using such a method, we disabled the Newtonian dynamics of the toe in the stance phase and enabled them again in the flight. This method gave us the ease for implementation as no additional event declaration was required. However, we have a drawback of this method which can be seen on the trajectories of toe masses.

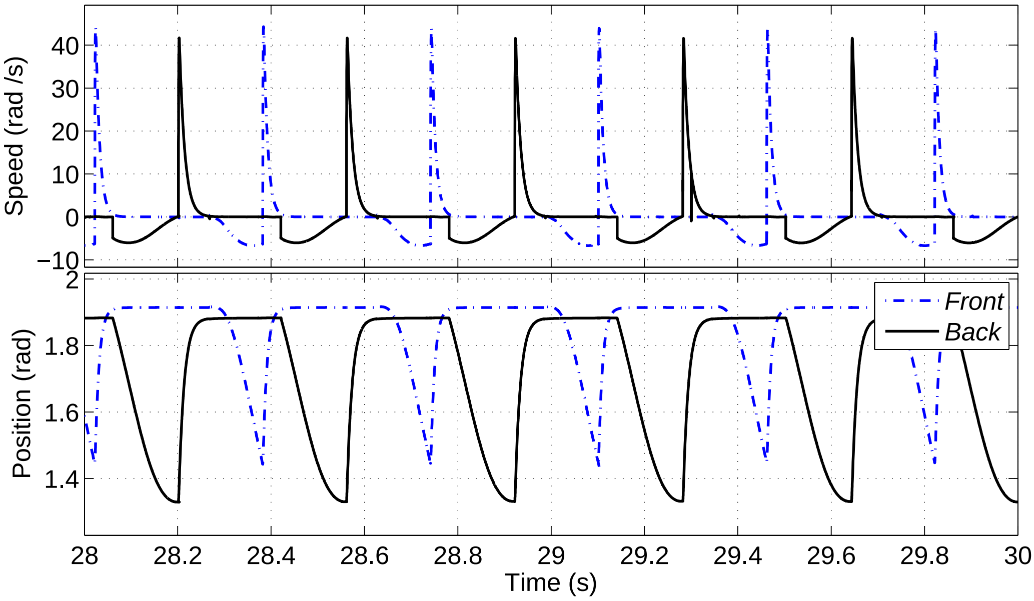

Figure 4.10: Rotational velocity (top) and position (bottom) of each hip joint during bounding gait.

Figure 4.10 shows the rotational velocity and position of each hip joint where the effects of trajectory tracking and ground friction can be seen. We see that the positioning of each leg works smoothly with respect to the PD controllers, however the ground friction force disturbs the trajectory tracking during stance phase. When investigated closely, one can see that when the legs are leaving the ground the rotational velocity of each leg reaches a peak with the intervention of the PD controller. At this state, the ground friction force disappears and leaves the dynamics of the toes in a fuzzy state. According to this situation, the the legs behave out of the control of the trajectory tracking system and moves very fast bounded with the saturation level of hip motor torque. However, we observed that these peaks give the robot an additional acceleration and have an important effect on the achieved ground speed. When we ran simulations in order to eliminate these peaks, we observed that robot cannot run with high speed with the given optimum parameter set.

4.5.4 Spinal Actuation

The main focus of this research is the effects of spinal actuation on the bounding gait and we can see the results of this theory by looking at the graphs showing increases in hopping height and horizontal speed. However, in order to understand the details of the spine we must investigate the spine motor torques and how they behave according to phases of the gait.

If we revisit the torque output of the spine motor in Figure 4.9, we can see that there are two major saturation periods in a single stride. The details of torque output with respect to the phase properties can also be seen below in Figure 4.11.

The figure shows 4 consecutive phases in a single stride of the flexible bounding gait along with the torque output of the spine motor. The shaded regions can be listed from darkest to lightest grey as double flight phase with concave body pose, back leg stance phase, double flight with convex pose and front leg stance phase. In the back stance phase, we see that spine motor produces a positive torque to change the body pose from concave to convex. We also observe that this positive torque affects the spring compression of the back leg and eventually gives an additional thrust to the robot. After this phase, the robot enters the double flight phase with convex pose, therefore the torque level of the motor becomes nearly zero. The most interesting observation from this graph is the behavior of the motor during the front stance phase. We know that in this phase, robot faces

