HORIZONTAL MOTION PLANNING FOR MULTI-LEGGED ROBOTS

by

Dian Jiao

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Summer 2017

HORIZONTAL MOTION PLANNING FOR MULTI-LEGGED ROBOTS

by

Dian Jiao

Approved: _____
Herbert G. Tanner, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____
Ajay K. Prasad, Ph.D.
Chair of the Department of Mechanical Engineering

Approved: _____
Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____
Ann L. Ardis, Ph.D.
Senior Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

First, I would like to thank Dr. Herbert G. Tanner. He is my advisor of my master thesis. I was lucky to be one of his students and studied in his group. This work can not be finished without his guidance. Here, I want to really thank him for teaching me not only about knowledge but also about patience and understanding. In these two years, Dr. Tanner met with me once a week to talk about my master thesis. He encouraged me to bring new ideas and solve the problem by myself. When I had difficulties, Dr. Tanner always provided the valuable suggestions and helped me to solve the problems. Under his guidance, I learned the fundamental knowledges of robotics and wrote my first thesis in my life. All I have learned through his guidance in these two years will help me succeed in my future study.

I also would like to appreciate Dr. Guoquan Huang, Dr. Christopher Rasmussen and Dr. Jeffrey Heinz for accepting my invitation and joining my thesis committee. I want to thank Dr. Huang, and Dr. Heinz to spend their time to review my thesis and attend my defense.

I would like to thank all the faculty in the University of Delaware Mechanical Engineering Department. I here thank them all, with deep sincerity, and thank them again for for introducing me to the Mechanical Engineering. I also would like to thank all the staff in the department who provided valuable support when I had questions.

I want to thank Adam Stager and Caili Li for their support and friendship. Adam can always bring some new ideas when I discussed some problems with him. During these two years, I learned how to use 3D Printer and Laser Cutter from him. He also reviewed my writing and helped me to fix the grammar and spelling mistakes. I also would like to appreciate Caili for teaching me MATLAB and discussing the problems that I met in my work.

I cannot finish my studies without my parents' support. They supported my study from 2009 and never interrupted from 2009 to 2017. I would like to express my sincere gratitude to my mother, Juan Hou, my father, Xinhua Jiao, and my uncle, Chun Jin. Thank them for everything they have done for me. Finally, I would like to thank to my fiancee, Rui Huang for accompanying me for past three years and bring me happiness.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Solving legged robot navigation problems is challenging because of the robots' locomotion limitations and its complex kinematics and dynamics. Generic locomotion models for the legged robots such as the Spring Loaded Inverted Pendulum (SLIP) and the Lateral Leg Spring (LLS) involve masses, accelerations and second–order differential equations. Common path planning methods such as the Probabilistic Roadmap (PRM) and the Rapidly–exploring Random Tree (RRT) have their own limitations. Both of them typically assume that the robots do not have any kinematic limitations. In addition, RRT and PRM are open–loop, and as such, they do not produce the feedback strategies that correct when the robot is away from the desired path. Consequently, re–planning is practically required for implementations of the RRT and PRM.

Templates are simplified models that capture salient features of robot motion behavior. The SLIP and the LLS are considered templates. Karydis et al. [1] provides a new locomotion template: the Switching Four Bar Mechanism (SFM) for legged robots. The SFM gives a static map between model parameters and robot displacement, and models the robot's motion without differential equation. On the other hand, the SFM offers the robot's configuration in closed form. The number of variables of the SFM can be reduced to one.

In this thesis, we derive the inverse kinematics of the SFM, which maps the robot displacement to model parameters. We also provide a method for solving the problem of legged robot navigation in a way that provides feedback strategies in cluttered planar environments. This is achieved by combining the SFM as the locomotion template for the legged robots with navigation functions for motion planning. In this way, an existing, multi–variable, probably correct motion planning method, is transformed into a tractable single–variable. Locomotion–specific optimization algorithms are applied

to our navigation method and are adjusted to hit a trade–off between efficacy and processing speed. Because our method provides feedback strategies, re–planning is not required. We provide some convergence conditions for our navigation method so that we can ensure that motion plans are always safe with regards to collisions with environmental boundaries. In conclusion, this thesis provides an approach that can be used for solving the planar navigation problem for robotic vehicle systems with kinematics given in closed–form.

Chapter 1

INTRODUCTION

Legged robots have great potential. They can be used in many complicated environments. The locomotion designs of most legged robot are based on locomotion of myriapod. Because most legged robots have different kinds of limitations, such as turning and speed restrictions, an appropriate navigation method for legged robots is needed. The navigation method for solving the problem of legged robot motion planning needs to be effective and tractable. Traditional motion planning methods may not be suitable for legged robot because the limitations of legged robots' locomotion need to be considered. The work in this thesis focuses on providing a method to solve the navigation motion planning problem for legged robots. We use the switching four bar mechanism model as the kinematic model. Our method solves the problem of planar legged robot navigation in cluttered planar environments in a way that provides a feedback law, checked against convergence and safety conditions, and is new, useful and challenging.

1.1 Legged Robot Applications

Legged robots have better mobility in rough terrain. Not only can the legs support the robot in rough terrain, but they also provide suspension so that the robot can step over obstacles. As the means for navigation and design of legged robots have become more mature, legged robots are able to be used in different fields in the real world. Here are some application examples:

BigDog is a four legged robot which was produced by Boston Dynamics in 2005. It was used to carry soldiers and heavy materials on rough terrain. BigDog is as big as a large dog or small mule. It has a 3 foot long and 2.5 foot high body. The weight of

BigDog is 240 lbs. BigDog can run at 4 miles per hour, climb 35 degrees slope and run in snow and water. It can carry 340 lbs of materials. BigDog locomotion is controlled by an on–board computer. The on–board computer also handles communications between the robot and users. The control system of BigDog solves the balance problem. BigDog has approximately 50 sensors. Those sensors are used to detect the battery power, environment temperature, balance of robot and so on. BigDog was considered for military applications but noise has been a concern.

Legged robots are lacking in terms of speed compared to animals. Cheetah is a four footed robot. It can run 28 miles per hour and jump over obstacles. It is the fastest legged robot in the world. Similar to an animal, the Cheetah has an articulated back that flexes back and forth on each step. Cheetah is as big as a small dog.

Legged robots also show great advantages in walking in the complex terrain. LAURON is designed for exploration of volcanoes, clearing of minefields and search for waste in ecologically–sensitive areas. It can be used in places that would be difficult to access for wheeled robots. LAURON is a six legged robot. It is produced by the FZI Computer Research Centre in Karlsruhe. The robot's design inspiration is from the Indian stick insect. Each leg is 50 cm. The angle of the robot's body can also be altered by using two independent axles (swivel and tip). Thus, LAURON has more than 20 degrees of freedom. There are 3D power sensors and elasticity measurement systems in each foot to detect collisions. The maximum torque of motor, which provides the power of joints, is around 20 Nm (or 40 m for a short time).

1.2 Previous Study of Legged Robots

The potential of legged robots has been acknowledged in recent years. Apart from the legged robots mentioned above, there are a lot of other experimental prototypes legged robots, which may have a simpler structure and fewer motors and controllers, but still exhibit basic locomotion abilities. Navigation methods of the legged robots is an important part of legged robot research. In what follows, we present some

previous designs of legged robots, some models that have been proposed in the literature as templates for legged locomotion, and some navigation methods for legged robots.

1.2.1 Previous Designs

The SEA hexapod is a six legged robot which is designed to produce an open–loop gait in which phase oscillators coordinate the desired joint relationships. Each leg has three, single degree of freedom, joints. The end of each leg is a rubberized foot. The SEA hexapod utilizes open–loop strategies during locomotion [2].

The OctoRoACH is designed with a simpler structure and manufacturing process. OctoRoACH is an eight legged robot. There are two motors in each side to control the leg velocity independently. By changing the velocity on one side, the robot can turn left or right. The faster side would be on the outside of a turn [3].

The SailRoACH is designed and produced in UC Berkeley's laboratory. SailRoACH is a 6 legged robot which is driven by two 7 mm brushed DC motors. One motor is in the left side and the other motor is in the right side. A worm gear is used to transfer the power from motor to legs. There is another motor that allows the robot for 360 degree rotation. The size of SailRoACH is 10 cm. The mass of SailRoACH is 45 grams. The left front leg, left rear leg and right mid leg are coupled as a pair, the right front leg, right rear leg and left mid leg are coupled as the other pair. The robot runs by alternating one pair with the other pair. The robot can turn in a 1.2 m radius at 1.6 m/s. The SailRoACH uses its sail to generate a torque, that turns the robot faster [4].

To further decrease the robot's weight and robot's size, a very small robot with 6 legs has been produced by smart composite microstructures, called RoACH. RoACH is a 2.4 grams robot. To turn, both mid legs of RoACH are shorter than the other four legs. Thus, the mid legs are not involved at the instant of leg's switching. Since the legs on the left side move different direction from the legs on the right, a moment is

applied to the robot. Therefore, one or both sides will slip. The slipping will make the robot turn left or right [5].

The 1STAR is a legged hexapedal robot using a single actuator to control in–plane locomotion. There are three legs on the left side and three legs on the right side. The robot has a 12 cm body and 73 gram weight including battery and control board. The robot has a variable leg sprawl angle in the transverse plane. The robot can change the distance from its body to the ground, and also the contact angle between the leg and surface. The sprawl angle can be set from positive 60 degrees to negative 90 degrees. STAR is another robot that combines the legged robot and wheeled robot. Turning the robot is achieved by driving each side of the legs with different velocity [6, 7].

The designs of bipedal robots have a lot differences from the multi-legged robots because of the large number of the robots' joints. BiPed robot is a 2–legged robot which has high performance in rough terrain, steep stairs and in environments with obstacles. Each leg of the robot has a thigh, a shin, and a foot. The mass of robot is 83 kg. There are 12 joints in the robot, each joint is provided the power by an AC servo motor. The joints at the thigh, shin, and foot have 6 degree of freedom. The hip joint, knee joint and two ankle joints have 3 degree of freedom [8].

The choice of navigation method for legged robots is based on the characteristic of the robots' locomotion. The RRT is used as the navigation method in most of the legged robots. The RRT planner can be used in OctoRoACH, SailRoAch, TAYL-RoACH, MiniRoACH, Bigdog, SEA hexapod and Biped V–3 [9, 10, 11]. Navigation methods are reviewed in next section.

1.2.2 Legged Robots Locomotion Templates

The navigation methods for legged robots are based on robots' dynamical locomotion templates. To the best of our knowledge, there are three dynamical locomotion templates that are widely used for legged robots. These are the Spring Loaded Inverted Pendulum (SLIP), the Lateral Leg Spring (LLS) and the Switching Four Bar

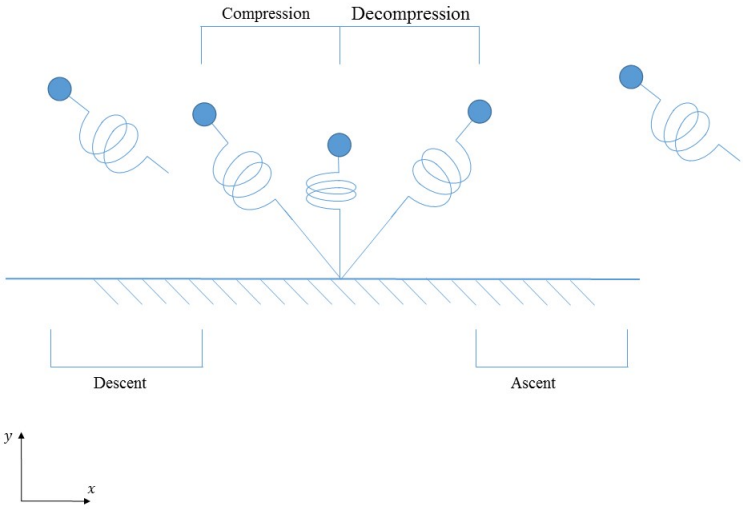


Figure 1.1: SLIP Model. Descent phase, compression phase, decompression phase and ascent phase.

Mechanism (SFM). The three templates have their own characteristics.

The Spring Loaded Inverted Pendulum (SLIP) is a fundamental template to analyze dynamical locomotion of legged robots. This model consists of a point mass which represents the total loaded mass force that is concentrated at that point (See Figure 1.1). The mass of leg is neglected. The leg is made by a spring which will be compressed and decompressed during the stance phase. SLIP has two phases. The stance phase is based on the rate of change of leg length. The flight phase is based on the vertical velocity. When the legs are on the ground, the legs are assumed as not to be slipping.

There are two subphases in the stance phase, compression phase and decompression phase. During the compression phase, the rate of change of leg length is negative. The energy is stored because the spring is compressed (See Figure 1.1). During the decompression phase, the rate of change of leg length is positive. The energy is released because the spring is decompressed [11].

There are two subphases in the flight phase, ascent phase and descent phase.

During the ascent phase, the vertical velocity is positive and the magnitude is decreasing. The potential energy is increasing because the distance between the leg and the ground is increasing (See Figure 1.1). During the descent phase, the vertical velocity is negative and the magnitude is increasing. The potential energy is decreasing because the distance between the leg and ground is decreasing [2].

The transition events in SLIP are defined as the transition instants when the different phases are switched. There are several transition events such as apex, touchdown and bottom. The apex event occurs at the switching instant from ascent to descent. At the apex event, the whole system reaches the maximum height, the system has maximum potential energy. At the touchdown event, the leg touches down to the ground. At the bottom event, the leg length reaches the minimum height, spring potential energy reaches the maximum.

The whole system can be formalized as a group of differential equations. The general solution for this group of differential equations is complex and unavailable in closed form because of the nonlinearities [11].

Lateral Leg Spring (LLS) is a template which can be used in modeling lateral stabilization and studying the turning problem for hexapedal robots. The LLS model is described in detail by Schmitt and Holmes [12].

The whole system consists of a rigid body which has a mass m and an inertia I. The weight of the legs which is in the lateral of the body are neglected. Each side has two legs which are made of spring. This model can be used to quadruped robots and hexapod robots.

A full step begins at the time t when the left legs touch down to the ground. The spring is relaxed at a leg angle relative to body orientation. After the spring relaxed, the left legs would begin its swing phase and the right legs would touch down to the ground at this moment with a negative degrees of angle. The two phases would be switched again and again. By setting different starting touchdown angles, the robot would turn left or right. LLS is also expressed as a set of nonlinear differential equations [12].

