NTNU

Norwegian University of
Science and Technology

Path-searching for Rolling Motion of the Two-Link Acrobot With Curved Links

Øystein Henriksen

Master of Science in Engineering Cybernetics

Submission date: Januar 2012

Supervisor: Anton Shiriaev, ITK

Co-supervisor: Uwe Mettin, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Path-searching for Rolling Motion of the Two-Link Acrobot With Curved Links

Øystein Sakspapir Henriksen

January 21, 2012

Dept. of Engineering Cybernetics, NTNU

Abstract

The Rolling Acrobot with Curved Links, is an underactuated, two-link planar robot. This study aims to discover a cyclic rolling motion of this robot, by actuation of the joint between the two links, which are suitably shaped in a curved fashion. This is not a trivial task, due to the underactuation of the Acrobot, and the fact that seven mathematical models will be needed because the contact point with ground changes during a cycle of the rolling motion. The motion in question consists of eight phases, which means that one of the mathematical models is used twice.

The search process for finding this rolling motion will be carried out as follows. First, suitable models will be derived using the Euler-Lagrange method. Second, the reduced system dynamics will be derived for each model. The purpose of the reduced system dynamics, is to reduce the system to one degree of freedom, assuming that a certain path is enforced by some control law. When this is done, the search for a closed trajectory of the rolling motion can begin. This is done by creating Bezier curves for each phase, which serves as virtual holonomic constraints on the system, and these curves should form a connected path in configuration space. Update laws must be provided for jumps between phases, because the meaning of the free variable will change when transitioning between certain models. Finally, an optimization routine searches for the locally optimal Bezier curves with respect to boundary conditions in configuration space and the required closed-property of the path. Moreover, the curve most be closed in state space, which means that interpolation conditions on the velocities are to be accounted for. The cost function for the search will demand that the values of the state variables at the end of the motion are equal to their initial values. If one such a trajectory is found, the cost function will be adapted to search for energy efficient trajectories, by including the resulting actuation force.

Contents

Chapter 1

Introduction

1.1 Motivation

The concept of a robot is well known for most people. It has influenced pop culture as well as mythology and philosophers through out history. The popular science-fiction franchises Star Wars and Star Trek both have humaniod robots. In 1950 the blueprints of a robot drawn by Leonardo DaVinci in the renaissance period were discovered. Ancient greek engineers created animal-like mechanical robots driven by air pressure. Jewish and Norse legends tells of clay golems and clay giants, and according to greek mythology a giant man made of bronze protected a cretian island from pirates. Humans are clearly fascinated by automated mechanical creations, even though the term "robot" was first introduced in 1920.

The field of robot locomotion considers the problem of automated physical movement of a mechanical system. Although our imagination is the only limit for accomplishing this problem, there are two approaches which dominates. These are legged locomotion and wheeled locomotion. Legged locomotion is what most people think of when the term robot is mentioned. It consists of a mechanical system with two legs which in some way emulates the walking motion of an animal or a human. Legged locomotion is not an easy task to undertake for any system. Several approaches and lots of theory exists, but due to the complex nature of this movement, there a many drawbacks. The energy cost of creating a humanoid walking robot is relatively high compared to a wheeled robot. This is because a lot of energy is neeeded to balance the robot as well as the loss of energy due to impacts between the leg and ground. There are also several stability issues. Imagine the sequenze of leg movement of a human being losing its balance. The complexity of this task takes years for a human to master, and still we fall down from time to time. One does not need to know

much about robotics to understand the difficulty of designing such a system.

Wheeled locomotion is a more stable approach of locomotion. Automated cars and wheeled robots are more energy efficient than walking robots. A car is basically stable all the time (when in the upright position), and the only energy needed for locomotion is the energy used to rotate the wheels. The drawbacks of wheeled locomotion is that it requires a relatively smooth terrain for moving. It would be highly unpractical to design a wheeled robot to climb a stairway or jump across a chasm. These are tasks that are more suitable for a legged lomotion robot.

Because of the pros and cons of both paradigms, it is favourable to combine the advantages of both wheeled- and legged locomotion. The idea behind such a hybrid system, is to harvest the best features of both the legged robot and the wheeled robot. Imagine a two-legged robot with wheels, like a human on rollerskates. The robot could then drive around energy efficiently on any smooth surface, and then step over any obstacle. This would result in a robot which utilizes the energy efficiency of the wheeled robot, as well as the versatility of the legged robot, which obviously would be advantageous.

The rollerskate-robot is just an example of a hybrid locomotion robot. This study will consider the Acrobot with Curved Links which has a different approach to achieving locomotion, but still utilizes the advantages of both wheeled and legged robots.

1.2 How this report is organized

The report is organized into the following chapters.

Chapter 1 is the introduction, which provides some motivation for this study.

Chapter 2 is the Problem Formulation. This chapter will give a brief introduction to the field of robot locomotion. It will describe the approaches of legged locomotion, wheeled locomotion and hybrid locomotion. There will be stated some equations of a technical nature, but by disregarding this, the reader should still be able to understand the text without prior knowledge of the subject.

Chapter 3 is the Literature Review. This chapter presents literature that relates to the Acrobot with Curved Links. There is currently little knowledge of this specific robot, and therefore the main part of this chapter will be concered with mathematical techniques and methods which will be used through out this study. There will be brief summaries of the papers or books in question, followed by a subjective evaluation. This evaluation will clearly state which aspect of these texts that are advantagous or disadvantagous to this study.

Chapter 4 is the Task Formulation. This chapter will present the task at hand in context of the Literature Review. This presentation will be detailed, and will incorporate scientific hypothesis and assumptions.

Chapter 5 is the Solution of Task. This is the main chapter of this report and will describe technically how the task at hand was solved. This chapter will also include theoretical sections which eplains in detail the methods used in the task solution. The nature of the task demands several simulation studies, and these simulation studies will be well documented and reviewed. This chapter demands that the reader has a theoretical background in the fields of control theory and mathematical modeling.

Chapter 6 is the Main Result and Conclusion. This chapter gives the main results of Chapter 5, and provides a conclusion as to whether the results are useful.

Finally, Chapter 7 gives a summary of the study.

The MATLAB files, Maple files and other files which have been used for calculations and design of this report, can be aquired electronically by sending an email request to *delern_ @hotmail.com*.

Chapter 2

Problem Formulation

2.1 Definitions

This chapter explains in detail the different paradigms of locomotion robots. Some abbreviations and concepts may be unfamiliar to the reader, and are stated in this section for easy referrence. These concepts will not be explained in a mathematical manner, but instead gives a brief explaination which should be sufficient to understand the purpose of this section.

- ZMP (Zero Moment Point) is a design architecture for robot locomotion. It is based on a strict stability criterion that the walking robot is stable at all times. When such a robot moves forward, the center of mass of the robot always moves in a straight line parallell to the ground.
- COT (Cost of transport) is a measure of energy consumption. It is defined as the energy consumed to move a unit weight a unit distance.

2.2 Legged Locomotion

Legged locomotion is the art of phsyical movements on legs. This may be on two legs, four legs or any other amount of legs. There exists several robots which utilizes this approach of locomotion, and some of them are illustrated in (Figur av roboter). There are several ways of designing such robots, and the different approaches have advantages as well as disadvantages. The approaches discussed in this study, will all be based on the differential equation stated in Equation 2.1. This equation describes

the dynamics of the robot, and is called the Equation of Motion (EOM).

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{2.1}$$

The legged locomotion robots considered in this study, are generally underactuated. This means that there are more degrees of freedom (DOF) than there are actuated joints. This makes control design hard, and to cope with this problem many different control strategies or architectures have been developed. These strategies have advantages and disadvantages, which will be discussed in this section.

When desinging legged locomotion robots, the two main aspects to consider are the versatility of the robot and its energy efficiency. High versatility of a robot often comes at the expense of high energy costs. One technique that is widely used for legged robots, are the Zero Moment Point technique (ZMP). The idea behind the ZMP approach, is that the robot should be stable at any point of the motion. As a ZMP robot moves forward, its center of mass moves along a straight line which is parallell to the ground. A lot of energy is consumed for this criterion to be satisfied, which gives the robot great versatility at the expense of a high energy cost. Honda's ASIMO robot is one of the most famous bipedal walking robots and it is designed using ZMP. ASIMO can walk, run, kick a ball and even play the fiddle. Although this is impressive, the energy cost of ASIMO's walking gait is estimated to be about ten times more expensive than that of a human, in terms of energy. The COT of ASIMO is estimated to be 3.2, and the COT of a human is 0.3.