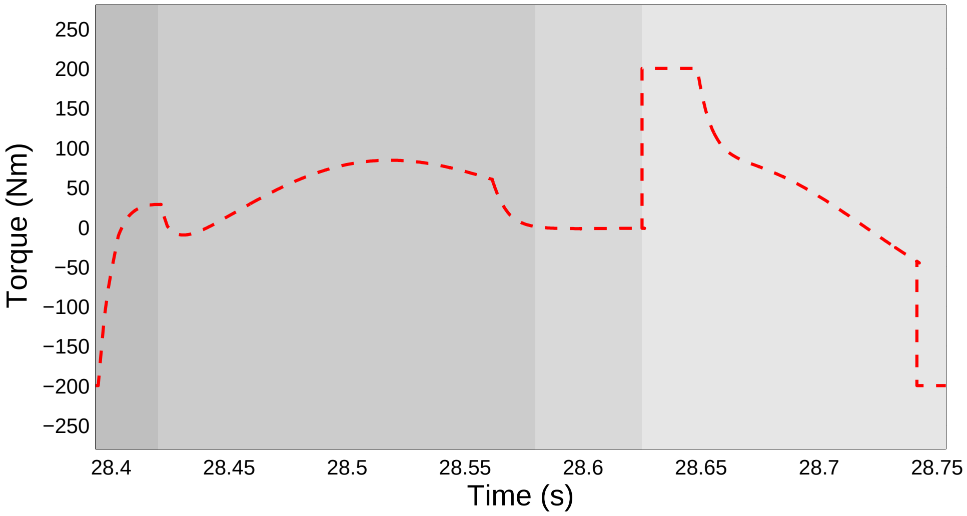

Figure 4.11: Torque output of the spine motor in a single stride that consists of 4 consecutive parts each shown with different shaded regions.

a great deal of impact forces on the front leg which is then transferred to spine joint. In the beginning of this phase, we see the motor reaching the saturation level to overcome the impact forces acting on the spine joint. In order to keep the robot in balance and the correct pose, this period of saturated positive torque lasts for some time until the impact forces are beaten. After that, the body is moved to the concave pose where we observe a torque in the negative direction. This output reaches to a saturated negative level after some point, which can be explained as the torque needed to keep the robot in the concave pose while the front leg experiences the thrust forces of the spring system. This thrust forces of the spring also acts on the spine joint which results in the saturation of the motor. We see that whenever the front leg leaves the ground, or the thrust forces of the spring disappears, the spine motor exists the saturation and returns back to normal torque output. This change in the output can be clearly seen the double flight phase with concave pose.

From this graph, we understand that spine motor uses a lot of power in the front leg stance phase in a stride. We observe this by looking at the two extreme outputs of the motor which are due to impact forces and the spring thrust forces acting on the spine joint. One interesting note we can express can be the trend of torque output in this phase. We see the torque level going from the positive extreme to negative extreme which can be soften by a passive compliance used. The damping of this compliance can resist the active forces on the joint and reduce the amount of torque produced by the spine motor in front leg stance phase. By using such a system, the power consumption can also be reduced.

Chapter 5

Conclusion

In this thesis we presented a novel flexible backed quadruped robot and a novel flexible bounding gait controller that are both inspired from the agile and fast land mammals in nature which use their flexible spine and musculoskeletal bodies. We first presented the state-of-the-art quadruped robots with changing controller strategies and leg structures. These robots achieved different performances due to their designs and controllers but they all shared a stiff trunk which we believed to limit their performances. We also presented different dynamic locomotion gaits performed by quadrupedal land mammals and investigated the properties of these gaits. We then focused on the bounding gait which is also commonly used by the multi-legged robot platforms due to its symmetric pattern and ease of implementation.

Having defined the background of the research, we presented our flexible backed quadruped robot with the spinal actuation. We defined a conceptual planar model with three motors on the robot, where two of them placed on the hips and the other one on the spinal joint. In order to make a comparison, we presented a stiff backed robot model with the same physical properties with our flexible model except from the actuated joint mechanism which connects two separate body parts. Being inspired from cheetah, we presented a novel flexible bounding gait that will fit our flexible robot. After that we used a repetitive optimization algorithm to find the best parameters that yields the best performance for bounding for both of these robot models. After creating a fair ground for comparison, we ran simulations in Working Model 2D environment to observe the performance differences between these two models. We observed that by using an actuated spine we can achieve a %17 increase in the horizontal speed and %6 increase in the hopping height of the system. We also noted that feet clearance had an increase of %150 and %350 for back and front legs respectively. We saw that total amount of power consumed in the flexible robot exceeded the stiff backed motor due the extensive usage of the spine motor. The results showed us that spine motor was used to give additional thrust to the system by increasing the amount of compression for the legs. In addition to these, we showed that the flexible body pose related with the phases defined in the flexible bounding controller increased the stride length of the robot which resulted in higher locomotion speeds.