In terms of some simple, but not completely trivial kinematic model for horizontal plane locomotion, the Switching Four Bar Mechanism (SFM) model is introduced. SFM is a model in the horizontal plane and consists of a rigid body and four rigid legs. The left front and right rear leg are coupled as a pair. The right front and left rear leg are coupled as the other pair. Once a pair of legs touch the ground, the other legs leave off the ground. When the legs touch the ground, no slipping is assumed between the legs and the ground. The angle from the leg's touchdown point to the mid of body is defined as the touchdown angle, and the angle from mid of body to liftoff point is defined as the liftoff angle. The robot would turn left or right if one pair's liftoff or touchdown angles are changed [1].

SFM gives a static map between model parameters and robot displacement. Because the velocity is not involved in the model, the parameters would not relate directly with time t. The only parameters are the touchdown angle, the liftoff angle, the robot body length and the robot leg lengths. The output of those parameters after a single step are the displacement of the robot in x axis, the displacement of the robot in y axis and the changing of the robot's direction. Based on this, no integration of an ODE is necessary.

1.2.3 Legged Robots Navigation Methods

To make legged robots autonomous, motion planning is needed. The three most common planning methods for legged robots are the Probabilistic Roadmap (PRM), the Rapidly–Exploring Random Tree (RRT), and the navigation functions [13].

The Probabilistic Roadmap (PRM) is a planning method that gives collision–free paths for robots. The obstacles need to be stationary. Probabilistic Roadmap (PRM) is open–loop. So far, PRM is only applicable to holonomic systems [14]. The method includes a learning phase and a query phase. There are two successive steps in the learning phase. The first is the construction step and the second is the expansion step. In the construction step, the objective is to obtain a reasonably connected graph with nodes and edges inside the free workspace. Nodes are generated through

a process of 'creation of random configurations,' using a 'local planner,' a set of 'node neighbors' and a 'distance function' [14]. The expansion step is necessary to improve the connectivity of the graph generated by the construction step [15].

Paths can be found between the robot input starting position and goal during the query phase. The query phase is faster compared to the learning phase. The roadmap is constructed in the learning phase. After the roadmap is constructed, PRM can be used for solving multiple–query planning which means it can solve many path planning problems for the same environment [11].

Rapidly–Exploring Random Tree is a simple planning method for solving the single–query path planning problem. Rapidly–Exploring Random Tree (RRT) is also open–loop. RRT has been adapted for use with non–holonomic systems [16].

RRT can provide a path quickly in a space with obstacles and boundary conditions. A variation of this method works by incrementally building two Rapidly–Exploring Random Trees (RRT), one is starting from the robot starting position and the other one is from the goal. Each tree grows branches into the space around them and towards each other.

RRT can be applied to robots which have motion constraints after some adjustments, in a form known as kino–dynamic path planning [17]. Compared to PRM, RRT has increased efficiency in solving single–query path planning problems, because RRT does not need a preprocessing, or a learning phase.

Navigation function methods integrate the robot's motion planning problem with the robot's control problem. A navigation function provides a feedback law and provable convergence conditions in clutter planar environments. The information needed includes the goal, boundary conditions and obstacles, encoded in the form of functions.

Once appropriately turned, the navigation function has only one minimum: the goal configuration [18]. By gradient descent, the robot would reduce the 'height' of its location on potential landscape and approach the goal. Once the height of its location on the grid is minimal, the robot arrives to the goal [19].

1.3 Challenges

1.3.1 Templates Model Challenges

The existing locomotion templates of legged robots have their respective advantages and disadvantages. Both SLIP and LLS models are inspired from insect and animal locomotion. The model of SLIP and LLS includes the velocity of the leg's motion. So there are differential equations in dynamic constraints. Due to gravity, the stance dynamics in SLIP and LLS has several non–integrable terms [20, 21]. Therefore, there is not an exact analytical solution for stance map in both templates.

Existing work on the SFM model provides expressions for the forward kinematics. In his work, the step motion is parameterized by the leg touchdown angles, the liftoff angles, the body length and the leg length. The forward kinematics provides the x displacement, y displacement and changing of robot direction if the motion parameters, the touchdown angles, the liftoff angles, the body length and the leg length, are known. For solving navigation problems, the method for computing the liftoff angle is needed, in a form that is analytically tractable.

1.3.2 Navigation Method Challenges

Although there are some navigation methods for legged robots, there is no simple and tractable method that can be used in general case for solving the legged robot navigation problem. Because of the working principle of PRM, most of the legged robots which have locomotion limitations cannot use it. For example, if a robot has turning limitations that the robot needs a turning radius to complete the turning behavior, PRM cannot guarantee that robots with turning radius can move from the one point to the next connected point.

Compared to the PRM, RRT can be used in robots that have locomotion limitations. But because of Randomness, RRT does not provide the shortest path from the robot starting position to goal (there are recent optimal extensions: RRT*) [22].

Both RRT and PRM are open–loop. They do not produce feedback strategies and thus cannot inform when the robot is away from the path. Consequently, re–planning is required for the RRT and PRM.

Navigation function is closed loop and has convergence guarantees. This method has never been used with the legged robots. This paper proposes a new idea about how to use the navigation function as a method of solving the legged robot motion planning problem.

1.4 Contributions

This thesis provides the method of combining the SFM with navigation functions, the existing multi–variable probably correct motion planning method (navigation function) can be transformed into a single–variable static kinematic map.

We provide a new idea of reducing the number of parameters of the SFM in navigation function. If motion planning goal is expressed as bringing the new function to zero, the path planning problem reduces to single–dimensional function minimization. Some optimization methods of navigation function are also proposed in this thesis.

The work in this thesis can also be generalized to provide an approach that can be used for solving the planar navigation problem for systems with kinematics given in closed–form.

Chapter 2

SFM MODEL AND ITS REALIZATION

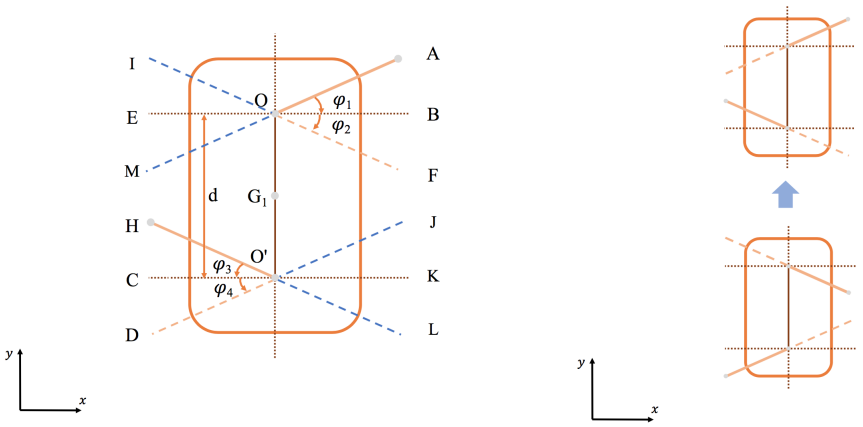
2.1 Main Claims

A feasible motion planning method for legged robots needs to accomodate the limitations of legged robot locomotion, such as turning radius or speed constraints. It is also desirable that a motion planning method provides a feedback law and provable convergence conditions. PRM is not typically used in the navigation of legged robots primarily because it cannot incorporate constraints easily. Both PRM and RRT do not produce feedback strategies; both of them are open–loop. The incorporation of constraints necessarily take some (at least) kinematic model of the legged robot. Some models, such as SLIP and LLS, introduce complexity challenges due to capturing dynamics by means of differential equations. Regardless of the modeling approach, motion planning is typically formulated either as a search, or an optimization problem, and such, it is inherently computationally and analytically complex.

For solving the legged robot motion planning problem, in this thesis, the Switching Four Bar Mechanism (SFM) becomes the legged robot kinematics model of choice. The reasons for selecting the SFM are as follows. First, SFM is modeling the legged robot motion without differential equations. Second, although there are several parameters in the SFM, the number of *variables* in SFM's equations can be reduced to one. This simplifies the computation of navigation strategies and may subsequent convergence analysis.

2.2 SFM Model Description

The Switching Four–Bar Mechanism (SFM) is a kinematic template for legged robots traveling at low speed. SFM can be used to model multiple miniature legged



(a) (b)

Figure 2.1: (a) The SFM template. Two pairs of rigid legs alternate, forming two four bar linkages. d is the distance between the two hip point joints, l is the leg length, and G is the mass center. (b) A single step of robot motion. The active pair is marked with dash lines.

robots in the horizontal plane. The SFM is a 2D model, that allows closed–form state propagation. The robot motion can be separated into the single step motion. Additionally, most of the forward kinematics parameters can be treated as the constant (See Section 2.3). The SFM model has been proposed as a template, alongside SLIP and LLS [1].

The SFM consists of a rigid body and four rigid legs. The robot's geometric center is defined as G. There are four rigid legs in the SFM (See Figure 2.1). The length of each leg is the same. The angle that starts from the leg touchdown point to the mid line is defined as the touchdown angle φ_1; the angle that starts from the mid line to lift off point is defined as the liftoff angle φ_2. For example, the left front leg touches the ground at point A, we define $\angle AOB$ as the touchdown angle of the right front leg. The right front leg leaves the ground at point F, we define $\angle BOF$ as the liftoff angle of the right front leg. $\angle AOB$ always has a positive value.

In the SFM, the four rigid legs are coupled to each other, the left front leg and the right rear leg are coupled as a pair, the right front leg and the left rear leg are

coupled as a pair (See Figure 2.1(a)). The legs in one pair have the same touchdown angle and liftoff angle. The velocity of legs in one pair is also the same. The motion of the legs in one pair is therefore symmetric relative to the longitudinal axis. The motion of legs in one pair is opposite in phase to the motion of legs in the other pair. For example, the leg OA makes up the right pair and the leg $O'J$ makes up the left pair, when the leg OA touches the ground at one instant, the leg $O'J$ leaves the ground at that instant (See Figure 2.1(b)). In the lateral side view, the motion of the foot makes a circle (See Figure 2.2).

The SFM needs to follow the following design specifications [1]:

Design specification 1 [1]: *Once the pair of legs touches the ground, the foot of each leg is fixed on the ground. There is no slipping while the foot remains on the ground.*

Design specification 2 [1]: *Once a pair of legs touches the ground, the other pair of legs leaves the ground. Both pairs of legs are on the ground only at switching instants.*

Design specification 3 [1]: *The robot motion is based on the switching activation of the left pair and the right pair.*

Design specification 4 [1]: *The two legs in one pair have the same touchdown angle* $\varphi_1 = \varphi_3$ *and the same liftoff angle* $\varphi_2 = \varphi_4$.

We define a sweep angle ψ as the absolute difference between the touchdown angle and liftoff angle. In order to avoid a collision between the legs and robot's body, the sweep angle is constrained as $-\pi < \psi < \pi$, the relationship between the touchdown angle φ_1 and the liftoff angle φ_2 is $-\frac{\pi}{2} < \varphi_2 < \varphi_1 < \frac{\pi}{2}$. The robot's body length is denoted d and robot's leg length is denoted l.

2.3 Forward Kinematics of the SFM

The SFM model is intended to capture the kinematics of the robot within a single step. In a single step, either the left pair of legs or the right pair of legs is activated. When the right pair of legs is activated, the input parameters of the forward

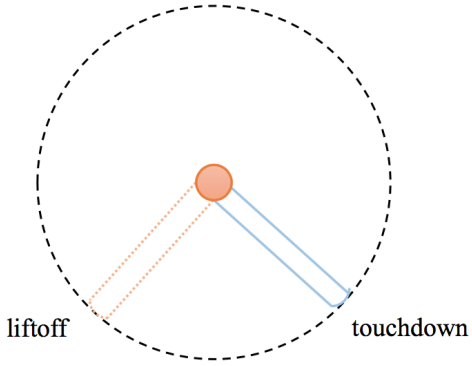


Figure 2.2: The lateral side view of a single leg motion. The sweep angle ψ is the absolute difference between the touchdown and liftoff. The step length is defined from touchdown point to liftoff point.

kinematics of the SFM are the touchdown angles φ_1, the liftoff angles φ_2, the body length d and the leg length l–all parameters are provided in Figure 2.1. The output of forward kinematics of the SFM are the displacement of robot in x direction Δx, the displacement of robot in y direction Δy, and the change in robot's direction $\Delta\theta$ (See Figure 2.3).

As mentioned before, only one pair of legs is activated in a single step. The robot's displacement in local frame Δx, Δy, and the changing of robot's direction in local frame $\Delta\theta$, can be computed. When the right pair of legs is activated, the step motion in local frame can be expressed as:

$$\Delta x = f(d, l, \varphi_1, \varphi_2) \tag{2.1}$$

$$\Delta y = g(d, l, \varphi_1, \varphi_2) \tag{2.2}$$

$$\Delta\theta = h(d, l, \varphi_1, \varphi_2) \tag{2.3}$$

With (2.1), (2.2) and (2.3), the change in displacement of the local frame can be determined if d, l, φ_1, φ_2 are known. The expansion of (2.1), (2.2) and (2.3) is given

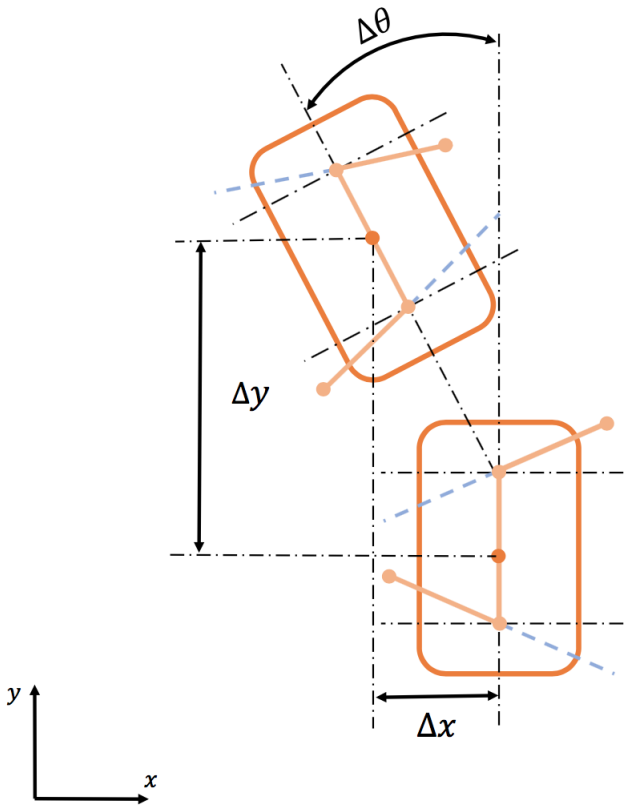


Figure 2.3: The turning motion in a single step. By changing the liftoff angle of one pair of legs in a single step, the robot is able to turn left or right. GG' in x axis is the displacement of robot in x. GG' in y axis is the displacement of robot in y. $\Delta\theta$ is the changing of robot's direction.

in Appendix A.