Dynamic walking is another bipedal design architecture which is based on the passive dynamics of the legs, and produces cyclic gaits for walking instead of the position control employed in ZMP. The dynamical walker are more energy efficient than the ZMP robots, because it exploits the passive dynamics of the system. This is achieved by allowing the stance leg to behave as an inverted pendulum. The only substantial energy loss in dynamic walking, is due to the legs impact with ground. The walking gait of a dynamic walker alternates between to phases. These are the single support phase and the double support phase. In the single support phase, only one leg (the stance leg) is in contact with ground, and in the double support phase both feet are in contact with ground. One complete cycle of a walking gait will result in four phases. This is because the robots needs to make to steps to return to the initial position. An important difference between dynamic walkers and ZMP robots, is that while the ZMP robots should be stable at all instances of the walking gait, the dynamic walkers only demands orbital stability. Orbital stability means that the although the robot may be unstable at some points in the walking cycle, the cycle in itself should be stable. This is advantageous in terms of energy. The ZMP robot would use energy at every time instant to steer the gait to its preferred path. This

is in contrast to the dynamic walker, which may deviate from its preferred path as long as the sequence of paths is stable. With other words, the ZMP robot must be locally stable at all times, and the dynamic walker may be unstable at times as long as the motion in itself is stable. It is intuitively clear that this stability criteria is less conservative than the ZMP stability criteria, and therfore also more energy efficient.

Although the dynamic walker is superior to the ZMP robot in terms of energy, ZMP is still far superior in terms of versatility. The ZMP robot may stop at any point in its walking cycle because it is guaranteed that it always remains stable. The dynamic walker however would certainly fall if it were to stop, because it is only guaranteed to be stable when moving. It is also susceptible to noise. It proves to be difficult for dynamic walker to handle slopes and inclined floors.

A humanoid robot should be able to handle multiple situations. At present, the dynamic walker is only capable of achieving stability when it is walking or running, which is not sufficient in situations were a robot is needed. At a higher energy cost, the ZMP robots are still the only robots which may conduct useful tasks.

2.3 Wheeled Locomotion

The advantages of wheeled locomotion are many. To date, it is the most energy efficient mode of transportation. The only energy loss is due to friction forces and air resistance, and the stability issues are trivial compared to leg locomotion. The versatility of a wheeled robot is also good. Steep slopes and rough terrain can easily be overcome by cars or military vehicles. A wheeled robot is easy to construct compared to legged robots. Basically, all it needs are a motor and a set of wheels. Due to these factors, the wheeled robot is preferrable in almost all practical situations.

But there are clearly landscapes that can not be tranversed with a purely wheeled locomotion robot. For example jumping over a chasm or climbing a set of stairs[1]. In these cases a legged robot is a more practical solution.

2.4 Combined Locomotion

The advantages and disadvantages of the different locomotion strategies are listed in table (tabell med fordeler ulemper wheeled legged) .

[1] Of course any wheeled vehicle can climb a set of stairs if its wheels are big enough, and its motor is powerful. But you would not want that in your living room.

Strategy	Advantages	Disadvantages
ZMP	High versatility. Always stable.	Low energy efficiency.
Dynamic walking	High energy efficiency. Exploits the passive dynamics of the system.	Low versatility. Only stable while performing a gait. Difficult to design.
Wheeled locomotion	High versatility. High energy efficiency. Easy to design. Stability is trivial.	Impossible to use in certain situations.

Table 2.1: Pros and cons of the different locomotion strategies.

The ideal locomotion robot would combine the advantages of the different strategies while at same time avoiding the disadvantages. The Acrobot with Curved Links is an example of a robot which utilizes the legged motion as well as the rolling motion of the wheel. It is comprised of two links which are joined together at the end. This point forms a joint which in turn is actuated by a motor. The endpieces and sidepieces of are formed as circle sections, and the two links are identical in shape. Being constructed in this manner, the Acrobot is able to achieve a rolling motion on its side and end, and the legged locomotion may be achieved by actuating the joint motor.

The Acrobot also utilizes the passive dynamics of the dynamic walker. Suppose that the motor has locked the Acrobot in a certain position. Then, the gravitational force will act on the system and make it roll. Because of the geometric shape of the Acrobot, it is intuitively plausible that it can start a motion from the resting position. The resting position means that the Acrobot is folded and lying on its side. If this is possible, the versatility would greatly increase. The dynamic walker would not be able to "get up" easily if it was lying on the ground.

Designing a locomotion gait for the Acrobot will be harder than the trivial gait design of the wheeled robot. It is also assumed that the energy cost will be slightly higher, since the energy cost of the wheeled robot is only due to friction and air resistance. Still, by utilizing the passive dynamics and the rolling motion, the Acrobot may be able to achieve acceptably low energy costs. In addition, if the utlilization of the legged motion is possible, the Acrobot may be able to overcome obstacles that the wheeled robot may not. The Acrobot should thus incorporate the following properties:

- Relatively low energy cost.
- High versatility.
- Have the ability to start from a resting position.
- Overcome obstacles that the wheeled robot may not.

These points constitute the ideal realization of the Acrobot with Curved Links. Realizing such a robot completely is a very large and time-consuming task and therefore this study will focus mainly on designing a stable locomotion gait which is a hybrid of legged locomotion and wheeled locomotion. Realization of such a gait may gain favourable insight in the dynamics of the Acrobot, and also motivate other parties to investigate it further.

Chapter 3

Literature Review and Terminology

This chapter will present literature that is relevant to completing the task at hand. At present time, there exists little written work directly related to the Acrobot with Curved Links. There is one paper which illustrates how a rolling motion with impact is realized. This particular paper however, utilizes a control scheme based on the analysis of energy and feedback control. This is a fundamentally different approach than what is carried out in this study, and thus its relevance is due to the modelling of the Acrobot. The main part of this chapter will consider literature regarding the mathematical techniques of designing virtual constraints and the reduced system dynamics. Some terminology will also be explained.

3.1 Terminology

Configuration Space The configuration of a robot, is a complete specification of the location of its every point. The set of all such configurations is called the configuration space.

Bezier curves The Bezier curve is basically a polynomial function, and have been chosen for the virtual constraint functions structure in this study. The curves are designed by defining certain control points. These control points form a convex hull, in which the curve is entirely contained. In Figure 3.1, this concept is illustrated. The control points P_0 to P_3 forms an area in which the curve is completely contained. Another important feature of the Bezier curve, is that the start- and endpoint of the curve is specified by P_0 and P_3 respectively.

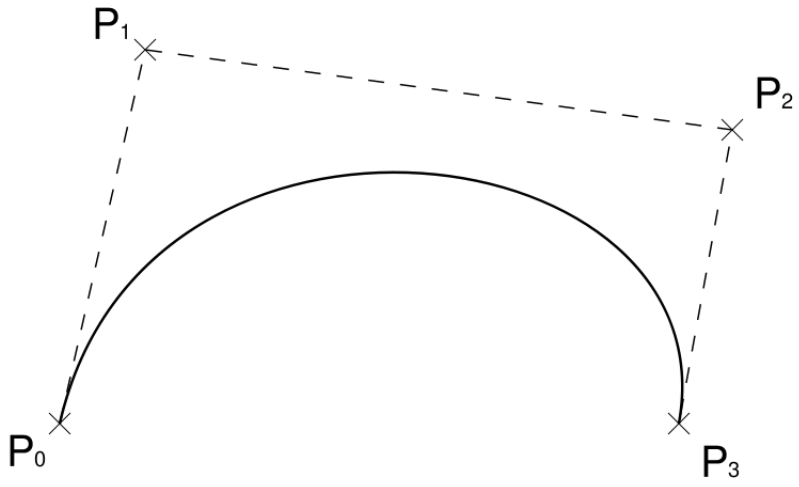

Figure 3.1: Third order Bezier curve.

The formal definition of a Bezier curve is stated below.

$$B(t) = \sum_{i=0}^{n} \begin{pmatrix} n \\ i \end{pmatrix} (1-t)^{n-i} t^i P_i \ , \ t \in [0,1]$$

where P_i are the control points and n is the order of the polynomial.

Phase In this study, the term phase will be used to refer to a specific part of the cycle of a specific motion. Figure 3.2 illustrates the eight phases of the rolling motion without impact.