In order to validate our conceptual model, we presented the mathematical investigation of the flexible backed robot system. We redefined the physical properties of the robot and derived all dynamic relations in the robot system. By finding the dynamics and the force equilibrium of the robot we managed to find a second order differential system which allowed us to create a simulation using integration techniques. We extended our model by adding detailed trajectory tracking and ground friction forces to increase the level of realism. For this robot model, we used an alternative bounding gait controller which we believed to enhance the locomotion performance. In the simulations ran in MATLAB environment, we observed that our robot achieved much faster locomotion with increased feet clearance. We noted that the changes in the spring stiffness and the discrete state ground friction force affected the result of the bounding gait by increasing the stride frequency. By looking at the resulting graphs, we validated our conceptual model and showed the spine role during the bounding gait.

We started our research by observing the absence of a flexible body in the dynamic running robot platforms. Inspiring from the nature, we believed that using an actuated spine would increase the locomotion performance by bringing robots one step closer to their natural correspondences. We presented novel models for a flexible backed quadruped robot and flexible bounding gait and showed that a flexible spine can increase the locomotion performance by means of giving additional thrust and power to the system as well as increasing the stride length.

5.1 Future Work

When we look at the torque outputs and power consumptions of the motors, we observe that there is a great deal of negative work done throughout the gait. One possible extension to the overall system could be implementing additional passive compliant mechanisms on joints which are actuated by motors. Mechanics of the rotational spring-damper systems can resist the extensive torque acting on the joints due to impact or thrust forces. Such systems can also reduce the amount of power consumed by the motors as they will be required to produce less torques.

One possible direction for the research is to find a passive compliant system for the spinal joint that will fit in the natural frequency of the gait. The ideal case will be this passive compliant system change the body poses as long as the robot moves in coordination with the natural frequency of the system. In such a case, the spine motor will only be used to restore the energy lost during one stride and let the robot enter the natural frequency again. Our current research results show directions for finding a suitable compliance whose damping and spring constants can be adjusted to make the robot reach the limit cycle. However, the structure of this system remains unclear as a simple rotational spring can also be as useful as a complex multi-body spine structure. Despite the fact that the structure of the mechanism cannot be foreseen, it is clear that such a system will increase the performance criteria of the robot as well as moving the overall design one step closer to its natural correspondences.

Another possible extension for this research could be investigating the role of foot retraction. Fast running land mammals use this method commonly in order to avoid land collision during high speed actions [10]. After the leg is swung backwards in the stance phase, it is retracted inside, towards the body to create a larger space between the tip of the toe and the ground. When the legs are pulled inside to a certain point, it becomes safer to place them for the touchdown position as the trajectory of the leg does not collide with obstacles on the ground or the ground itself. Implementing such as system for a fast running robot like ours can increase the chances of reaching higher speeds without colliding with the ground.

Bibliography

[1] R. M. Alexander. The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research*, 3:49–59, 1984.

[2] M. D. Berkemeier. Modeling the dynamics of quadrupedal running. *The International Journal of Robotics Research*, 17:971–985, 1998.

[3] B. Boszczyk, A. Boszczyk, and R. Putz. Comparativeand functional anatomy of the mammalian lumbar spine. *Anatomical Record*, 264(2):157–168, OCT 1 2001.

[4] M. Buehler, R. Battaglia, A. Cocosco, G. Hawker, J. Sarkis, and K. Yamazaki. Scout: A simple quadruped that walks, climbs, and runs. In *International Conference on Robotics and Automation*, volume 2, pages 1707 – 1712, 1998.

[5] D. Campbell and M.Buehler. Preliminary bounding experiments in a dynamic hexapod. In *International Symposium on Experimental Robotics*, 2002.

[6] T. M. Caro. Ungulate antipredator behaviour: Preliminary and comparative data from african bovids. *Behavior*, 128(3-4):189–228, 1994.

[7] P. Chatzakos and E. Papadopoulos. Bio-inspired design of electrically-driven bounding quadrupeds via parametric analysis. *Mechanism and Machine Theory*, 44:559–579, 2009.

[8] S. Chernova and M. Veloso. An evolutionary approach to gait learning for four-legged robots. In *International Conference on Intelligent Robots and Systems*, 2004.

[9] U. Culha and U. Saranli. Quadrupedal bounding with an actuated spinal joint. In *International Conference on Robotics and Automation*, 2011.

[10] L. M. Day and B. C. Jayne. Interspecific scaling of the morphology and posture of the limbs during the locomotion of cats (felidae). *Journal of Experimental Biology*, 210:642–654, 2007.

[11] I. Design Simulation Technologies. *Working Model 2D User's Guide*. Canton, MI, 2007.

[12] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning cpg-based biped locomotion with a policy gradient method: Application to a humanoid robot. *International Journal of Robotics Research*, 27:213–228, 2008.