If the touchdown angle φ_1 and the liftoff angle φ_2 are equal to each other, $\varphi_1 = \varphi_2$, in a single step, the displacement of robot in x direction, Δx, and the change in robot's direction, $\Delta\theta$, are equal to 0. In this case, the system will move forward in a straight line (See Figure 2.1(b)). The step length only depends on the SFM's body and leg length.

If the touchdown angle φ_1 and the liftoff angle φ_2 are not equal to each other, the displacement of robot in x direction, Δx, and the change in robot's direction, $\Delta\theta$, are not equal to 0. In this case, the robot will turn left or right (See Figure 2.3). When $\Delta\theta$ has a positive value, the robot would turn right. When $\Delta\theta$ has a negative value, the robot would turn left. The step length of robot depends on the touchdown angles φ_1, the liftoff angles φ_2, the body length d, and the leg length l.

In order to simplify computation and control, we reduce the variables in the forward kinematics of the SFM. There are four parameters in the forward kinematics of the SFM: body length; leg length; touchdown angle; and liftoff angle. The body length and the leg length are set to constants, as they refer to the robot's geometry. The value of the touchdown angle can be also treated as a constant. For example, we can set the touch down angle to be $\frac{\pi}{2}$. By changing the liftoff angle, turning can be achieved. With the value of the touchdown angle to be $\frac{\pi}{2}$. The boundary condition of the liftoff angle is set at $-\frac{\pi}{2} < \varphi_2 < \frac{\pi}{2}$. By discretization and exhaustive search, we found that the maximum turning angle of the SFM in a single step is 12.7 degrees, when the touchdown angle is set to be $\frac{\pi}{2}$ and the liftoff angle is set to be 0.

The SFM kinematics can be separated into two single step functions. The left pair displacement equations, $f_L(d, l, \varphi_1, \varphi_2)$, have exactly the same structure as those of the right pair, which are denoted $f_R(d, l, \varphi_1, \varphi_2)$, and differ only in the signs of Δx and $\Delta\theta$. Because there is symmetry in the SFM model about the longitudinal axis, the motion of one pair mirrors the other pair.

2.4 Closed–Form State Propagation of SFM

In each pair, there is a progression of the position vector $P_{G-G'} = (\Delta x, \Delta y)$ in local frame and the direction of model is $\Delta\theta$ (See Figure 2.3). By reducing variables in the forward kinematics of the SFM, the right pair state propagation in the local frame becomes a function of only one variable φ_1 [1]:

$$\Delta x = |OG'| \sin(w' - |\Delta\theta|) - |O_1G| \sin(w) \tag{2.4}$$

$$\Delta y = |OG'| \sin(w' - |\Delta\theta|) - |O_1G| \cos(w) \tag{2.5}$$

$$\theta = \varphi_1 - \varphi_2 + \angle AO_1A' \tag{2.6}$$

In a single step, when the left pair is activated, the displacement of the SFM in the x direction, the displacement in the y direction, and the change in SFM's orientation is [1]:

$$\Delta x = -|OG'| \sin(w' - |\Delta\theta|) + |O_1G| \sin(w) \tag{2.7}$$

$$\Delta y = |OG'| \cos(w' - |\Delta\theta|) - |O_1G| \cos(w) \tag{2.8}$$

$$\theta = -\varphi_1 + \varphi_2 - \angle AO_1A' \tag{2.9}$$

Let us denote the state of the model in the global frame before initiating the step motion of the right pair as R^{L-}, the state of the model in global frame after completing the right pair step as R^{L+}, the state of the model in the global frame before initiating the left pair step as L^{L-}, the state of the model in the global frame after the completion of the left pair's step as L^{L+}.

Let us also define a 4×4 homogeneous transformation matrix T^R [1] that maps Δx, Δy, and $\Delta\theta$ from local frame back to global frame. When the right pair is activated first, the robot is at R^{L-}. When the step of the right pair is completed, by substituting the input parameters d, l, φ_1, and φ_2 into (2.4), (2.5) and (2.6), the model output Δx, Δy, and $\Delta\theta$ can be determined. The homogeneous transformation T^R can be used to

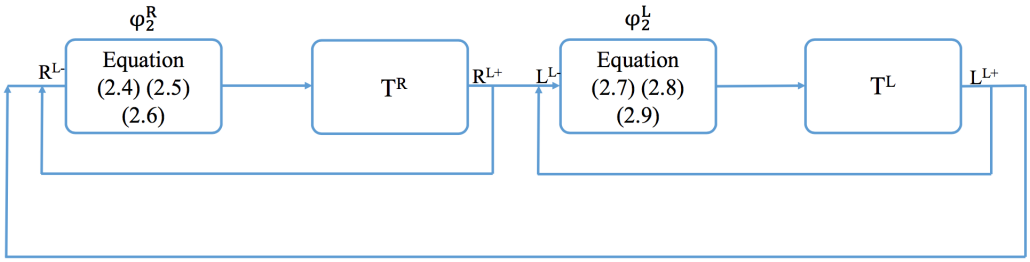


Figure 2.4: The closed–form state propagation of the forward kinematics of the SFM. The order of the right pair and the left pair activation can be arranged arbitrary.

map the local model state to R^{L+}. At this moment, the model is at R^{L+}. The left pair motion can follow the same process.

Once a step motion is completed, the SFM's position at the end of that step becomes the initial position for the motion of the pair activated in the next step. For example, when the right pair is completed, the model is at R^{L+}. The state of the model at the end of right step, R^{L+}, is the initial position at the beginning of the left step, L^{L-}, in other words, we have a reset: $R^{R+} = L^{L-}$.

Figure 2.4 shows a diagram of computation of the state propagation of the SFM. The closed form of state propagation equations supports the design of a feedback law. The order of steps does not need to be constant from right to left and back. It can also be the case that a right step can be succeeded by another right step or a left step by another left step. In fact, the order of pair activation can be arranged arbitrarily.

The inverse kinematics of the SFM also has the closed form state propagation which will be the focus of Chapter 3.

2.5 Application of SFM

The SFM model is not just an abstract mathematical notion, but it can be physically realized. As evidence of that, we describe the fabrication of a legged robot designed to implement the SFM motion (See Figure 2.5).

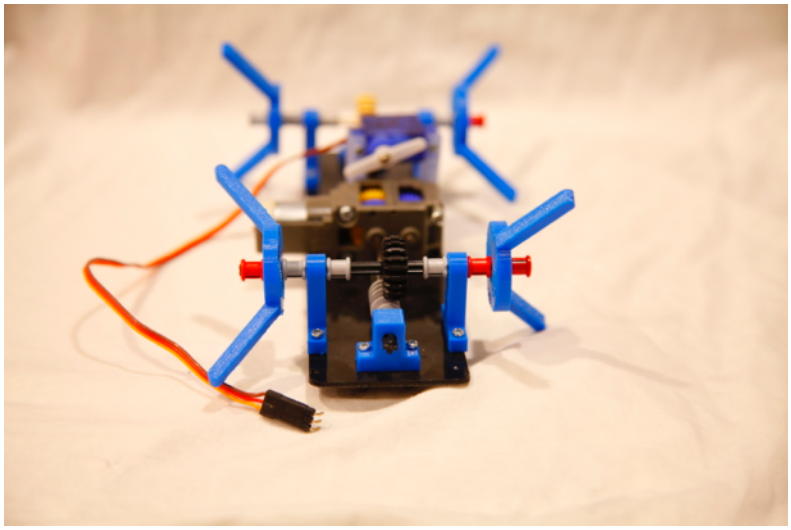


Figure 2.5: Front view of the robot which is designed to realize the SFM.

This robot has 4 rimless wheels and a mid leg. There are two legs in each rimless wheel. The two legs in one wheel are placed in 180 degrees phase difference (See Figure 2.6).

All legs have the same length, same width, and same thickness. The robot has therefore 8 legs in total, with each side having 4 legs. All the robot's legs are 3D printed. The robot is driven by a single DC motor. A worm and gear system is used to transfer the motion from the motor to the wheels. The worm and gear system is also used to reduce the speed of the motor and increase the torque which is applied to the legs (See Figure 2.7).

There is a mid leg that is placed in the robot's geometric center. The mid leg is used to regulate the liftoff angle. When the robot is to turn left or right, the mid leg interrupts the step by lifting the active pair off the ground and thereby enforcing a specific liftoff angle. By changing the values of the liftoff angle, right or left turns are achieved. As mentioned in Section 2.3, the robot's maximum turning angle is 12.7 degrees. The robot's maximum achievable value for the liftoff angle is 45 degrees and thus the liftoff angle can be changed from 0 degrees to 45 degrees.

The robot follows a straight line when the values of the liftoff angles are equal to the values of the touchdown angles. When the robot follows the straight line, the

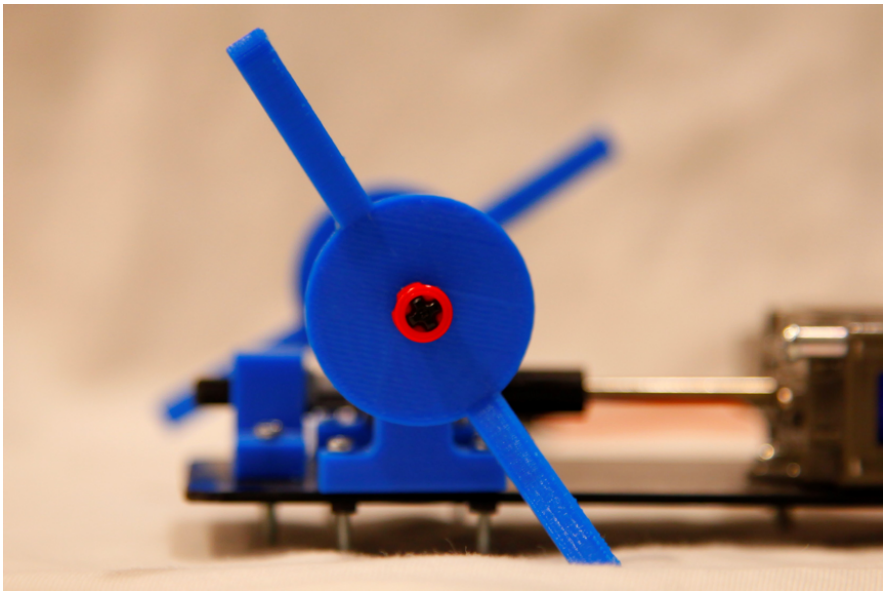


Figure 2.6: Side view of the left front leg. The two legs in one wheel are placed in 180 degrees opposite. The angle between the leg and the ground is 75 degrees.

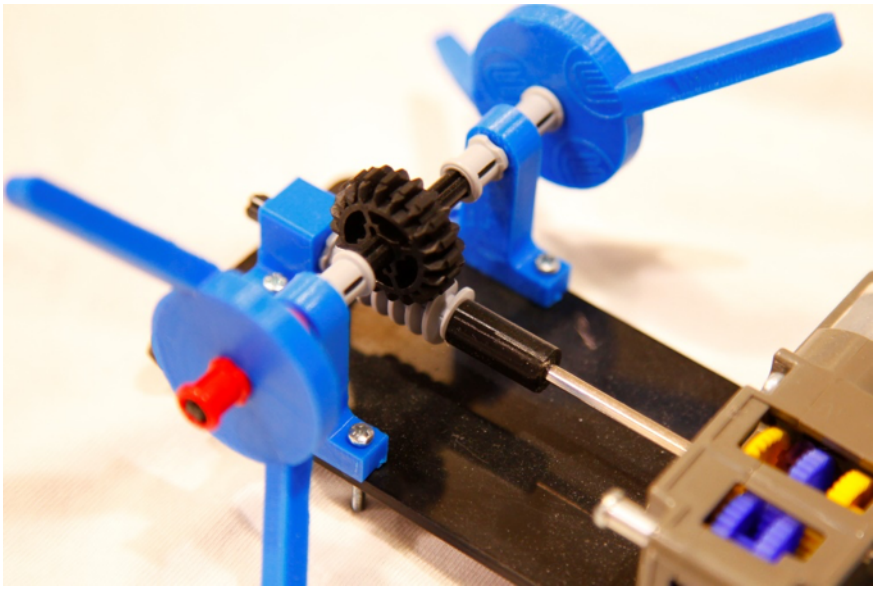


Figure 2.7: The worm and gear system transfers motion from the motor to the wheel and reduces the speed of the legs.