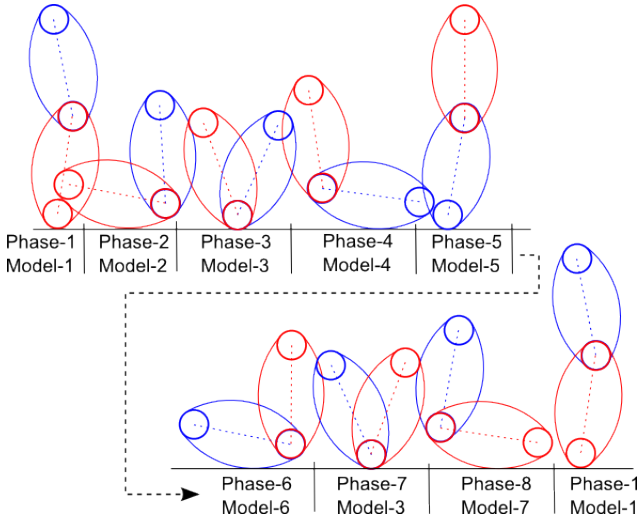

Figure 3.2: The phases and corresponding models of the rolling motion without impact that is proposed in this study.

3.2 Continous Rolling Motion Control of the Acrobot

The paper [11] aims to control the Acrobot with Curved Links by analysing the energy of the system during a rolling motion with impact. Sampei divides the motion into three phases; the upward phase, the downward phase and the impact phase. By intuitively reasoning what the objective of each phase should be, feedback controllers are designed using output zeroing and LQR.

To achive the desired rolling motion, Sampei proposes five distinct continous models and one discreet model which describes the impact with ground. The five continous models are needed because the contact point with ground changes during the motion. The continous models have four degrees of freedom (4DOF); the absolute angle of Link1 with respect to ground, the relative angle of Link2 with respect to Link1 and the x and y-coordinates of the Acrobot. The system are reduced to 2DOF by imposing virtual constraints on the system such that the Acrobot is always in contact with ground, and that there are strong enough friction between ground and the Acrobot to avoid slipping. These models are derived using the Euler-Lagrange method. The rolling motion with impact was realized by numerical simulation.

Relevance to this study The control scheme of [11] is fundamentally different from the approach in this study. The relevance to this paper is due to the modelling process. The rolling motion with impact which is realized in [11], demands five distinct continous mathematical models and one discreet model of the impact. This study will focus on the rolling motion without impact which is realized by using seven distinct models, where five of these are the five continous models proposed by [11]. The physical properties of the Acrobot with Curved Links are the same as those in [11]. This is because the Acrobot in[11] is based on a physical prototype, and if this study should produce interesting results, these may be tested on the prototype.

The result of the numerical simulation study of [11] is interesting because the realized motion is a combination of the impact with ground and passive rolling motion. This makes the motion a hybrid of legged locomotion and wheeled locomotion. The drawback is that energy is lost in the impact phase. By realizing such a motion without impact, this potential energy loss may be avoided. Of course, the rolling motion without impact may possibly consume more energy than the rolling motion with impact, but without the impact one source of guaranteed energy dissipation is removed.

3.3 Fundamentals of Robot Modeling and Control

The book [8] explains the fundamentals of the modeling and control of robots, and specifically of robot arms. In this study, only the modeling part will be of interest.

The problem of forward kinematics, is to determine the position and orientation of the end effector given the values for the joint variables of the robot. To achieve this, one needs to attach coordinate frames to each link and define homogeneous transformation matrices which relates the coordinate frames. In [8] this is done by using a standard convention called the Denavit-Hartenberg Convention.

When the kinematics of the system have been established, one needs to derive the kinetic and potential energy of the system. These expression will be used to form the Lagrangian, which is defined as:

$$L = K - P \tag{3.1}$$

where L is the Lagrangian of the system, and K and P is the kinetic- and potential energy, respectively. Finally, the Lagrangian is used in the Euler-Lagrange equation to derive the equations of motion.

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} = \tau_k \tag{3.2}$$

where L is the Lagrangian of the system, q is the state vector and τ_k is the generalized force associated with q_k.

The homogenous transformation matrix, is a matrix representation of translational- and rotational motion for a rigid object. The definitions given here, is according to [8].

$$H_m^k = \begin{bmatrix} R_m^k & d^k \\ 0 & 1 \end{bmatrix}, \quad R \in SO(3), \quad d \in \mathbb{R}^3 \tag{3.3}$$

H_m^k relates coordinate frame m to coordinate frame k. A point p given in frame m, is denoted p^m.The same point p given in frame k, is denoted p^k. The homogenous transformation matrix relates $p^m$and p^k as follows:

$$\begin{bmatrix} p^k \\ 1 \end{bmatrix} = H_m^k \begin{bmatrix} p^m \\ 1 \end{bmatrix} \tag{3.4}$$

Relevance to this study This book is mainly concerned with the modeling and control of robot arms. Still, the basic theory of forward kinematics may be utilized for the Acrobot. The book strongly emphasizes the use of the Denavit-Hartenberg convention for deriving the forward kinematics. This convention is useful if the system to be modeled has the structure of a three dimensional robot arm. The Acrobot model is planar (two dimensional) and is comprised of only two links. This makes the modeling of the forward kinematics easy, and no such modeling convention is neccessary.

The derivation of the kinetic and potential energy is based on the forward kinematics. The forward kinematics gives the cartesian coordinates of the center of mass of both links. Then, due to the geometric shape of the Acrobot, it is trivial to derive the kinetic and potential energy functions. Finally, the Lagrangian can be formed and the equations of motion may be derived using the Euler-Lagrange equation of Equation 3.2.

Although the Acrobot consists of only two links, the kinematic equations are still complex. This is because the Acrobot is not rigidly suspended to the ground. Therefore, the calculations of the forward kinematics and the equations of motion was done using the symbolic computational tool, Maple. Using a such a tool is time-saving and avoids human errors.

3.4 Bipedal Legged Locomotion

The book [3] attempts to collect all individual publications about design and control of bipedal legged locomotion in one self-contained text. Bipedal legged locomotion is a subclass of legged locomotion, which is restricted to walking robots with two legs.

The book explains in detail the steps of designing and controling a bipedal robot. It also contains extensive theoretical sections and appendices which formally states mathematical proofs and concepts. The Acrobot with Curved Links is not a bipedal walking robot in the sense according to [3], which means that an extensive review of this book is not necessary for the completion of the task in this study. Instead, the mathematical concepts which are relevant will be thoroughly explored in the following.

Relevance to this study The relevance of [3] is quite extensive, and therefore this paragraph will be divided into subsections, for easy referrence and readability.

3.4.1 Autonomous Systems with Impulse Effects

According to [3] the autonomous system with impulse effects consists of three things.

1. An autonomous ordinary differential equation of the form $\dot{x}(t) = f(x(t))$.
2. A hyper surface S at which solutions of the differential equation undergo a discrete transition that is modeled as an instantaneous reinitialization of the differential equation.
3. A rule $\triangle : S \to X$ that specifies the new initial condition as a function of the point at which the solution impacts S.

Such a system is denoted by

$$\Sigma : \begin{cases} \dot{x}(t) = f(x(t)) & x^-(t) \notin S \\ x^+(t) = \triangle(x^-(t)) & x^-(t) \in S \end{cases} \tag{3.5}$$

where S is called the impact surface or switching surface and $\triangle$ is the impact map. It can be seen from Equation 3.5, that the system behaves according to differential equation while the state vector is not on the switching surface. Otherwise, when the state vector lies on the switching surface, the state vector is instantaneously mapped to the new point $x^+(t)$ by the impact map. This mapping will be referred to as a switch.

One of the key aspects of the rolling motion without impact, is the fact that it contains no impacts with ground. The "impacts" in the context of the Acrobot is therefore simply a change of coordinates and equation of motion. In light of this fact, such a strict definition as Equation 3.5 may seem redundant, but this is not the case. The reason for this will be explained thoroughly in Chapter 5, but for now it suffice to say that the meaning of the state variables vary from model to model.

The notation x^+ and x^- are used to denote the values of the state vector just after- and just before the switch, respectively.

3.4.2 Virtual Constraints and The Reduced System Dynamics

The notion of virtual constraints are essential to the completion of the task in this study. Before stating the definition of the virtual constraint, the concept of an underactuated system will explained. This will establish the usefulness and power of imposing such constraints on the system.

An underactuated system, is a system with a lower number of actuators than degrees of freedom. The control design process for such a system, is considered non-trivial. This will be illustrated by the following example. The standard robot equation (Equation 2.1) is restated here for convenience.