[13] J. Estremera and K. J. Waldron. Thrust control, stabilization and energetics of a quadruped running robot. *The International Journal of Robotics Research*, 27:1135–1151, 2008.

[14] G. Gabrielli and T. H. von Karman. What price speed? *Mechanical Engineering*, 72(10):775–781, 1950.

[15] P. P. Gambaryan. *How Mammals Run: Anatomical Adaptations*. Wiley, 1974.

[16] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[17] F. Iida and R. Pfeifer. cheap rapid locomotion of a quadruped robot: Self-stabilization of bounding gait. *Intelligent Autonomous Systems*, 8:642–649, 2004.

[18] K. F. Leeser. Locomotion experiments on a planar quadruped robot with articulated back spine. Master's thesis, Massachusetts Institute of Technology, 1996.

[19] D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans. Automatic gait optimization with gaussian process regression. In *International Joint Conference on Artificial Intelligence*, 2007.

[20] L. D. Maes, M. Herbin, R. Hackert, V. L. Bels, and A. Abourachid. Steady locomotion in dogs: Temporal and associated spatial coordination patterns and the effect of speed. *Journal of Experimental Biology*, 211:138–149, 2008.

[21] J. A. Nelder and R. Mead. A simplex method for function minimization. *Journal of Computation*, 7:308–313, 1965.

[22] D. Papadopoulos and M. Buehler. Stable running in a quadruped robot with compliant legs. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 1, pages 444 –449 vol.1, 2000.

[23] I. Poulakakis. On the passive dynamics of quadrupedal running. Master's thesis, Department of Mechanical Engineering, McGill University, 2002.

[24] I. Poulakakis and J. A. Smith. Modeling and experiments of untethered quadrupedal running with a bounding gait: The scout ii robot. *The International Journal of Robotics Research*, 24:239–256, 2005.

[25] M. Raibert, K. Blankespoor, G. Nelson, R. Playter, and M. Buehler. Bigdog, the rough-terrain quadruped robot. In *Proc. of the International Federation of Automatic Control*, 2008.

[26] M. H. Raibert. *Legged robots that balance.* MIT Press, Cambridge, MA, USA, 1986.

[27] M. H. Raibert. Trotting, pacing and bounding by a quadruped robot. *J. of Biomechanics*, 23(1):79–98, 1990.

[28] U. Saranli, M. Buehler, and D. E. Koditschek. Rhex: A simple and highly mobile robot. *The International Journal of Robotics Research*, 20(7):616–631, July 2001.

[29] M. F. Silva and J. T. Machado. A historical perspective of legged robots. *Journal of Vibration and Control*, 13:1447–1486, 2007.

[30] J. Smith, I. Sharf, and M. Trentini. Paw: a hybrid wheeled-leg robot. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 4043 –4048, may 2006.

[31] J. A. Smith, I. Sharf, and M. Trentini. Bounding gait in a hybrid wheeled-leg robot. In *International Conference on Intelligent Robots and Systems*, 2006.

[32] T. Takuma, M. Ikeda, and T. Masuda. Facilitating multi-modal locomotion in a quadruped robot utilizing passive oscillation of the spine structure. In *International Conference on Intelligent Robots and Systems*, 2010.

[33] C. R. Taylor, N. C. Heglund, and G. M. O. Maloiy. Energetics and mechanics of terrestrial locomotion. 1. metabolic energy consumption as a function of speed and body size in birds and mammals. *Journal of Experimental Biology*, 97:1–21, 1982.

[34] K. Tsujita, K. Tsuchiya, and A. Onat. Adaptive gait pattern control of a quadruped locomotion robot. In *International Conference on Intelligent Robots and Systems*, 2001.

Appendix A

Code

A.1 Nelder-Mead Optimization Method Algorithm

```
Calculate initial $P_i$ and $y_i$

While ( minimum not reached )
        Determine $h$, calculate $\bar{P}$
        Form $P^* = (1+\alpha)\bar{P} - \alpha P_h$
        Calculate $y^*$
        If $y^* < y_l$
           Form $P^{**} = (1+\gamma)P^* - \gamma\bar{P}$
           Calculate $y^{**}$
           If $y^{**} < y_l$
              Replace $P_h$ by $P^{**}$
           Else
              Replace $P_h$ by $P^*$
        Else
           If $y^* > y_i, i \neq h$
              If $y^* \leq y_h$
```

```
            Replace $P_h$ by $P^*$
          Form $P^{**} = \beta P_h + (1 - \beta)\bar{P}$
          Calculate $y^{**}$
          If $y^{**} > y_h$
            Replace all $P_i$'s by $(P_i + P_l)/2$
          Else
            Replace $P_h$ by $P^{**}$
        Else
          Replace $P_h$ by $P^*$
      Calculate convergence
End
```