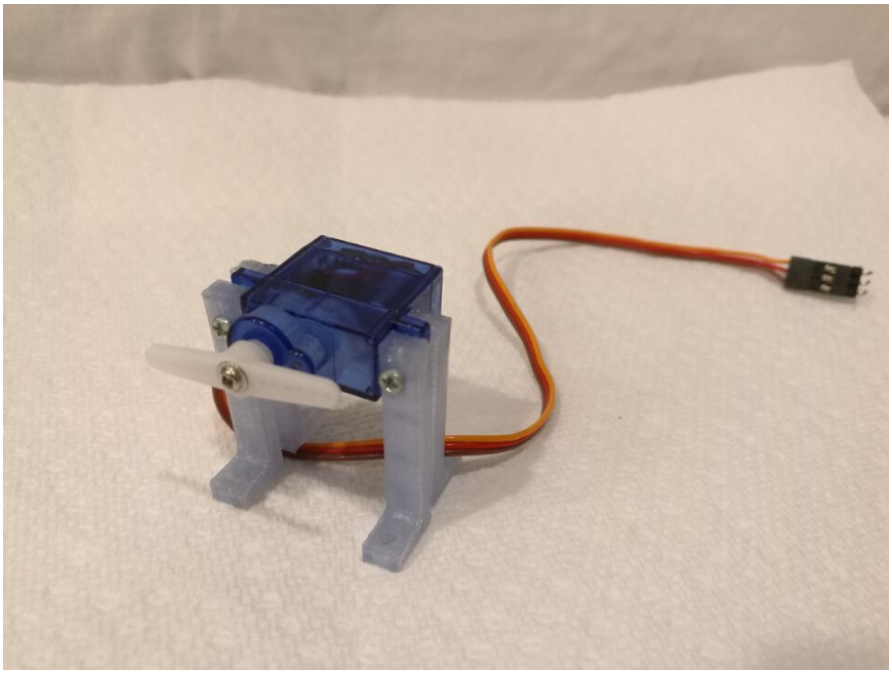


Figure 2.8: The mid leg is controlled by servo actuator. This actuator has to be placed at the mass center of the robot's body.

mid leg does not need to touch down the ground. The robot turns right or left when the values of the liftoff angles are not equal to the values of the touchdown angles. When the robot is to turn right or left, the mid leg needs to touch down the ground so that values of the liftoff angles can be changed. The mid leg is controlled by servo actuator (See Figure 2.8).

2.6 Advantages of SFM model

The SFM model is a very simple kinematic model for planar motion of legged robots. It models the robot's motion through static kinematic maps, without differential equations. The number of variables in the SFM can be reduced to one and because the position is given in closed form, the model supports the development of feedback strategies. Given a reference motion path, the desired displacements in x and y directions, and the change in orientation can be computed, and the SFM can be called to implement them. What this means is that we would need to solve for the values of the liftoff angles so that we can realize the desired displacements in a step by step fashion.

For that, we develop two methods for solving liftoff angle. A method for solving liftoff angle by deriving the inverse kinematics of the SFM, and a way of determining liftoff angle from derivative of navigation function, are provided in next chapter.

Chapter 3

INVERSE KINEMATICS AND SFM MOTION PLANNING

As mentioned in Chapter 1, the navigation function (potential function) can be used to solve the motion planning problem for the SFM. The idea is to build a navigation function around the only variable of the SFM: the liftoff angle. To link the desired direction of motion in the workspace to the liftoff angle, we need to combine the SFM and the navigation functions. This chapter presents a method of computing liftoff angle–inverse kinematics of SFM, and a method of combing the SFM and navigation functions, which enable the construction of a feedback law and the establishment of convergence conditions.

3.1 Inverse Kinematics on the SFM

Once the SFM parameters d, l, φ_1, φ_2, φ_3, and φ_4 are given, the output Δx, Δy, and $\Delta\theta$ can be computed from forward kinematics. Now, we deal with the inverse problem: assuming there is a planned path for the robot, and given the robot's current position, the displacement of robot in the x direction, Δx, the displacement of robot in the y direction, Δy, and the change in direction $\Delta\theta$ from robot's current pose that minimize the deviation from the planned path determine the liftoff angle φ_2. In the forward kinematics of the SFM, the right pair step motion in the local frame can be expressed as:

$$\Delta x = f(d, l, \varphi_1, \varphi_2) \tag{3.1}$$

$$\Delta y = g(d, l, \varphi_1, \varphi_2) \tag{3.2}$$

$$\Delta\theta = h(d, l, \varphi_1, \varphi_2) \tag{3.3}$$

In (3.1), (3.2) and (3.3), $\Delta\theta$ is found as a component of the expressions for Δx and Δy. Once the desired direction of motion, Δx, Δy, and $\Delta\theta$ in the workspace is given, one way to find φ_2 is inverse kinematics of SFM. The equation for the liftoff angle φ_2 can be expressed as:

$$\varphi_2 = k(d, l, \Delta x, \varphi_2) \tag{3.4}$$

$$\varphi_2 = p(d, l, \Delta y, \varphi_2) \tag{3.5}$$

$$\varphi_2 = s(d, l, \Delta\theta, \varphi_2) \tag{3.6}$$

Because of the trigonometric functions in (3.4), (3.5) and (3.6), an explicit solution for φ_2 is not straight forward. On the other hand, because the SFM is a single–variable kinematic model, each of these equations solved for φ_2 will in principle give a different solution. Arbitrary triplets of Δx, Δy and $\Delta\theta$ may not be feasible. If one is given a triple and is asked to find an appropriate φ_2, a reasonable approach would be to find some type of least squares solution. In practice, however, it is the last equation, (3.6), which is the most important of them all. If a single solution needs to be picked, that would be it, cause it determines the direction of motion for the robot. The expression for (3.6) is given in Appendix B.

Because the expression for (3.6) can be very complex, the method of choice here is offering some tradeoffs between accuracy and complexity or computation speed. To simplify the computation of the inverse kinematics of the SFM, namely, computing liftoff angles from Δx, Δy, and $\Delta\theta$, random optimization is used as an alternative to direct solution of the nonlinear algebraic equations (3.4), (3.5) and (3.6). Depending on discretization and available computation time, there will be some error between the φ_2 solution and the estimated one, which has not shown to be a significant problem in our implementation. Fifty values are sampled uniformly from the range of liftoff angles. Each angle results to some value for Δx, Δy, and $\Delta\theta$. We choose the angle that results in one point as our solution for φ_2 which Δx, Δy, and $\Delta\theta$ closer to the ones required to stay on the planned path.

In the next section, we provide another method to find φ_2 which avoids to solve very complex equations (3.4), (3.5) and (3.6).

3.2 Integrating SFM to Navigation Functions

The potential function creates steepest descend using a navigation function generates a collision–free path to the goal. By following that path, the robot will avoid obstacles and move from its starting configuration to its goal configuration [23].

Because the SFM is a 2D model, the potential function will be used under the 2D workspace. Suppose that the robot's workspace is a sphere world, we denote $q_0(x, y)$ the position of the center of the workspace. Borrowed from [23], the spherical obstacles are centered at $q_1(x, y)$, ... $q_n(x, y)$. r_0 is the radius of the outer workspace boundary and r_i is the radius of sphere obstacles. The obstacle distance functions are expressed as [23]:

$$\beta_0(q) = -d^2(q, q_0) + r_0^2 \tag{3.7}$$

$$\beta_i(q) = d^2(q, q_i) - r_i^2 \tag{3.8}$$

$$\beta(q) = \prod_{i=0}^{n} \beta_i(q) \tag{3.9}$$

Note that $\beta(q)$ has zero value on the boundary of the obstacles, and it has positive value at all points in the interior of the workspace. The β is used to form a repulsive–like function. The attractive portion of the navigation function is a power of distance to the goal, i.e.,

$$\gamma_k(q) = (d(q, q_{goal}))^{2k} \tag{3.10}$$

where $\gamma_k(q)$ is zero at the goal and will increase as q moves away from the goal. With (3.7), (3.8), (3.9), and (3.10), and for an appropriate constant k [23], the navigation function on a sphere world can be expressed as:

$$V(q) = \frac{d^2(q, q_{goal})}{[(d(q, q_{goal}))^{2k} + \beta(q)]^{1/k}} \tag{3.11}$$

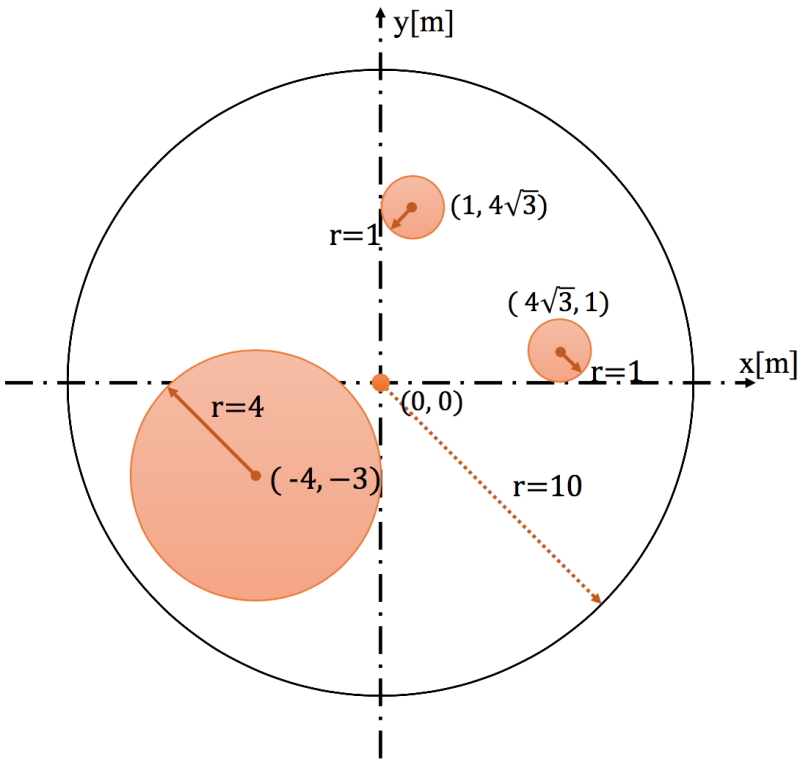


Figure 3.1: The robot's workspace is a sphere world. There are three spherical obstacles in this workspace.

If appropriately tuned, V is guaranteed to have a single minimum at q_{goal}. The boundary of the obstacles has the unitary value, and V varies continuously in free–space.

We simulated a workspace with three obstacles (See Figure 3.1), with the radius of the workspace set at 10, and the center of the workspace at $(0, 0)$. The radius of the first obstacle is 4 and the center of the obstacle is at $(-4, -3)$. The radius of the second obstacle is 1 and the center of the obstacle is at $(1, 4\sqrt{3})$. The radius of the third obstacle is 1 and the center of the obstacle is at $(4\sqrt{3}, 1)$.

Figure 3.2 shows the navigation function in x and y coordinates for $k = 3$. The goal configuration has the minimal potential value of zero, and the position of the goal is at $(5, -5)$.

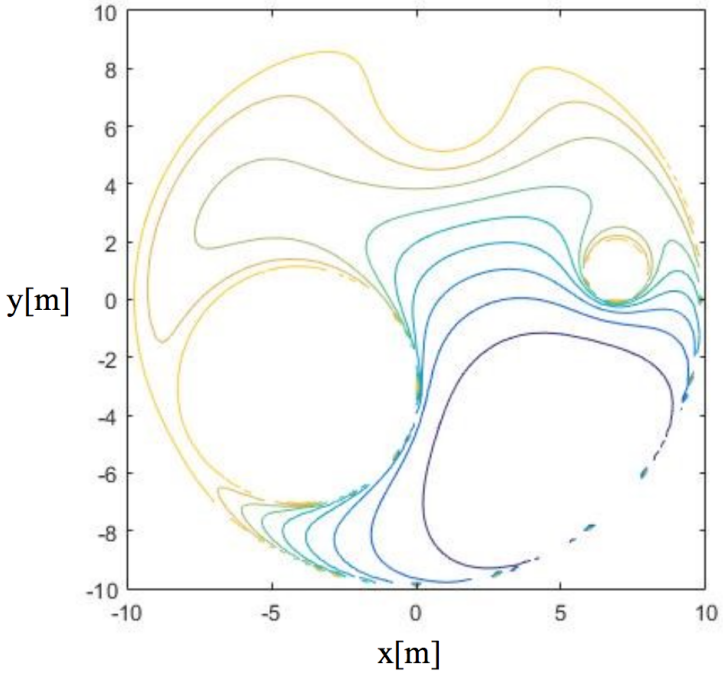


Figure 3.2: The navigation function for the work space with three obstacles for k=3. The robot's starting position can be set in any place.

The gradient is a vector which points in the direction that locally maximally increases the potential value. The method of gradient descent is straight forward: In principle, when the robot takes a step in the direction opposite the gradient, the value of the function is decreased. This can be done in an iterative fashion, with the position of the robot updated in each step. This process is repeated again and again until the robot reaches the goal.

For the legged robots which use the SFM model as their locomotion template, the opposite gradient direction that suggests a desired Δx and Δy may not feasible because of the SFM's turning limitations. In some cases, the robot may not be able to take the step by following the opposite direction of gradient at all (The SFM also only moves forward). As mentioned in Chapter 2, the maximum turning angle of the SFM is practically 12.7 degrees. To go back, the SFM has to transcribe a complete circle with that maximum curvature.

Gradient descent can not be directly applied to the legged robot using the SFM locomotion template, but it can still be used to determine φ_2. We have the potential function with respect to Δx and Δy, but forward kinematics can express both of those with respect to φ_2, the liftoff angle. Since we have a function Δx and Δy with respect to liftoff angle, the derivative of V with respect to Δx and Δy can be written as:

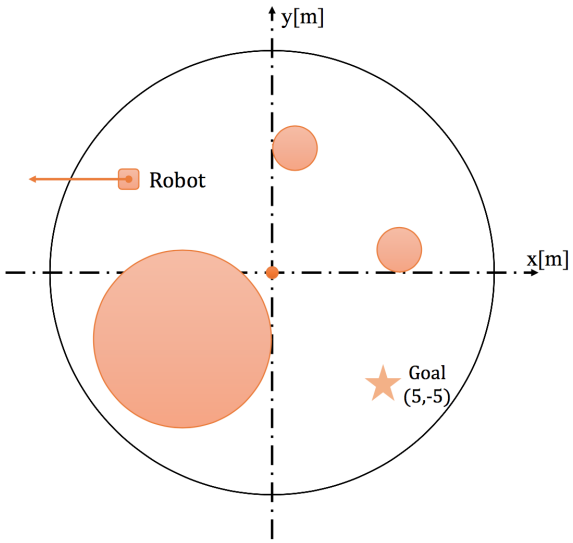
$$\frac{dV}{d\varphi_2} = \frac{dV}{dx} \cdot \frac{dx}{d\varphi_2} + \frac{dV}{dy} \cdot \frac{dy}{d\varphi_2} \tag{3.12}$$

The question is now what value of φ_2 gives this derivative the minimum (hopefully negative) value. If this minimizing value for φ_2 is in the prescribed range, then the local motion direction can be followed by the robot. Because $\frac{dV}{d\varphi_2}$ can be very complex, the random optimization method mentioned in Section 3.1 is used to improve computational processing speed. The 50 different values of φ_2 are sampled from a uniform distribution and yield 50 different values for $\frac{dV}{d\varphi_2}$. Depending on whether one activates the left pair or the right pair, different Δx and Δy are produced, and therefore from one step to next, there are two different functions $\frac{dV}{d\varphi_2}$ that must be analyzed: one is for the left pair, and the other is for the right pair. We pick the lowest value of $\frac{dV}{d\varphi_2}$

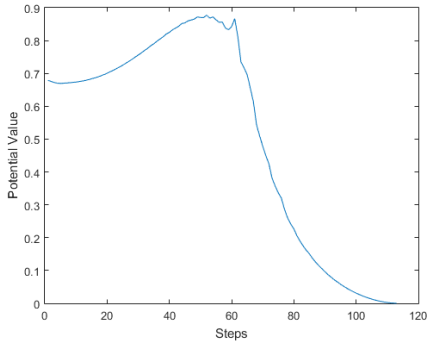
to be the solution of the φ_2. Since φ_2 is determined, the Δx, Δy, and $\Delta\theta$ of the next step can be determined and which pair need to active is known.