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{3.6}$$

Assume that $M(q)$, $C(q,\dot{q}) \in \mathbb{R}^{2x2}$ and q, τ, $G(q) \in \mathbb{R}^{2x1}$. Assume further that $\tau = \begin{bmatrix} u & 0 \end{bmatrix}^T$. Now, Equation 3.6 may be written in the following form

$$\begin{aligned} f_1(q,\dot{q},\ddot{q}) &= u \\ f_2(q,\dot{q},\ddot{q}) &= 0 \end{aligned} \tag{3.7}$$

Clearly, the function f_1 may be controlled by the external input u, but the other function, f_2, may not be controlled directly, and thus the system is underactuated. However, underactuated system are not impossible to control, but they require more sophisticated approaches for control than fully actuated systems. One such approach is to impose a set of virtual constraints on the system.

Imagine that a controller was somehow able to make the following equation invariant.

$$q_1 = \varphi(q_2) \tag{3.8}$$

Equation 3.8 is an example of a virtual constraint. The constraint is *virtual* because it does not arise from any kind of phsyical connection between $q_1$and q_2.

It is merely imposed on the system by some arbitrary control law. By deriving the first- and second order time derivative of Equation 3.8, we get

$$\begin{aligned} \dot{q}_1 &= \varphi'(q_2)\dot{q}_2 \\ \ddot{q}_1 &= \varphi''(q_2)\dot{q}_2^2 + \varphi'(q_2)\ddot{q}_2 \end{aligned} \tag{3.9}$$

Now, q_1 and its first- and second order time derivative may be expressed solely as functions of φ, q_2, $\dot{q}_2$ and $\ddot{q}_2$. By inserting this result into Equation 3.7, we get

$$\begin{aligned} f_3(q_2, \dot{q}_2, \ddot{q}_2, \varphi) &= u \\ f_4(q_2, \dot{q}_2, \ddot{q}_2, \varphi) &= 0 \end{aligned} \tag{3.10}$$

We now see that the robot equation is only dependent on q_2 and its first- and second order time derivative, and q_1 has vanished. If u is chosen to be $u = f_3$, Equation 3.10 is reduced to the following system

$$f_4(q_2, \dot{q}_2, \ddot{q}_2, \varphi) = 0 \tag{3.11}$$

Equation 3.11 can be written on the following form

$$\alpha(s)\ddot{s} + \beta(s)\dot{s}^2 + \gamma(s) = 0 \tag{3.12}$$

where, this specific case, $s = q_1$. Equation 3.12 is known as the *Reduced Dynamics Equation.*

The virtual constraint, φ, can now be arbitrarily chosen. Since $u = f_3$, the actuated equation of Equation 3.10 will also be trivial. The virtual constraint may be viewed as a path in the configuration space of the system. Then, if the desired path is known, one simply choses a constraint function φ which realizes this path. In other systems, the ideal path may not be known, due to the complexity of the motion. In fact, the desired motion might not even be possible.

When choosing virtual constraints there are several things to consider. Consider the function $y = \varphi(x)$.

1. The constraint function φ must be chosen such that each value of x corresponds to exactly one value of y.

2. Because of Equation 3.9, it is necessary that $\frac{d\varphi}{dx}$and $\frac{d^2\varphi}{dx^2}$ exists.

If the conditions above are satisfied, the structure of φ may be chosen. Popular choices are polynomial functions, trigonometric functions and Bezier functions. In

this study, the Bezier functions (or Bezier curves) has been chosen as the structure for the virtual constraints.

Finding the right choice of virtual constraint can be a trivial matter, but may also prove to be a challenge. The latter is the case for the Acrobot. The rolling motion without impact which is the desired motion of this study, is highly complex. It is not easy to intuitively "guess" how to design the virtual constraints. In other words, it is hard to guess how the path in configuration space should look like.

3.5 Fundamentals of Numerical Optimization

The book [5] covers theory and methods for solving optimization problems. It introduces the reader to notion of unconstrained and constrained optimization problems, and effective algorithms for solving these problems numerically. These problems are often formulated as a minimization problem. For example, the problem of minimizing the function $f(x) = x^2$ has a solution where the input variable $x = 0$. This is an unconstrained optimization problem. By adding the constraint $x \geq 1$, the problem becomes a constrained optimization problem, with solution $x = 1$.

Relevance to this study The optimization algorithm in this study is a built-in function in MATLAB. This function is called "fmincon", and is an algorithm for solving non-linear optimization problems based on the line search algorithm[1]. The line search strategy chooses a direction p_k and searches along this direction for a new and lower function value. How far the algorithm travels along p_k for each iteration, varies for each algorithm.

The optimization problem in this study, is defined as a constrained optimization problem. The mathematical formulation of such problems, will be lent from [5], and is stated below

$$\underset{x \in \mathbb{R}^n}{min\, f(x)} \quad subject\, to \begin{cases} c_i(x) = 0, & i \in E \\ c_i(x) \geq 0, & i \in I \end{cases} \tag{3.13}$$

where

- x is the vector of *variables*, also called *unknowns* or *parameters.*
- $f(x)$ is the *objective function.* This is the function to be minimized.

[1] In fact, fmincon may use several different algorithms, but line search is used in this study.

- $c_i(x)$ are *constraint functions*. These functions defines certain equalities or inequalities that the vector x must satisfy for the solution to be valid.
- I and E are sets of indices.

It should be mentioned that the objective function is often called the cost function in optimization literature. In this study, there will be a distinction between these terms. The objective function is the function that does the actual numerical simulation of the Acrobot. The cost function is contained inside the objective function, and is concerned with measuring how good the current solution of the objective function is. By making this distinction, it should be easy to change the cost function while the objective function remains unchanged.

3.6 The Graphical Simulator

The Graphical Simulator is a stand-alone program written in Microsoft Visual Studios C#. The purpose of this program, was to aid the modelling- and simulation process.

The Graphical Simulator takes values of θ_1 and θ_2 as input[2], and generates a graphical picture of the Acrobot for this specific configuration. This is done in the code by calculating the position of the geometric center of Link1 and Link2. The calculation is based on the results from the modelling process. The mathematical formulas obtained for the position vectors of these geometric centers proved to be complex. Therefore, by using the Graphical Simulator, the formulas could be thoroughly checked.

The Graphical Simulator also has an animation feature which can be used to visualize numerical simulations of the Acrobot. When a Matlab simulation of the system is completed, the resulting vectors for θ_1 and θ_2 are stored in text files which are read by the Graphical Simulator. The Graphical Simulator then shows an animation of the simulated motion. The animation may be stopped at any point, and the user can single-step through any part of the simulation. The step size can be specified manually. These features proved to be useful in debugging the Matlab simulation-script, and served as valuable feedback during the tuning of control parameters and choice of initial conditions.

Finally, it should be noted that the there exists several bugs in the code of the Graphical Simulator. As stated before in this section, the purpose of this program

[2]In this study, θ_1 and θ_2 are used to denote the configuration space of the Acrobot.

was to aid the modeling- and simulation process, and not to be a user-friendly program. Therefore, if the bugs were easily avoided, they were not fixed.

Figure 3.3 shows a screenshot of the Graphical Simulator.

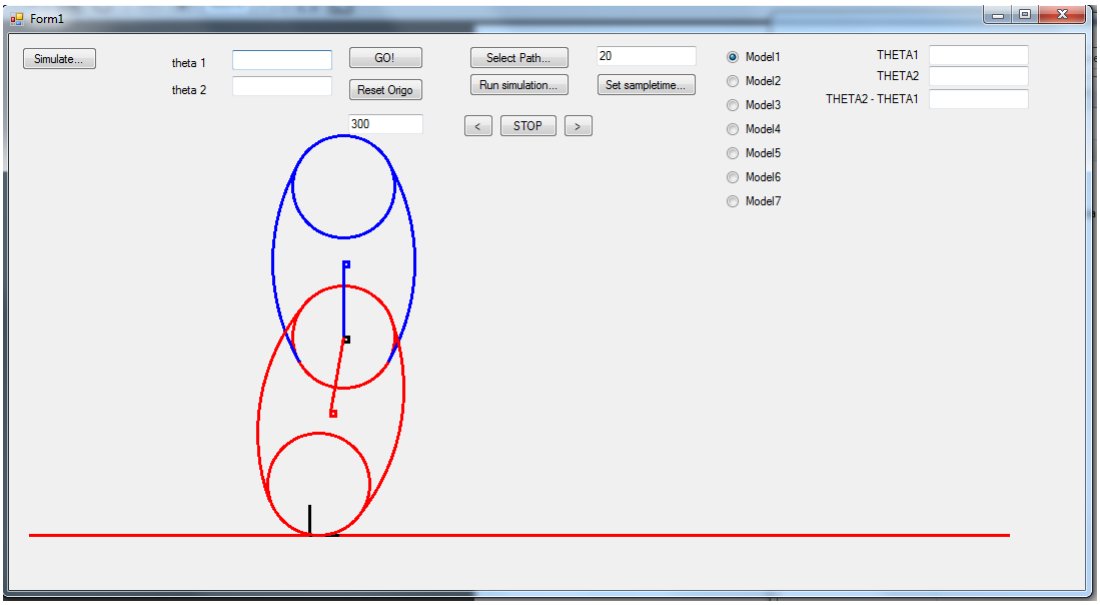

Figure 3.3: Screenshot of the Graphical Simulator.

Chapter 4

Task Formulation

This chapter offers a detailed description of the task at hand. The problem will be described in light of the definitions and concepts of the previous sections. Detailed mathematical derivations, however, is the topic of Chapter 5.

4.1 Modeling the System with Impulse Effects

The realization of the rolling motion without impact is the goal of this task. To achieve this goal, a hybrid system with impulse effects will be designed. This hybrid system will consist of seven different mathematical models. One complete cycle of this motion, will require eight phases and one model is assigned to each phase.

These models will be 4DOF, and then reduced to 2DOF by imposing two constraints. These constraints will be:

1. There is assumed that the friction at the contact point with ground is strong enough to prevent slipping.
2. The Acrobot should be in contact with ground at all times (no jumping).

The state vector of these 2DOF models will be denoted $\theta(t) = [\theta_1(t)\, \dot{\theta}_1(t)\, \theta_2(t)\, \dot{\theta}_2(t)]^T$, where

1. $\theta_1(t)$ denotes the absolute angle of Link-1 relative to ground.
2. $\theta_2(t)$ denotes the relative angle between Link-1 and Link-2.
3. $\dot{\theta}_1(t)$ and $\dot{\theta}_2(t)$ denotes the time derivative of $\theta_1(t)$ and $\theta_2(t)$, respectively.

After this reduction process, the equations of motion for the models will be derived using the Euler-Lagrange method. Furthermore, the reduced dynamic system is derived for each model, by imposing a generic virtual constraint function $\varphi(s)$. Depending on the models configuration space, s will be used as either $\theta_1(t)$ ord $\theta_2(t)$.

After the models have been derived, their respective configurations spaces will be deduced. Each individual configuration space will be stated with clear boundaries. These boundaries will define the valid configuration space of each model. The term *valid configuration space* will be used to describe the part of the configuration space which is allowed. For example, the models does not take into account configurations where the link which is not in contact with the ground, goes through the ground. According to the model-equations, this is a perfectly legal configuration. Therefore, to remove these obviously faulty configurations, the term valid configuration has been introduced.

When the valid configuration spaces have been established, the impact surfaces and impact maps will be designed. For the Acrobot, the impact surfaces will be one of the boundary lines of the valid configurations space. The impact maps will mostly be trivial, but some will not. However, there will be no loss of energy at any of these switches. The impact maps will only change the meaning of the coordinates.

4.2 Searching for Paths

When the system with impulse effects is ready, a search for paths that realizes the rolling motion without impact can begin. The search will be constructed in the following manner. First, there will be installed simple PD-controllers on some of the model's EOM. Furthermore, these controllers will be tuned manually according to feedback from the Graphical Simulator, to find a path that is close to the rolling motion without impact. When such a path is found, it is divided into eight pieces, according to each phase. Then, a Bezier curve will be fitted to each of the path-curves of the phases. These Bezier curves will in turn be used as virtual constraints in each phase, and serve as an initial guess for a search routine.

The search routine mentioned above, will be based on a non-linear optimization function that utilizes the line-search algorithm. The objective function will be based on numerical simulation of the Reduced Dynamics Systems, and the vector of parameters, x, will be composed of the following.

1. The control points of the eight different Bezier curves (virtual constraints).

2. The initial condition $\dot{s}_b$ of Phase-2.

3. A vector containing the values of s at the end of each phase.

The objective function will be bounded by upper- and lower bounds of the control points. These bounds will serve as constraint functions, $c_i(x)$. The objective function will ensure that any solution is kept within the bounds of the configuration space.

The search for paths of the rolling motion without impact is no trivial matter. There is no way of knowing what the results of such a search will be, and therefore there will be a lot of experimenting with the objective function, cost function and parameters of the search routine.

Chapter 5

Main Part

This is the main chapter of this study. The previous chapter gave a step-by-step presentation of the task in context of the literature and concepts which is related to the Acrobot with Curved Links. This chapter explains these steps in detail, and shows and discusses the different simulation studies which are carried out.

The first part of this chapter will introduce som terminology that has been created by the author for the purpose of this study. This terminology is specific to the Acrobot with Curved Links.

5.1 Specific Terminology related to the Task

The Acrobot or The Acrobot with Curved Links The terms *Acrobot* or *Acrobot with Curved Links* will be used interchangeably, and will refer to The Acrobot with Curved Links.

RMWOI (Rolling Motion WihtOut Impact) *RMWOI* is an abbreviation of *Rolling Motion WithOut Impact*. This refers to the specific motion with will be the subject of this chapter, and not some arbitrary rolling motion without impact. Also, if the term *rolling motion* is used, it is assumed to be the RMWOI unless stated otherwise.

Link-1 and Link-2 The terms *Link-1* and *Link-2* will be used to refer to the specific links of the Acrobot.

Model-N and Phase-M The models and phases have been given names. The models names are Model-1, Model-2 and up to Model-7. The names of the phases

are Phase-1, Phase-2 and up to Phase-8. The name of the phases are chronological in the sense of their names, meaning that Phase-(M-1) always preceedes Phase-M in the RMWOI. This is not the case for the models. The names of the models have no connection with the phase-names.

The Acrobot model or The complete Acrobot model Both these terms will refer to the complete Acrobot model, and this term refers to the collection of all the seven models.

Active model The term *active model* refers to the specific model of the complete acrobot model which is currently active. For example, during Phase-7 of the RMWOI, Model-3 is the active model.

Endpice, Sidepiece and Joint-endpiece The term *sidepiece* is used to describe any of the two sidepieces of each link. The term *endpiece* is used to describe the endpiece of each link which is not at the joint. The term *joint-endpiece* is used to specify the endpiece at the joint of the two links. The text will also discrimate between the left- and right sidepieces. The left- and right sidepieces of a link are defined as follows. Start at the endpiece, and move towards the joint-endpiece. Then the left sidepiece will be to your left, and the right sidepiece to your right.

CORM (Center Of Rolling Motion) Any motion of the RMWOI may be viewed as piecewise rolling motions. For example, when the Acrobot is lying on one of its sidepieces, the center of its rolling motion will be located at the center of the circle formed by the circle section of the sidepiece. The same reasoning holds for when the endpiece is in contact with ground.

COM (Center Of Mass) and GC (Geometric Center) The terms *center of mass* and *geometric center* should be well known. They are mentioned here, because in context of the Acrobot there is a vital difference between them. Since the links of the Acrobot are geometrically identical, their GC are identical. Their physical properties however, are not the same, and therefore neither are their COM.

5.2 Deriving the Model

5.2.1 Introduction

The Rolling Acrobot with Curved Links, is an underactuated robot composed of two links which are joined togheter at one of the endpoints of both links (see Figure 5.1).

The joint between the two links are actuated by a motor, which makes the relative angle between the links controllable. The Acrobot is designed to move in a two-dimensional plane, which leads to planar mathematical models. Any movement in the third dimension, which may occur in physical simulations, will therefore be regarded as faulty behaviour, and is not accounted for in the modelling process.