In some cases, we find that from one step to next, the values of $\frac{dV}{d\varphi_2}$ for both pairs are larger than 0, because of the SFM's turning limitations. This means that it is currently impossible to let the SFM moves to a configuration that decreases the potential compared to the current configuration. In some other cases, we also find that a myopic choice of the type of step may not be best. For example, activating the left pair may decrease the potential value from the current configuration, but looking three steps ahead may reveal that a temporary increase now can allow a more drastic decrease later. Thus, the active pair is determined by performing a binary search several steps ahead (5 is chosen) to determine, which of the two pairs, left or right, decreases the potential function the most in a 5 step horizon. The number of searching steps can be set according to the computation time that can be afforded.

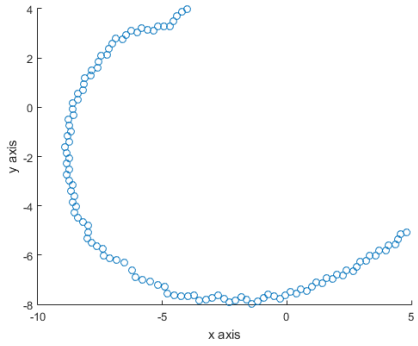
We present two simulation examples for the information of the workspace mentioned in Section 3.2. The robot's body length is 0.14 and the robot's leg length is 0.1. The robot's touchdown angle is set to be 45 degrees as mentioned in Section 3.2. The planned paths of the robot starting in same position but opposite direction are shown in Figure 3.3 and Figure 3.4:



(a)

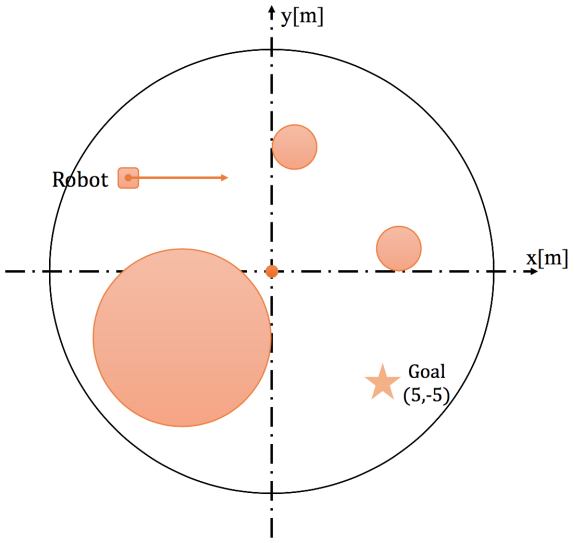


(b)

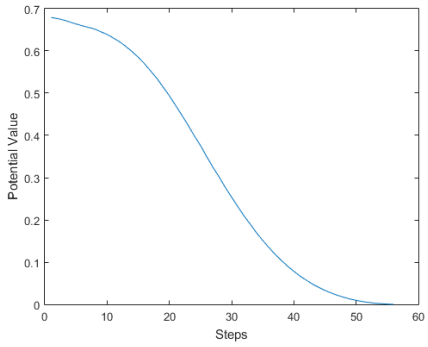


(c)

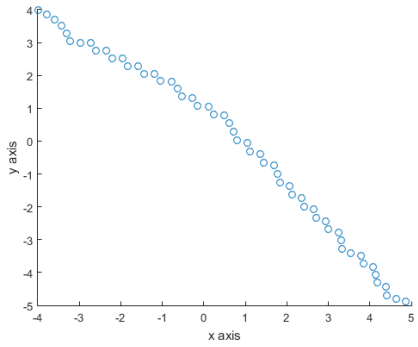
Figure 3.3: (a) shows the robot starts at $(-4, 4)$. The robot faces to the negative direction of x axis. (b) shows the potential value in each step. After 110 steps, the potential value approaches to zero, (c) shows the robot's path which is given by gradient descent.



(a)



(b)



(c)

Figure 3.4: (a) shows the robot starts at $(-4, 4)$. The robot faces to the positive direction of x axis; (b) shows the potential value in each step. It is a simple case that the potential functions always decrease; (c) shows the path of the robot. The starting configuration is at $(-4, 4)$.

Figure 3.3 shows the evolution of the potential (b) and of the configuration (c) when the robot starts at $(-4, 4)$. The direction of the robot is towards the negative direction of x axis. The goal is at $(5, -5)$. The y axis in Figure 3.3(b) represents the potential value. The x axis in Figure represents the number of steps. After 114 steps, the potential value decreases to 0 and the robot arrives to the goal.

Figure 3.4 shows that the robot starts from the same point but facing opposite. The planned path in Figure 3.4(c) is different from the planned path in Figure 3.3(c). The reason of this is because the locomotion limitations of the SFM now do not constrain it so much from achieving negative derivative values. So, the binary search for a 5 future steps horizon gives a different result. After 56 steps, the robot arrives to the goal. From this simulation, we find that our method can be applied in a sphere world. However, this simulation does not consider uncertainty. Thus, another simulation example is provided, taking into account the uncertainly that a real miniature legged robot would exhibit [1].

legged robots are subjected to many sources of uncertainty, including parametric uncertainty from unmodeled frictional force. We provide the following simulation that solves the SFM robot motion planning problem with normal distributed uncertainties in a more complex environment, and a comparison of the results of our simulation with previous work [1] is provided.

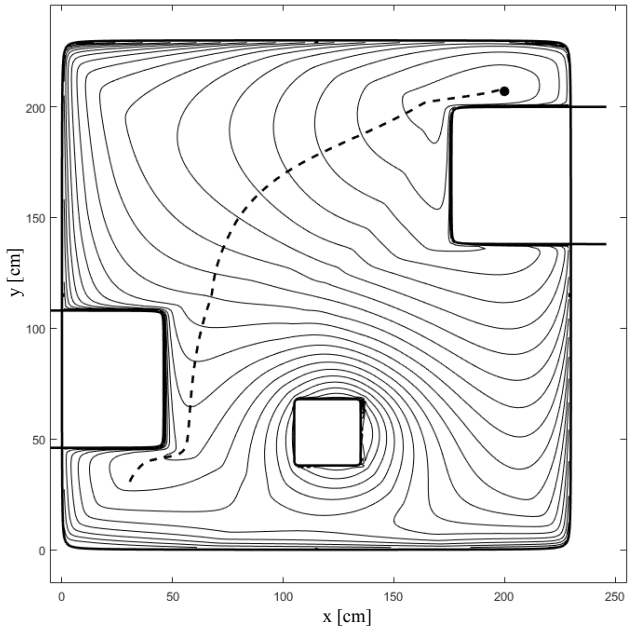


Figure 3.5: The contour plot of the potential field generated by the navigation function. A collision free path between robot configuration and goal configuration can be planned by using gradient descent in a potential field.

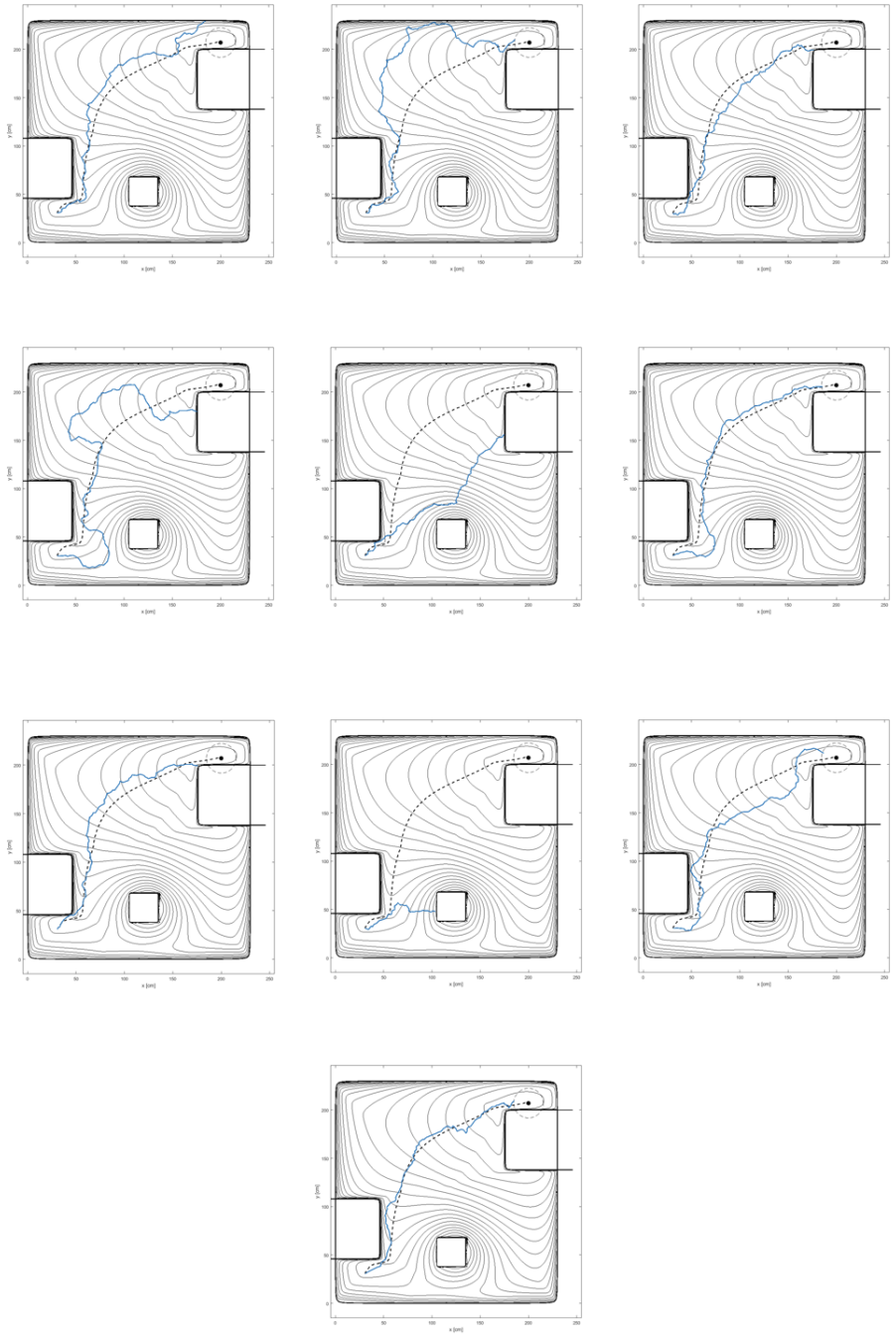


Figure 3.6: Simulated response of the system commanded to follow a desired path, when uncertainty affects its state at the end of every single step. The dashed path shows the path of the gradient descent in the absence of uncertainty. Robot reached the target 4 times out of 10. $\sigma_x = \sigma_y = 1$cm, $\sigma_\theta = 5$ degrees.

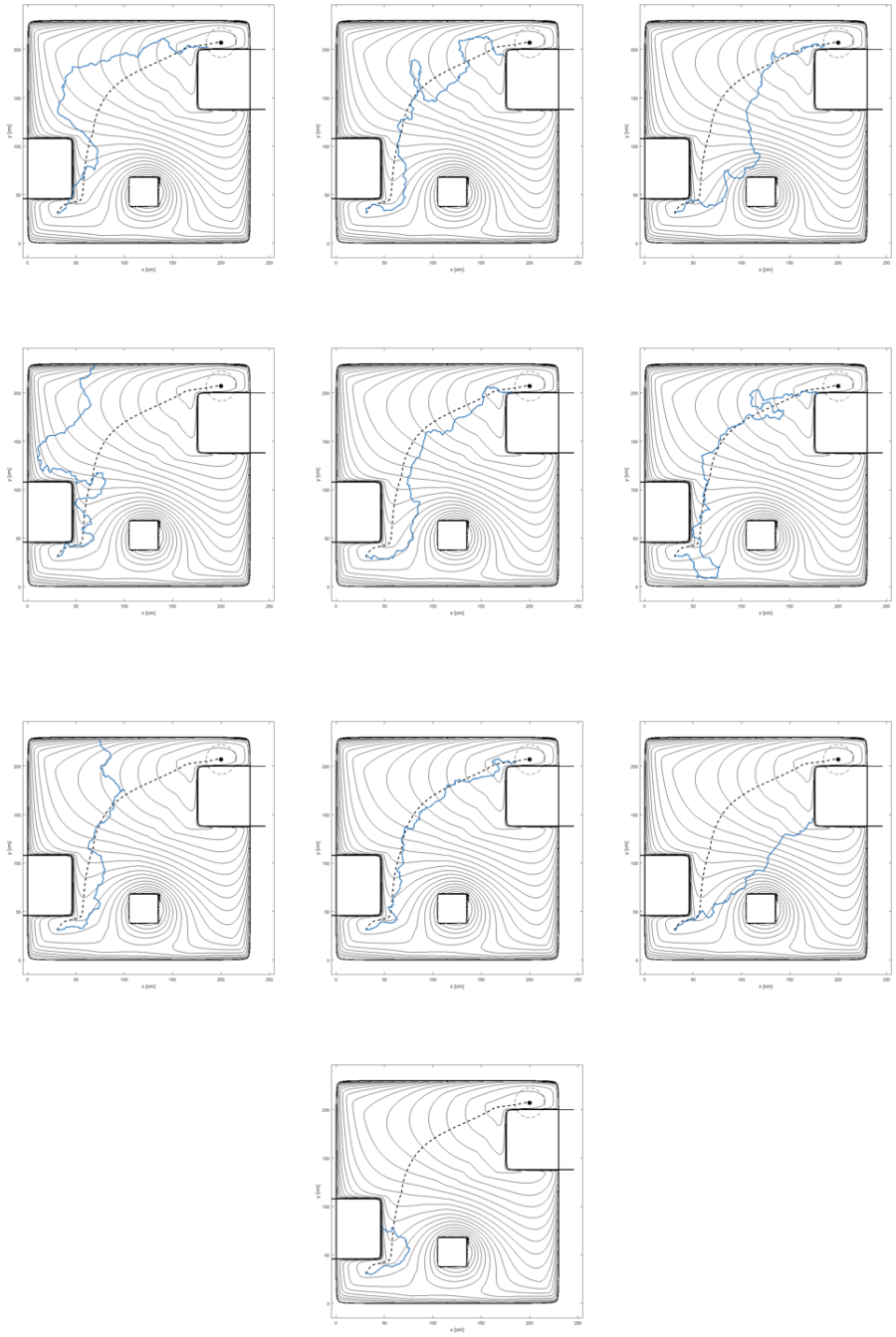


Figure 3.7: Simulated response of the system commanded to follow a desired path, when uncertainty affects its state at the end of every single step. Again, the dashed path shows straightforward gradient descent without uncertainty. Robot reached the target 3 times out of 10. $\sigma_x = \sigma_y = 2$cm, $\sigma_\theta = 10$ degrees

First, we use a navigation function [18] to generate a collision free path between the robot configuration and goal configuration in a more complex workspace. The modeled environment, as well as the contour plot of the potential field generated by the navigation function, is shown in Fig. 3.5. Setting a static destination, the global convergence is guaranteed from the properties of the navigation function [18]. At every time instance, the navigation function provides a desired direction that the robot should proceed by calculating its negative gradient. So, the kinematics of the robot can be taken into consideration in a relatively independent pattern, which is described below.

The inverse kinematics of the SFM is used to solve the liftoff angle when the robot change orientation, as denoted $\Delta\theta$, and given by the gradient descent of the navigation function. Because of the robot's motion constraints, the closest available $\Delta\theta$ is chosen if the desired change of orientation is not reachable. Instead of explicitly solving the inverse kinematics, we lookup the liftoff angle from a pre-calculated table using the forward kinematics of the robot. Once the liftoff angle is determined, Δx, Δy and $\Delta\theta$ are calculated based on the forward kinematics. This process represents a single step iteration.

Uncertainties are introduced into every iteration of a step as described above. The updated x, y, and θ in the global frame are perturbed with normal distributed random numbers. For the variances, two trials of example variances are tested. For trial 1, σ_x and σ_y are equal to 1cm and σ_θ is equal to 5 degrees. For trial 2, σ_x and σ_y are equal to 2cm and σ_θ is equal to 10 degrees.

The percentage of success for the paths are reported. The results of 300 simulation runs for each case reveal that the percentage of success for the paths in trial 1 is 25.6% and 7.6% in trial 2. The success rate of trail 2 is smaller than trail 1, because of the larger variance. The actual paths, with the uncertainty considered, are shown in Figure 3.6 to Figure 3.7. The desired path as determined by the potential field of our workspace and is shown in Figure 3.5 as a dash curve.

Finally, the results are compared with Karydis's work [1], which used the same uncertainties and the same robot configurations, but with a different navigation method

(RRT). With the same uncertainty cases and in a similar but a bit more complex environment, [1] reports trial 1, with a 100% success rate, and an 85% success rate in trial 2. We see that, even with a simpler workspace, the successful rate of our simulations using the proposed method is not as good as those proposed in Karydis's work [1]. The most critical reason is, in Karydis's case, both the value of liftoff angle and touchdown angle are treated as variables. To simplify the computation process, the only variable in the kinematics of the SFM in this thesis is liftoff angle. Changing both liftoff angle and touchdown angle provide additional robot turning space.

3.3 Feedback and Path Re–Planning

Both (3.6) from Section 3.1 and (3.12) from Section 3.2 provide the method to find the solution of φ_2. Section 3.2 also provides information about which pair needs to be activated. As in Section 2.3 , we define a 4×4 homogeneous transformation matrix T^R that maps Δx, Δy, and $\Delta\theta$ from the local frame back to the global frame. The computation flow and state propagation are shown in Figure 3.8. After we have the solution of φ_2 and the information about which pair to be activated, and with d, l, φ_1 constants, we can get Δx, Δy, and $\Delta\theta$ for the next step. If we need to activate the left pair, Δx, Δy and $\Delta\theta$ can be computed from (2.7), (2.8) and (2.9). If we need to activate the right pair, Δx, Δy, and $\Delta\theta$ can be computed from (2.4), (2.5) and (2.6). The homogeneous transformations T^R and T^L can be used to map the local model state to R^{L+} or R^{R+}. Then with the new model state being R^{L+} or R^{R+}, we can substitute the robot's new position in x and y as the input parameters to (3.12) to find the solution of φ_2 for the next step. The process will be repeated until the robot arrives to the goal, or the algorithm reports that it can not proceed due to the kinematic constraints.

The navigation function naturally provides feedback. There is no need for re–planning.

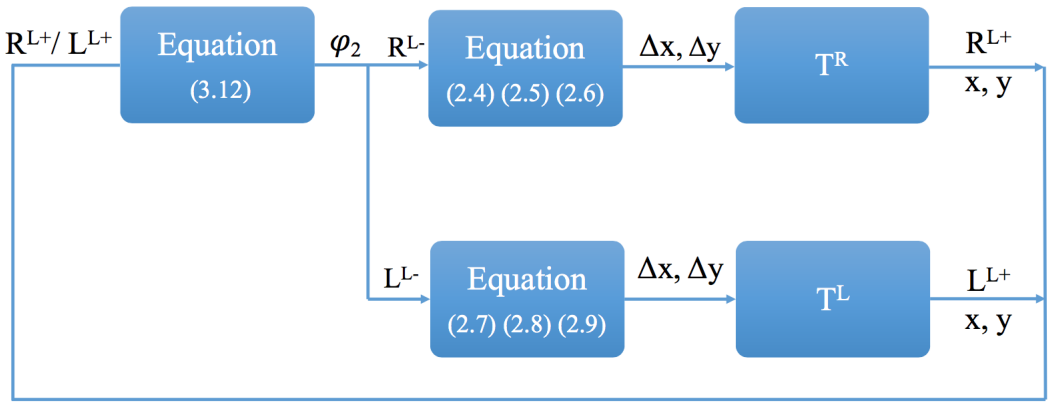


Figure 3.8: The closed–form state propagation of the inverse kinematics of the SFM. The order of the right pair and the left pair activation can be arranged arbitrary. The liftoff angle can be computed from (3.12).

3.4 Limitations of Inverse Kinematics and SFM Motion Planning

As we mentioned in Section 3.1, an explicit solution for φ_2 using (3.4), (3.5), and (3.6) is not straight forward because of the trigonometric functions. Also, the expression for (3.6) that is given in Appendix B is very complex. The computation of solving (3.6) to find φ_2 is intractable.

As we mentioned in Section 3.3, due to the locomotion limitations of the SFM, direct gradient descent may suggest motion directions which are infeasible for the SFM. When combining the navigation function with the SFM, however, we produce a navigation strategy that respects the kinematics of a legged robot. The kinematics of the SFM suggests a minimum turning radius and allows only forward motion. The turning radius is dependent on d and l. If the robot is close to the workspace boundaries, the robot may not be able to avoid collision. Algebraic conditions that safeguard against this, or inform on the occurrence of such situation are discussed in Chapter 4. The limitation of the kinematics of the SFM that prevents the robot from moving in reverse may be relaxed through future extensions of the SFM template.

Chapter 4

CONVERGENCE CONDITIONS

Kinematic constraints–finite motion primitives, turning radius limitations, etc–may render the motion planning problem infeasible, even if navigation functions offer directions for motion. For example, if the robot's position is very near to the boundary of the workspace, there may not exist a path from the robot's current position to the goal because the robot may not have enough space to maneuver. Requiring that every step reduces the navigation potential may be overly restrictive; the robot may be allowed to temporarily increase its potential while maneuvering into a position from which it can effectively start decreasing it again. We search for such opportunities using integral versions of Lyapunov–like theorems. The conditions derived based on such theorems are still sufficient for convergence but not necessary. If they are satisfied, we can claim that there exists a path from the robot's current position to the goal.

4.1 Convergence Conditions Using Integral Version of Lyapunov–Like Theorems

Borrowed from [24], following [25], we treat SFM as a switching system with dwell time

$$\dot{x} = f_{\sigma(t)}(x) \tag{4.1}$$

where $x \in \mathbb{R}^n$ is the model's state, and $\sigma : \mathbb{R}_+ \in \mathbb{N}_+;\ t \mapsto p$ is a piecewise constant switching signal that indicates the vector field from a parameterized family $\left\{f_p : p \in \mathbb{N}_+\right\}$ which is active at a given time instant $t \in \mathbb{R}_+$. We assume that each $f_{\sigma(t)}$ satisfies the standard Lipschitz continuity conditions for all $p \in \mathbb{N}$, and the solutions of (4.1) are defined in the Caratheodory sense as in [25, 24]. Of course the SFM's forward

kinematics are available to us in closed form. Straight forward time differentiation of (2.4)–(2.9) in this case can provide (4.1) with p=2. The switching signal belongs in a class denoted $\sigma \in S^{+}_{dwell}$ (cf.[26]) defined as follows.

Definition 1. (switching signals) *[24] The set S^{+}_{dwell} contains all piecewise constant functions from $\mathbb{R}_+$ onto $P \subset \mathbb{N}_+$, for which the inverse image of every $p \in P$ is a (possibly infinite) collection of intervals in $\mathbb{R}_+$ of measure no less that $\tau_D > 0$ and no more than $T_D < \infty$.*

Theorem 1. *[24] Let $V(x)$ be a differentiable, positive definite function in a domain $\mathcal{D} \subseteq \mathbb{R}^n$ that contains the origin. For some class–$\mathcal{K}$ function [24] α and γ, and within each of the switching intervals $\left(\tau_i, \tau_{i+1}\right]$ of some $\sigma \in S^{+}_{dwell}$, assume that every solution of (4.1) satisfies*

$$\sup_{t\in(\tau_i,\tau_{i+1}]} \left\{ \frac{\partial V}{\partial x} f_{\sigma(t)}(x) \right\} < \frac{\alpha(V\big(x(\tau_i)\big))}{\tau_{i+1} - \tau_i} \tag{4.2}$$

$$\sup_{t\in(\tau_i,\tau_{i+1}]} \left\{ \frac{\partial V}{\partial x} f_{\sigma(t)}(x) \right\} < \frac{1 - (V\big(x(\tau_i)\big)}{\tau_{i+1} - \tau_i} \tag{4.3}$$

$$V\big(x(\tau_{i+1})\big) - V\big(x(\tau_i)\big) \leq -\gamma(\|x(\tau_i)\|) \tag{4.4}$$

Then all trajectories of (4.1) starting in $\mathcal{D}$ asymptotically converge to the origin. Moreover, if $\mathcal{D} = \mathbb{R}^n$, the system is globally asymptotically stable.

For this analysis, we will relax the notion of switching time as used in the theorem. For us, "switching times" will be time instances between collections of consecutive steps, not the time of transition between, say, a single left step to a following right step. If one considers the minimum time period required to complete the shortest single step in the model, that value sets the dwell time for the system. We will refer to the time interval between τ_i and τ_{i+1} as the "horizon". For computational expediency, this horizon is upper bounded to 10 steps.

In convergence condition (4.2), the

$$\sup_{t\in(\tau_i,\tau_{i+1}]} \left\{ \frac{\partial V}{\partial x} f_{\sigma(t)}(x) \right\}$$

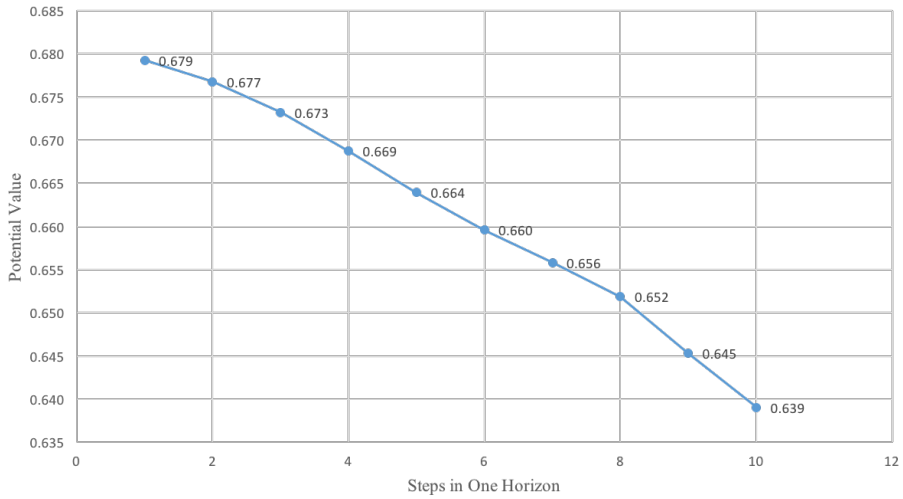


Figure 4.1: Each point in the figure is the potential value in consecutive robot steps.

is understood in the discrete–time state evolution of the SFM as the maximum increase of the potential (navigation function) in a single step over the whole horizon. In condition (4.4), $V\big(x(\tau_{i+1})\big) - V\big(x(\tau_i)\big)$ is the difference in the potential between two switching times. γ is a $\mathcal{K}$–class function and the $\|x(\tau_i)\|$ is the variable of the function γ. In convergence condition (4.3), $1-V\big(x(\tau_i)\big)$ indicates about how much the potential can be increased before a collisions–recall that the navigation function is one on the workspace boundary.