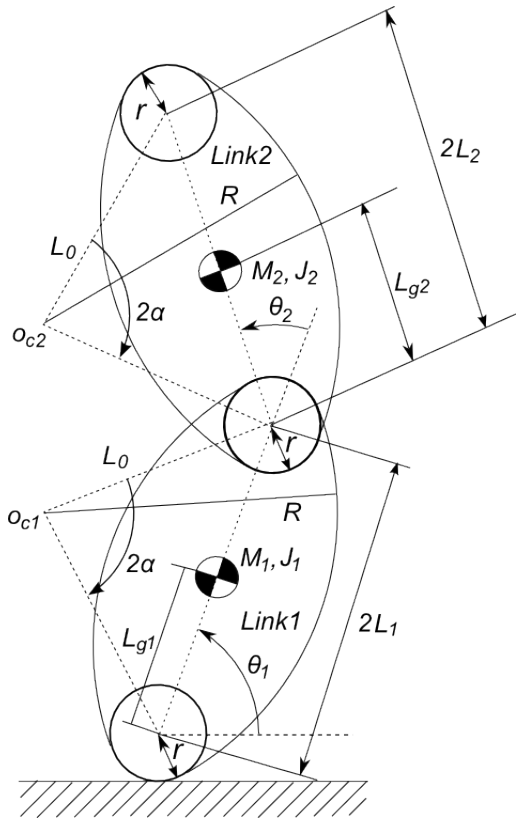

Figure 5.1: The Acrobot and its physical properties.

The rolling motion without impacts can be realized by using seven different mathematical models. During this motion, the Acrobots contact point with ground

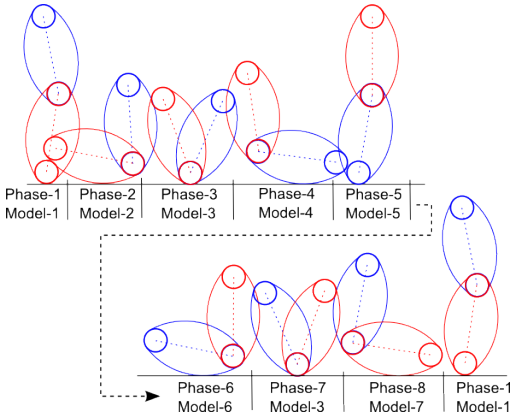

Figure 5.2: Sketch of the Rolling Motion WithOut Impact (RMWOI).

changes[1]. Figure 5.2 illustrates the essential aspects of the RMWOI. It is important to notice that the phase-names are chronological, but the model-names are not.

Figure 5.2 also shows the relationship between the each individual phase and the models. During Phase-1 the active model is Model-1, during Phase-5 the active model is Model-5 and so on.

Now, having made this sketch of the rolling motion, the criteria of the when to use the different models may be stated. These criteria are given in Table (ref table om criteria).

Modelname	Criterion for use
Model-1	The endpiece of Link-1 in contact with ground.
Model-2	The right sidepiece of Link-1 in contact with ground.
Model-3	The joint-endpiece in contact with ground.
Model-4	The left sidepiece of Link-2 in contact with ground.
Model-5	The endpiece of Link-2 in contact with ground.
Model-6	The right sidepiece of Link-2 in contact with ground.
Model-7	The left sidepiece of Link-1 in contact with ground.

Table 5.1: Criteria for which model to use.

The reader may have noticed the distinction between left- and right side of the sidepieces in Table 5.1. The definition of the left- and right sidepieces are found in

[1]The contact point actually changes continously, due to the curved side- and endpices of the links. When speaking of a change of contact point in this study, it is meant a change from a sidepiece being in contact with ground to an endpiece being in contact with ground, or vice versa.

Section 5.1. This distinction is an important one, and the reason for this will be thoroughly explained later in this section.

Consider Figure 5.3. This model shows the physical parameters of the Acrobot. The notion of degrees of freedom may be defined as the number of paramters needed to complete specify an objects orientation and position. By this definition, it should be clear that the Acrobot model is 4DOF; two angles, θ_1 and θ_2, denotes the orientation, and two cartesian coordinates are needed to describe its position in the two-dimensional plane.

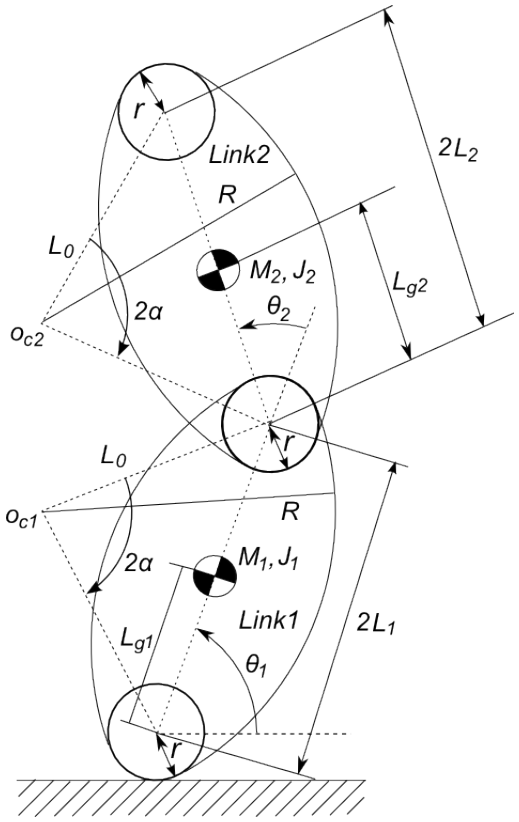

Figure 5.3: Physical parameters of the Acrobot.

Since only one parameter is actuated, namely θ_2, the degree of underactuation is $4-1=3DOF$. This may be reduced to only 1DOF, by introducing some constraints to the model. These constraints are stated in Section 4.1, but will be restated here for convenience.

1. There is assumed that the friction at the contact point with ground is strong

enough to prevent slipping.

2. The Acrobot should be in contact with ground at all times (no jumping).

If these constraints are ensured, then there is possible to derive the position of the Acrobot in the two-dimensional plane by its orientation variables, θ_1 and θ_2. Consider Figure 5.4. Assume that Link-1 of the Acrobot is positioned the manner illustrated by Figure 5.4a. The coordinates[2] x_c^0 and y_c^0 defines the CORM, and are located at the center of the circle formed by the contour of the sidepiece. Now, a coordinate frame, o_0, is placed on the surface of the ground, as shown in the figure. Due to the constraints just introduced, the only legal motion of the Acrobot will be to roll left or right, and as a consequence $y_c^0 = R$. It should also be intuitively clear, that by the simple geometric relationship, $x_c^0 = R\omega$, where ω is the rolling angle. This rolling angle may easily be deduced from θ_1 and θ_2. As for Figure 5.4 b, the same reasoning holds, and will not be discussed further.

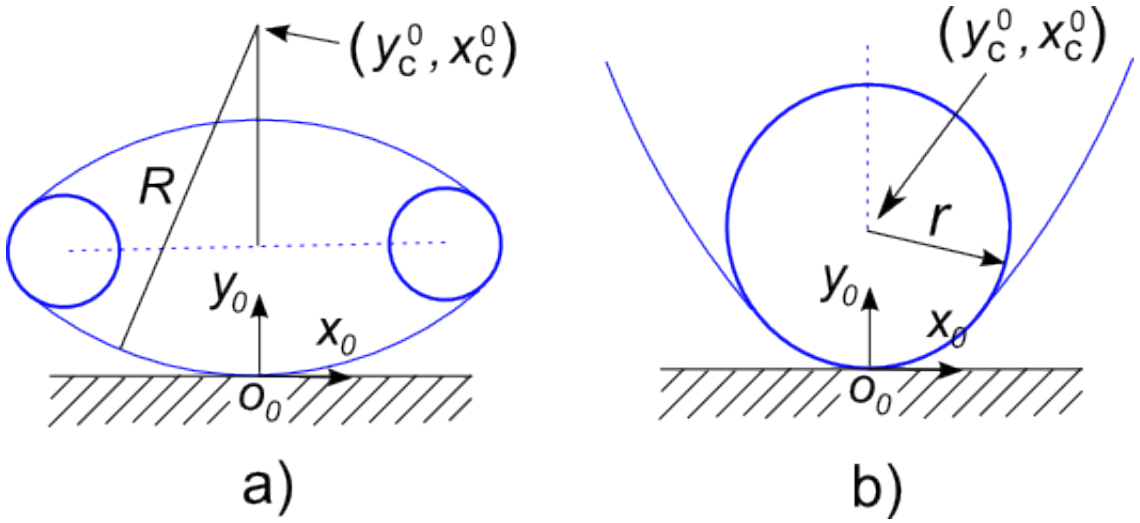

Figure 5.4: The CORM (Center Of Rolling Motion) if the Acrobot.

Therefore, by imposing these constraints on the system, the resulting Acrobot model will be 2DOF. It is important to keep in mind, that these constraints are purely assumptions, which most be ensured manually. The equations of motions of the different models are not "aware" of them.

[2]The coordinates x_c^0 and y_c^0 are rigidly placed at the center of rolling motion (CORM). The "0" denotes that their values are given relative to coordinate frame o_0.