To determine the supremum shown above, we examine the changes in potential values between subsequent steps over a horizon of 10 steps. Figure 4.1 shows a sequence of potential values for a horizon of 10 steps, while Figure 4.2 shows another evolution over five horizons. Within a single horizon, we can find the largest potential difference between steps and set the slope of this in–between–steps line segment equal to

$$\sup_{t\in(\tau_i,\tau_{i+1}]} \left\{ \frac{\partial V}{\partial x} f_{\sigma(t)}(x) \right\}$$

When (4.2), (4.3), and (4.4) are all satisfied, we allow the robot to proceed with its motion for that horizon.

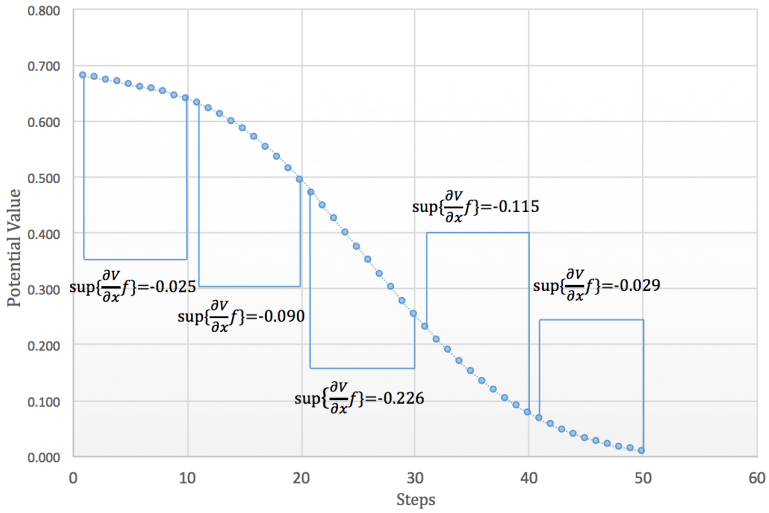


Figure 4.2: Potential value variation over a period of give horizon (50 steps).

4.2 Embedding the Convergence Conditions into the Planning Algorithm

To show the combined use of the conditions (4.2)–(4.4) with the navigation function having the SFM inverse kinematics, we use the same workspace that mentioned in Chapter 3. The radius of the workspace is 10, the center of the workspace is at $(0, 0)$. The radius of the first obstacle is 4 and the center of the obstacle is at $(-4, -3)$. The radius of the second obstacle is 1 and the center of the obstacle is at $(1, 4)$. The radius of the third obstacle is 1 and the center of the obstacle is at $(4, 1)$. The goal is at $(5, -5)$. We provide two examples of embedding convergence conditions into the planning algorithm. In the first example, the robot starts at $(-4, 4)$. The direction of the robot towards the positive direction of the x axis. Figure 4.3(b) shows an example of (4.2) and (4.3) being satisfied over a period of five horizons, with each horizon having 10 steps. As shown in Figure 4.3(b), the maximum increase of the potential function in one horizon is always less than the values of the $\mathcal{K}$–class function α and the values of the function $1 - V\big(x(\tau_i)\big)$. The Figure 4.3(c) shows that the condition (4.4) is satisfied. The red line is the negative of the $\mathcal{K}$–class function. As shown in Figure 4.3(c), all differences of potential functions between neighboring horizons are less than the value

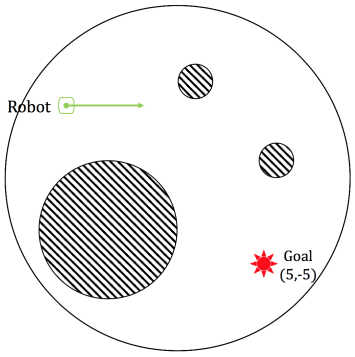
of $-\gamma$. Since (4.2), (4.3), and (4.4) are satisfied, we can allow the robot to implement the motion plan for the current horizon.

In some cases, the convergence conditions are not satisfied. This is illustrated in the second example. We assume now that the robot starts at $(-4, 4)$, the same starting position as the previous example. The initial direction of the robot is now towards, however, set the negative direction of the x axis. Figure 4.4(b) shows that (4.3) is not satisfied when we look ahead for eleven horizons. Figure 4.4(c) shows that the (4.4) is not satisfied. The red line is the negative of the $\mathcal{K}$–class function. Since (4.2) is satisfied, but (4.3) and (4.4) are not satisfied, the algorithm halts and reports that there is no feasible motion plan. The algorithm's conditions in this case are too conservative. We have seen in Chapter 3 (Figure 3.3) that from that configuration, there is indeed a path from the robot's current position to the goal.

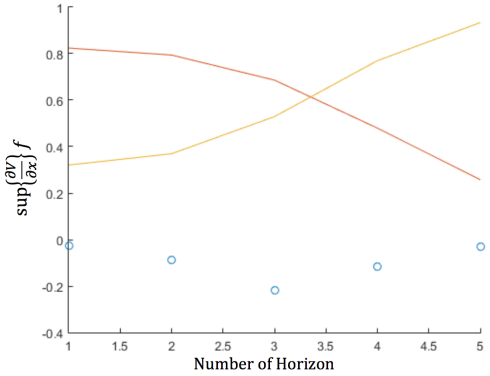
4.3 Limitations of Convergence Conditions

As mentioned in Section 4.2, the convergence conditions may not always be satisfied, even when there are feasible solutions. The problem is exacerbated when the potential function increases at a very slow speed, and a small horizon is used. As we assume that there are 10 steps in one horizon, the differences of the potential functions between the final step and the first step in the same horizon may have positive values. If the horizon is extended, this problem may be avoided. On the other hand, a large interval may make the construction of the required $\mathcal{K}$–class functions more difficult.

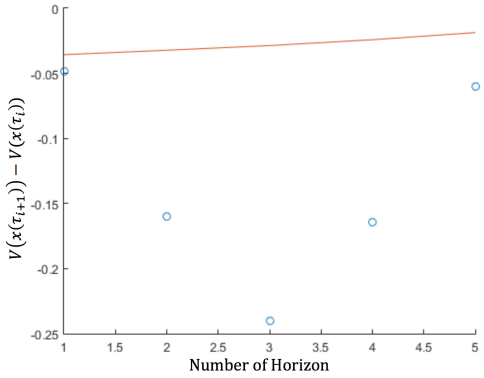
The complexity of the workspace may contribute to the violation of the convergence conditions.



(a)

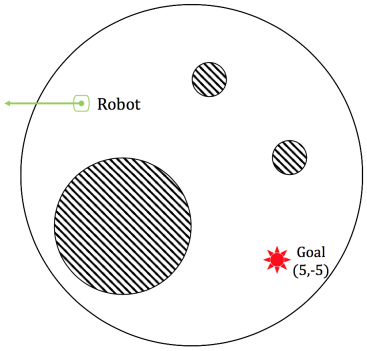


(b)

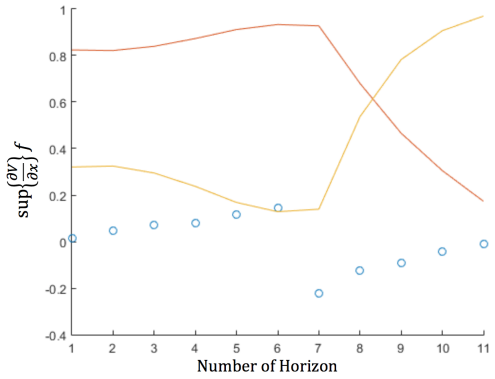


(c)

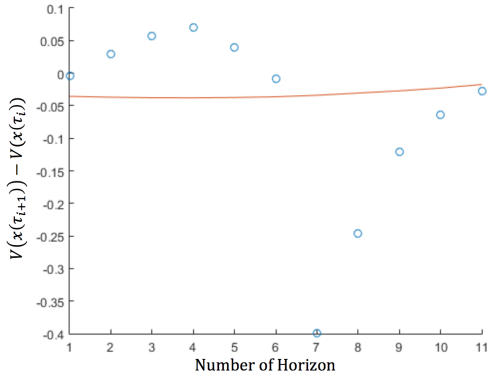
Figure 4.3: (a) shows the workspace and the robot's initial configuration. The red line in (b) is the $\mathcal{K}$– class function α, and the yellow line in (b) is the function of $1-V\big(x(\tau_i)\big)$. The points in (b) are the values of the maximum increase of the potential function, i.e., $(\tau_{i+1}-\tau_i)\sup\left\{\frac{\partial V}{\partial x}f_{\sigma(t)}(x)\right\}$. The red line in (c) is $-\gamma$, and the individual points mark the difference $V(\tau_{i+1})-V(\tau_i)$.



(a)



(b)



(c)

Figure 4.4: (a) shows the workspace and the robot's initial configuration. The red line in (b) is the $\mathcal{K}$–class function α, and the yellow line in (b) is the function of $1-V\big(x(\tau_i)\big)$. The points in (b) are the values of the maximum increase of the potential function, i.e., $(\tau_{i+1}-\tau_i)\sup\left\{\frac{\partial V}{\partial x}f_{\sigma(t)}(x)\right\}$. The red line in (c) is negative of the $\mathcal{K}$–class function.

Chapter 5

CONCLUSIONS, LIMITATIONS AND CHALLENGES

This thesis provides a method that combines navigation functions with the SFM kinematics, and uses the results to develop a new feedback law for navigation of multi–legged robots in cluttered planar environments. In this chapter, we conclude with a short summary of the contributions of this work, that include the methods we use to combine the SFM with the navigation functions and how we optimize the SFM path in terms of the model's liftoff angles. We also mention some advantages and limitations of our method. The reported navigation method and optimization methods can also be applied in other areas of robotics for solving navigation problems in a given environment.

5.1 Conclusions

There is now a navigation method for legged robots modeled based on the SFM that, as long as certain (conservative) conditions are satisfied, provides a feedback law with provable convergence in cluttered planar environments. Chapter 2 reviews SFM as a kinematic template with no differential equations with known, closed–form forward kinematics [1]. The motion planning method developed here, however, requires to combine the SFM with navigation function. The method of combining SFM and navigation function is reported in Chapter 3. The details of the navigation method for the legged robot that provides a feedback law and provable convergence in cluttered planar environments are reported in Chapter 3 and Chapter 4. We also develop the inverse kinematics of the SFM, as a method of solving liftoff angle. If the legged robots' displacements in x and y directions are given, the inverse kinematics of the SFM maps the robot's displacement to the required the liftoff angle.

In this thesis, we designed an algorithm that locally optimizes the SFM path in terms of the liftoff angles. If motion planning goal is encoded as bring the function to zero, the path planning problem reduces to a problem of single–dimensional function minimization. Randomized algorithms are used for optimization to estimate the required liftoff angles. This method can also be applied to general motion planning problems of the legged robots which have locomotion limitations.

Because the SFM kinematics do not require numerical integration, integral Lyapunov–like, convergence conditions can be directly embedded into the planning algorithm. The convergence conditions are presented in Chapter 4.

5.2 Limitations

There are still some limitations that can be relaxed in the future. For the convergence conditions, the switching interval $\tau_{i+1} - \tau_i$ needs to be selected carefully; a small interval may exclude possible solutions and a large interval may restrict the construction of $\mathcal{K}$–functions needed for the convergence conditions.

The development of our navigation motion planning method can also be continued in the future. We use the binary search to search several steps ahead and choose the best result which decreases the potential value the most. In some cases, we find that search for 5 steps would not provide the best result. By searching over more steps, the method may provide different results. The more steps we search, the better result we may get, at the expense of computation time. The required time of the binary search will exponentially increase.

The SFM still has locomotion limitations. In the current SFM formulation, legged robots are not able to go backward. The model can be improved in the future.

5.3 Challenges

Because the SFM imposes constraints in locomotion, gradient descent may not provide feasible directions of motion. As mentioned in Chapter 2, the maximum angle that the robots can turn is around 12.7 degrees. If the robot needs to turn 120 degrees

in one step, so that the potential function decreases the most, there is no liftoff angle that can do that.

The current method is used to solve the legged robots navigation problems in a given environment with the circular shape obstacles, but it should be straight forward to extend the method to star–shape environments.

BIBLIOGRAPHY

[1] Konstantinos Karydis. *A data-driven hierarchical framework for planning, navigation, and control of uncertain systems: applications to miniature legged robots*. PhD thesis, University of Delaware, 2015.

[2] Matt Travers, Alex Ansari, and Howie Choset. A dynamical systems approach to obstacle navigation for a series-elastic hexapod robot. In *Proceedings of the IEEE 55th Conference on Decision and Control*, pages 5152–5157.

[3] David Zarrouk, Duncan W Haldane, and Ronald S Fearing. Dynamic legged locomotion for palm-size robots. In *Proceedings of the SPIE Defense Security*, pages 94671S–94671S, 2015.

[4] N. J. Kohut, D. Zarrouk, K. C. Peterson, and R. S. Fearing. Aerodynamic steering of a 10 cm high-speed running robot. In *IEEE International Conference on Intelligent Robots and Systems*, pages 5593–5599, 2013.

[5] Aaron M. Hoover, Erik Steltz, and Ronald S. Fearing. RoACH: An autonomous 2.4g crawling hexapod robot. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 26–33, 2008.

[6] David Zarrouk and Ronald S. Fearing. 1STAR, A one-actuator steerable robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2569–2569, 2014.

[7] David Zarrouk, Andrew Pullin, Nick Kohut, and Ronald S. Fearing. STAR, a sprawl tuned autonomous robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 20–25, 2013.

[8] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Aral, Noriho Koyachi, and Kazuo Tanie. Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, 17(3):280–289, 2001.

[9] Martin Wermelinger, Remo Diethelm, Philipp Kruesi, Roland Siegwart, and Marco Hutter. Navigation Planning for Legged Robots in Challenging Terrain. *proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1184–1189, 2016.

[10] Ming-fai Fong. *Mechanical Design of a Simple Bipedal Robot*. PhD thesis, Massachusetts Institute of Technology, 2005.

[11] Mohamad Shafiee Motahar, Sushant Veer, and Ioannis Poulakakis. Composing limit cycles for motion planning of 3D bipedal walkers. In *Proceedings of the IEEE 55th Conference on Decision and Control*, pages 6368–6374. IEEE, 2016.