Although the seven models differ, there are some design principles which hold for all of them. The angle θ_1 is the absolute angle of Link-1 relative to ground, and θ_2 is the relative angle between Link-1 and Link-2. It should be noted that there are several ways of measuring these angles. Therefore, a specific value of θ_1 may result in different orientations of Link-1 relative to ground for different models. Also, the geometrical centers of the the two links are described with the coordinate frames o_1 and o_2 for Link-1 and Link-2, respectively. These points are not to be confused with the Center of Mass of each link. Although the links are identical in shape, their physical properties are not. Furthermore, the coordinate frame o_m describes the connection point of the links, and finally, coordinate frame o_c is the center of rolling motion (CORM).

5.2.2 Derivation of Model-1

This undersection will show the derivation of Model-1 in detail. The process of deriving each model is almost identical, and therefore the decision has been made to show this process for only one model. There are, however, some important aspects of some of the models, which will be treated in detail in later sections.

5.2.2.1 Deriving the Forward Kinematics

Model-1 (Figure 5.5) is used when the endpiece of Link-1 is in contact with ground. The parameters are listed in Table 5.3.

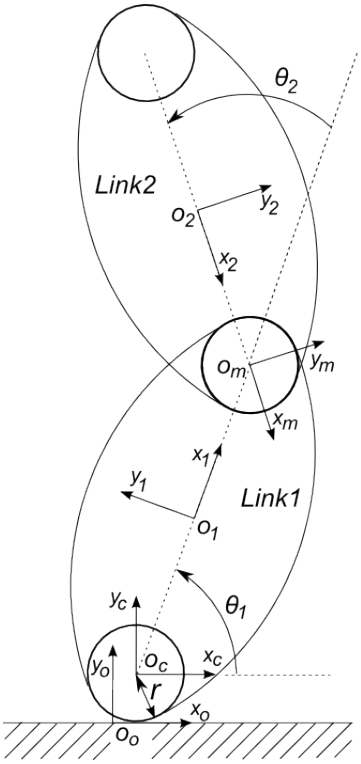

Figure 5.5: Model-1

θ_1:	The absolute angle of Link-1
θ_2:	The angle of Link-2, relative to Link-1
o_0:	Coordinate frame fixed to ground (world frame)
o_1:	Coordinate frame fixed to the GC of Link-1
o_2:	Coordinate frame fixed to the GC of Link-2
o_m:	Coordinate frame fixed to Link-2, and centered at the joint between the two links
o_c:	Coordinate frame fixed to the center of the rolling motion
r :	The radius of the endpiece of Link-1

Table 5.3: Parameters of Model-1.

The transformation matrices are found to be

$$H_c^0 = \begin{bmatrix} 1 & 0 & 0 & (\frac{1}{2}\pi - \theta_1)r \\ 0 & 1 & 0 & r \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

$$H_1^c = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & L_1 cos(\theta_1) \\ sin(\theta_1) & cos(\theta_1) & 0 & L_1 sin(\theta_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

$$H_m^1 = \begin{bmatrix} -cos(\theta_2) & sin(\theta_2) & 0 & L_1 \\ -sin(\theta_2) & -cos(\theta_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

$$H_2^m = \begin{bmatrix} 1 & 0 & 0 & -L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

It follows that

$$H_1^0 = H_c^0 H_1^c \tag{5.5}$$

$$H_2^0 = H_c^0 H_1^c H_m^1 H_2^m \tag{5.6}$$

The location of the COM of Link1 and Link2 are given as

$$p_{COM1}^1 = \begin{bmatrix} L_{g1} - L_1 & 0 & 0 \end{bmatrix}^T \tag{5.7}$$

$$p_{COM2}^2 = \begin{bmatrix} L_2 - L_{g2} & 0 & 0 \end{bmatrix}^T \tag{5.8}$$

Equation (5.7) and Equation (5.8) are expressed relative to o_0:

$$p_{COM1}^0 = H_1^0 \begin{bmatrix} (p_{COM1}^1)^T & 1 \end{bmatrix}^T \tag{5.9}$$

$$p_{COM2}^0 = H_2^0 \begin{bmatrix} (p_{COM2}^2)^T & 1 \end{bmatrix}^T \tag{5.10}$$

where

$$p_{COM1}^0 = \begin{bmatrix} x_{COM1}^0 & y_{COM1}^0 & 0 & 1 \end{bmatrix}^T \tag{5.11}$$

$$p_{COM2}^0 = \begin{bmatrix} x_{COM2}^0 & y_{COM2}^0 & 0 & 1 \end{bmatrix}^T \tag{5.12}$$

Differentiation of p_{COM1}^0 and p_{COM2}^0 with respect to time, yields

$$\frac{d}{dt} p_{COM1}^0 = \begin{bmatrix} \dot{x}_{COM1} & \dot{y}_{COM1} & 0 & 1 \end{bmatrix}^T \tag{5.13}$$

$$\frac{d}{dt} p_{COM2}^0 = \begin{bmatrix} \dot{x}_{COM2} & \dot{y}_{COM2} & 0 & 1 \end{bmatrix}^T \tag{5.14}$$

Finally, the translational velocity of the two COM can be expressed as

$$v_1 = \begin{bmatrix} \dot{x}_{COM1} & \dot{y}_{COM1} \end{bmatrix}^T \tag{5.15}$$

$$v_2 = \begin{bmatrix} \dot{x}_{COM2} & \dot{y}_{COM2} \end{bmatrix}^T \tag{5.16}$$

5.2.2.2 Deriving the EOM by the Euler-Lagrange method

The Euler-Lagrange method is described in Section 3.3. This section describes the Euler-Lagrange method specifically for the Acrobot.

The kinetic energy is the sum of the translational- and rotational energy of the system

$$K = E_{trans} + E_{rot} \tag{5.17}$$

The total mass of the Acrobot may be divided into M_1 and M_2, which represent the mass of Link1 and Link2, respectively. The translational energy becomes

$$E_{trans} = \frac{1}{2}M_1 v_1^2 + \frac{1}{2}M_2 v_2^2 \tag{5.18}$$

where v_1 and v_2 are the velocity of the COM for Link1 and Link2, respectively.

The total rotational energy of the system can be described as

$$E_{rot} = \frac{1}{2}J_1\omega_1^2(\dot{\theta}_1) + \frac{1}{2}J_2\omega_2^2(\dot{\theta}_1, \dot{\theta}_2) \tag{5.19}$$

where $\omega_1(\dot{\theta}_1)$ and $\omega_2(\dot{\theta}_1, \dot{\theta}_2)$ represents the angular velocity of Link1 and Link2, respectively. J_1 is the moment of inertia of Link1 about its COM and J_2 is the moment of inertia of Link2 about its COM.

The structure of Equation 5.19 will differ amongst the seven models, because the functions $\omega_1(\dot{\theta}_1)$ and $\omega_2(\dot{\theta}_1, \dot{\theta}_2)$ are composed differently for each model. Table 5.4 shows $\omega_1(\dot{\theta}_1)$ and $\omega_2(\dot{\theta}_1, \dot{\theta}_2)$ for the models.

Model	$\omega_1(\dot{\theta}_1)$	$\omega_2(\dot{\theta}_1, \dot{\theta}_2)$
Model-1	$\dot{\theta}_1$	$\dot{\theta}_1 + \dot{\theta}_2$
Model-2	$\dot{\theta}_1$	$\dot{\theta}_1 + \dot{\theta}_2$
Model-3	$-\dot{\theta}_1$	$\dot{\theta}_2 - \dot{\theta}_1$
Model-4	$-\dot{\theta}_1$	$\dot{\theta}_2 - \dot{\theta}_1$
Model-5	$-\dot{\theta}_1$	$\dot{\theta}_2 - \dot{\theta}_1$
Model-6	$-\dot{\theta}_1$	$\dot{\theta}_2 - \dot{\theta}_1$
Model-7	$-\dot{\theta}_1$	$\dot{\theta}_2 - \dot{\theta}_1$

Table 5.4: Angular velocities for the different models.

The potential energy of the system can be expressed as

$$P = M_1 g y_{COM1} + M_2 g y_{COM2} \tag{5.20}$$

where y_{COM1} and y_{COM2} are the vertical distance from COM of Link1 to ground and COM of Link2 to ground, respectively. The gravity constant is g.

The Euler-Lagrange equation is stated below

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_k} - \frac{\partial L}{\partial \theta_k} = \tau_k \tag{5.21}$$

where $k \in [1, 2]$, τ_k is the force associated with θ_k and L is the Lagrangian.