[12] J. Schmitt, M. Garcia, R. C. Razo, P. Holmes, and R. J. Full. Dynamics and stability of legged locomotion in the horizontal plane: A test case using insects. *Biological Cybernetics*, 86(5):343–353, 2002.

[13] Nicolas Perrin, Christian Ott, Johannes Englsberger, Olivier Stasse, Lamiraux Florent, and Darwin Caldwell. Continuous Legged Locomotion Planning. *IEEE Transactions on Robotics*, 33(1):234–239, 2016.

[14] Lydia E. Kavraki, Petr Švestka, Jean Claude Latombe, and Mark H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[15] Kris Hauser. *Motion planning for legged and humanoid robots*. PhD thesis, Stanford University, 2008.

[16] J Kuffner and S M LaValle. RRT-connect: An efficient approach to single-query path planning. *IEEE International Conference on Robotics and Automation*, pages 995–1001, 2000.

[17] Steven M. LaValle and James J. Kuffner. Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

[18] Elon Rimon and Daniel Koditschek. Exact Robot Navigation Using Artificial Potential Functions. *IEEE Robotics and Automation*, 8(5):501–518, 1992.

[19] Roland Philippsen and Roland Siegwart. An interpolated dynamic navigation function. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2005, pages 3782–3789, 2005.

[20] P. Holmes, Poincaré, and Celestial Mechanics. Dynamical Systems Theory and Chaos. Technical report, Cornell University, 1990.

[21] E. T. Whittaker. *A treatise on the Analytical Dynamics of Particles and Rigid Bodies*. Cambridge, University Press, 1917.

[22] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[23] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrum. *Principles of Robot Motion*. Cambridge, MIT Press, 2005.

[24] Herbert G. Tanner. Relaxed Stability Conditions for Switched Systems with Dwell Time. *Asian Journal of Control*, 16(2):313–320, 2013.

[25] D. Liberzon. *Switching in Systems and Control*. Birkhauser, Boston, MA, USA, 2003.

[26] J. Hespanha. Uniform stability of switched linear systems: Extensions of LaSalle's invariance principle. *IEEE Trans. on Automatic Control*, 49(4):470–482, 2004.

Appendix A

DERIVATION OF CLOSED-FORM EXPRESSIONS

This analysis is borrowed from [1]. The forward kinematics of the SFM when the right pair is activated are given below.

$$\Delta x = |O_1G'| \sin(\omega' - |\Delta\theta|) - |O_1G| \sin\omega \tag{A.1}$$

$$\Delta y = |O_1G'| \cos(\omega' - |\Delta\theta|) - |O_1G| \cos\omega \tag{A.2}$$

$$\Delta\theta = \varphi_2 - \varphi_1 + \angle AO_1A' \tag{A.3}$$

For clarity of presentation we have dropped the dependence on φ_1, and $|O_1G'| \triangleq r(\varphi_1)$, $|O_1G'| \triangleq r(\varphi_1)$, $\omega' = \omega(\varphi_1)$, and $\angle AO_1A' \triangleq \mathcal{X}(\varphi_1)$.

Figure A.1 shows the SFM model when the right pair is activated. The function of the displacement of the robot in the x direction, the displacement in the y direction, and the change in orientation are (A.1) (A.2) and (A.3)

A.1 Derivation of (A.1) and (A.2)

$$\Delta x = |O_1G'| \big[\cos(|\Delta\theta|) \sin(\omega') - \sin(|\Delta\theta|) cos(\omega') \big] - |O_1G| \sin(\omega)$$

$$\Delta y = |O_1G'| \big[\cos(|\Delta\theta|) \cos(\omega') + \sin(|\Delta\theta|) \sin(\omega') \big] - |O_1G| \cos(\omega)$$

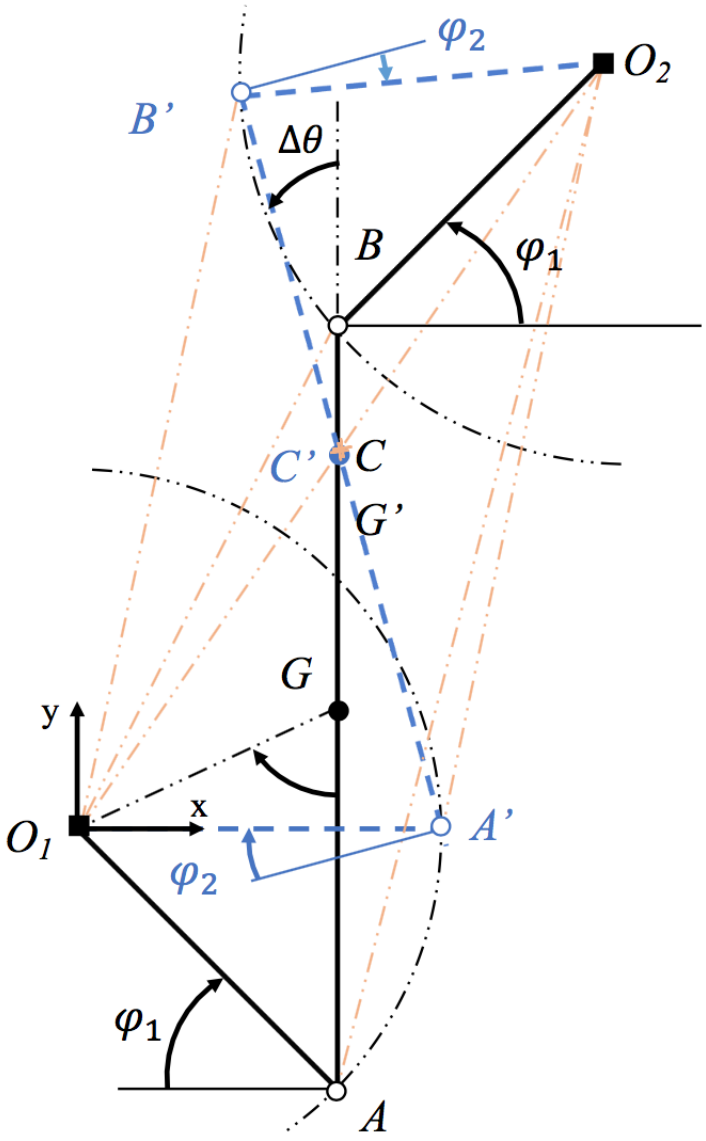


Figure A.1: Geometric analysis of SFM. Bold thick lines and unprimed symbols annotate the right pair configuration at touchdown, while dashed thick lines and primed letters describe the mechanism at the liftoff configuration. Thin dashed–dotted lines outline the various triangles we refer to in text. Due to the no–slip assumption, the mechanism pivots around the touchdown points, O_1 and O_2. Points C and C' denote the intersection of the torso of the model and the segment O_1O_2.

A.2 Derivation of (A.3)

$$\begin{aligned}\Delta\theta &= \angle A'C'O_1 - \angle ACO_1 \\ &= (\frac{\pi}{2} + \varphi_1 - \angle A'O_1C') - (\frac{\pi}{2} + \varphi_2 - \angle AO_1C') \\ &= \varphi_2 - \varphi_1 + (\angle AO_1O_2 - \angle A'O_1O_2) \\ &= \varphi_2 - \varphi_1 + \angle AO_1A'\end{aligned}$$

$$\begin{aligned}|O_1G'| &= \sqrt{l^2 + \frac{d^2}{4} - ld\cos(\frac{\pi}{2} - \varphi_1)} \\ \omega &= \arcsin\left(\frac{l\sin(\frac{\pi}{2} - \varphi_1)}{|O_1G|}\right) \\ |O_1G'| &= \sqrt{l^2 + \frac{d^2}{4} - ld\cos(\frac{\pi}{2} - \varphi_2)} \\ \omega' &= \arcsin\left(\frac{l\sin(\frac{\pi}{2} - \varphi_2)}{|O_1G'|}\right)\end{aligned}$$

$$\begin{aligned}\angle AO_1A' &= \arctan 2\left(\frac{\sin(\angle AO_1A')}{\cos(\angle AO_1A')}\right) \\ &= \arctan 2\left(\frac{\sin(\angle AO_1O_2 - \angle A'O_1O_2)}{\cos(\angle AO_1O_2 - \angle A'O_1O_2)}\right)\end{aligned}$$

or equivalently

$$\angle AO_1A' = \arctan 2\left(\frac{\sin(\angle AO_1O_2)\cos(\angle A'O_1O_2) - \cos(\angle AO_1O_2)\sin(\angle A'O_1O_2)}{\cos(\angle AO_1O_2)\cos(\angle A'O_1O_2) + \sin(\angle AO_1O_2)\sin(\angle A'O_1O_2)}\right)$$

$$|O_1O_2| = \sqrt{2l^2 + d^2 + 2l\Big(l\cos(\varphi_1 + \varphi_2) + d(\sin(\varphi_1) - \sin(\varphi_1)\Big)}$$

$$\begin{aligned}\angle O_1AO_2 &= \frac{\pi}{2} - \varphi_1 + \angle BAO_2 \\ \angle O_1A'O_2 &= \frac{\pi}{2} - \varphi_2 + \angle B'A'O_2\end{aligned}$$

$$|AO_2| = \sqrt{l^2 + d^2 - 2ld\cos(\frac{\pi}{2} + \varphi_1)}$$

$$\angle BAO_2 = \arcsin\left(\frac{l\sin(\frac{\pi}{2} + \varphi_1)}{|AO_2|}\right)$$

$$|A'O_2| = \sqrt{l^2 + d^2 - 2ld\cos(\frac{\pi}{2} + \varphi_2)}$$

$$\angle B'A'O_2 = \arcsin\left(\frac{l\sin(\frac{\pi}{2} + \varphi_2)}{|AO_2|}\right)$$

Appendix B

INVERSE KINEMATICS OF SFM

The expression of the $\Delta\theta$ of the forward kinematics when the right pair is activated is given below:

$$\begin{aligned}\Delta\theta &= \varphi_2 - \varphi_1 \\ &+ \arctan\left(\begin{array}{r}\sqrt{d^2 + l^2 + 2\,ld\sin(\varphi_1)} \\ ((d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l)\sin(\pi/2 - \varphi_1 + h) \\ -\sin(\pi/2 - \varphi_2 + j)\,(d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)) \\ \hline \sin(\pi/2 - \varphi_2 + j)\left(d^2 + l^2 + 2\,ld\sin(\varphi_1)\right) \\ \sin(\pi/2 - \varphi_1 + h) - (d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l) \\ (d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)\end{array}\right)\end{aligned} \tag{B.1}$$

Where

$$h = \arcsin\left(\frac{l\cos(\varphi_1)}{\sqrt{d^2 + l^2 + 2\,ld\sin(\varphi_1)}}\right)$$

$$j = \arcsin\left(\frac{l\cos(\varphi_1)}{\sqrt{d^2 + l^2 + 2\,ld\sin(\varphi_2)}}\right)$$

We assumed the robot body length is much larger than the leg length ($d \gg l$), the h and j in (B.1) is approach to zero. The (B.1) becomes:

$$\begin{aligned}\Delta\theta &= \varphi_2 - \varphi_1 \\ &+ \arctan\left(\begin{array}{r} \sqrt{d^2 + l^2 + 2\,ld\sin(\varphi_1)} \\ ((d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l)\sin(\pi/2 - \varphi_1) \\ -\sin(\pi/2 - \varphi_2)\,(d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)) \\ \hline \sin(\pi/2 - \varphi_2)\left(d^2 + l^2 + 2\,ld\sin(\varphi_1)\right) \\ \sin(\pi/2 - \varphi_1) - (d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l) \\ (d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)\end{array}\right)\end{aligned} \tag{B.2}$$

By using trigonometric identities and formulas, (B.2) can be simplified:

$$\begin{aligned}\Delta\theta &= \varphi_2 - \varphi_1 \\ &+ \arctan\left(\begin{array}{r} \sqrt{d^2 + l^2 + 2\,ld\sin(\varphi_1)}\,((d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l) \\ \cos(\varphi_1) - \cos(\varphi_2)\,(d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)) \\ \hline \cos(\varphi_2)\left(d^2 + l^2 + 2\,ld\sin(\varphi_1)\right)\cos(\varphi_1) \\ -(d\sin(\varphi_2) - l\cos(2\,\varphi_1) - l) \\ (d\sin(\varphi_1) - l\cos(2\,\varphi_1) - l)\end{array}\right)\end{aligned} \tag{B.3}$$

In order to simplify (B.3), we denote m, n, and p which does not include φ_1:

$$m = d^2 + l^2 + 2ld\sin(\varphi_1)$$

$$n = \sin(\varphi_1)$$

$$p = \cos(2\varphi_1)$$

Then, equation (B.3) becomes:

$$\tan\left(\Delta\theta - \varphi_2 + \varphi_1\right) = \frac{\begin{array}{r}\sqrt{m}\left(1/2\ \left(d\sin\left(\varphi_2\right) - lp - l\right)\sqrt{2\,p+2}\right. \\ \left. -p\left(dn - lp - l\right)\right)\end{array}}{\begin{array}{r}1/2\ \cos\left(\varphi_2\right) m\sqrt{2\,p+2} \\ -\left(d\sin\left(\varphi_2\right) - lp - l\right)\left(dn - lp - l\right)\end{array}} \tag{B.4}$$

We denote a which does not include φ_1, and b, which is the only parameter including φ_2:

$$a = \tan(\Delta\theta + \varphi_1)$$

$$b = \sin(\varphi_2)$$

Then, equation (B.4) becomes:

$$\frac{\left(a - \frac{b}{\sqrt{-b^2+1}}\right)}{\left(1 + \frac{ab}{\sqrt{-b^2+1}}\right)} = \frac{\sqrt{m}\left(1/2\ \left(db - lp - l\right)\sqrt{2\,p+2} - p\left(dn - lp - l\right)\right)}{1/2\sqrt{-b^2+1}m\sqrt{2\,p+2} - \left(db - lp - l\right)\left(dn - lp - l\right)} \tag{B.5}$$

All the parameters m, n, p, and a, do not include φ_2, where d is the robot body length, and l is the robot leg length, the only parameter which includes φ_2 is b. Since b is a very simple function of φ_2 ($b = \sin(\varphi_2)$), the numerical solution of φ_2 can be solved from (B.5).