The result of Equation 5.21 is the equations of motion (EOM) for Model-1.

5.2.2.3 Notes About the Remaining Models

The derivation of the EOM for Model-1, shows the general process for all of the models. The are subtle important differences of course, but these differences does not justify all seven derivations to have its own section in this study. The transformation matrices, images and parameters will therefore not be stated here.

Model-3 however, is unique. When modeling this model, an important decision had to be made, and it is related to the physical composition of the Acrobot. The modeling process is based on the assumption that the geometric shape of the links is identical. This is not problematic for any model, except for Model-3. This model is the active model in Phase-3 and Phase-7. It is used when the joint-endpiece is in contact with ground, and this is the reason for its uniqueness. Which of the links

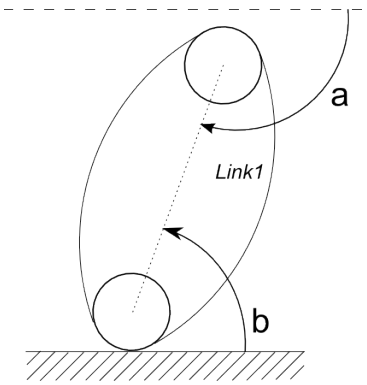

Figure 5.6: Two different ways of measuring θ_1 as the absolute angle of Link-1 with respect to ground. It is obvious that angles a and b have different values.

are in contact with ground at this point? If the links are identical, then both of the links should touch the ground simultaneously.

Because of this problem, the decision was made to assume that Link-1 was touching ground. This decision, in turn, implicates that the joint-endpiece of Link-1 is slightly bigger or different than the joint-endpiece of Link-2. The author of this study was not able to retrieve any information about this problem, and was therefore forced to decide.

The implication of this may be significant if this work is to be used on the physical prototype of the Acrobot. At present, there is not much to be done about this issue, but it is mentioned here for future reference.

5.3 Defining the Configuration Space

Section 5.2 showed how to reduce the Acrobot model from 4DOF to 2DOF. It proceeded to show the derivation of the equations of motion. This section is concerned with defining the configuration space of the resulting models. The configuration space are completely described by θ_1 and θ_2.

Before defining the configuration space, some discussion of the measurements of the angles are in place. As stated earlier in Section 5.2, there are several ways to measure the angles θ_1 and θ_2. This is best illustrated by an example (see Figure 5.6). For models where θ_1 is measured identically, the transitions are trivial. A transition between models which measures θ_1 differently, however, is non-trivial. Consider Figure 5.6. The Acrobot is standing on its endpiece. Assume that it is now falling to the right. Then, at some point, it will be the sidepiece of the link which is in

contact with ground. Assume further that Model-A is used when it stands on its endpiece, and Model-B is used when its lying on its sidepiece. Model-A measures θ_1 as b, and Model-B measures θ_1 as a. It is obvious that $a \neq b$ at the transition point. Therefore, the value of θ_1 has to be modified somehow before it is used in Model-B. From this example, it can also be seen that $\dot{a} > 0$ and $\dot{b} < 0$, which means that $\dot{\theta}_1$ must also be modified in the transition. The seven different models may be divided into two sub-groups based on the way it measures θ_1. Model-1 and Model-2 belongs to Group-A, and the remaining five models belong to Group-B. Now, the only non-trivial transitions occurs between models of different sub-groups.

The process of defining the configuration space is based on analysing the individual models. Images of these models may be found in the electronic appendix, or by sending an email request to *delern_ @hotmail.com*. The resulting configuration space is shown in Figure 5.7.

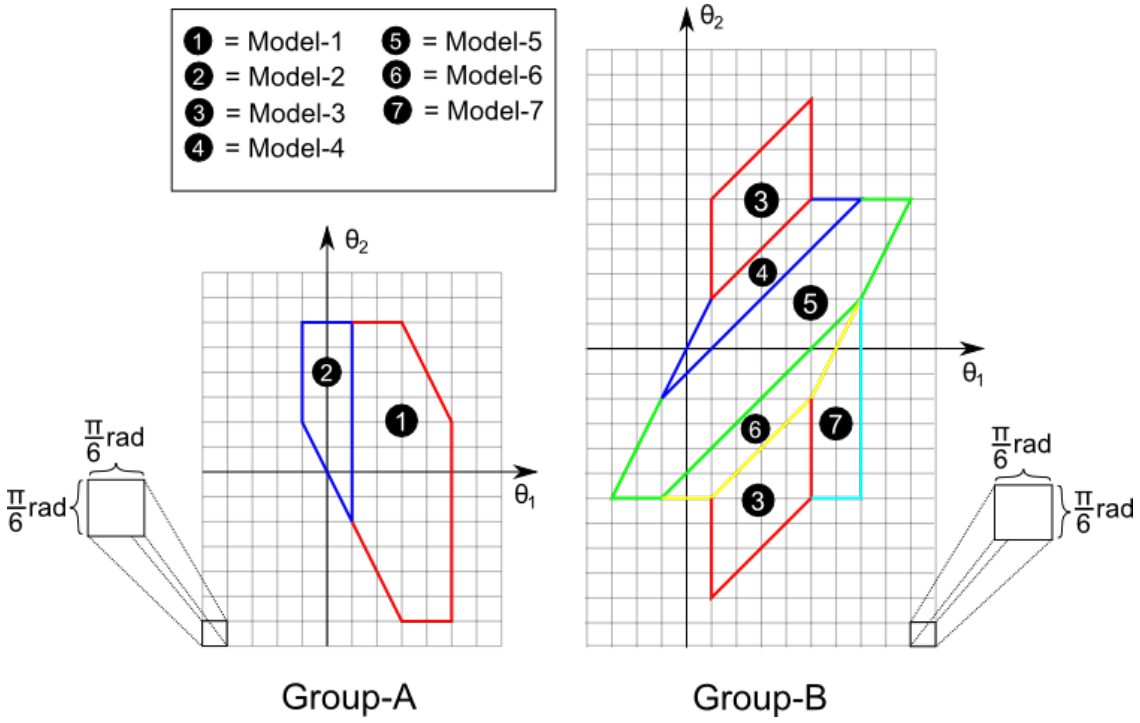

Figure 5.7: Configuration space of the seven models, divided into Group-A and Group-B. The coloured lines defines closed areas where each model is active and valid.

Having defined the configuration space, the system with impulse effects may be

stated. This is the topic of the next section.

5.4 The System with Impulse Effects

The equations of motion for the seven models can be expressed in the following manner

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = \tau \tag{5.22}$$

where $\theta = [\theta_1, \theta_2]^{T3}$ and $\tau = [0,\ u]^T$. These models are systems of second order differential equations. It is well known, that any higher-order system may be reduced to a larger seg of first-order differential equations, by introducing a new state vector. Therefore, by assuming no external torque, Equation 5.22 may be written in the form

$$\dot{x} = f(x) \tag{5.23}$$

where $x = [x_1, x_2, x_3, x_4]^T = [\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2]^T$ is the new state vector. The RMWOI is comprised of eight phases, and to denote the EOM of Phase-N we use the following notation

$$\dot{x}_{(N)} = f(x)_{(N)} \tag{5.24}$$

where $x^{(N)}$ is the state vector of Phase-N and $f(x)^{(N)}$ is the EOM of the model corresponding to Phase-N.

By using the same notation for the impact map and impact surface of Phase-N, the complete system with impulse effects may be written as

$$\Sigma^{(N)} : \begin{cases} \dot{x}^{(N)} = f(x)^{(N)} & x_-^{(N)} \notin S^{(N)} \\ x_+^{(N+1)} = \triangle^{(N)}(x_-^{(N)}) & x_-^{(N)} \in S^{(N)} \end{cases}, \ N \in [1, 8], \ x_+^{(9)} = x_+^{(1)} \tag{5.25}$$

where $S^{(N)}$ is the impact surface of Phase-N, $\triangle^{(N)}$ denotes the impact map from Phase-N to Phase-(N+1), $x_+^{(N+1)}$ denotes the state vector of Phase-(N+1) just after the impact, and $x_-^{(N)}$ denotes the state vector of Phase-N just before the impact.

The impact surfaces $S^{(N)}$ depends only on the configuration variables θ_1 and θ_2, which corresponds to x_1 and x_3, respectively. By analysing the configuration space,

[3] This is the robot equation of motion. In this study, the decision has been made to denote the configuration variables by θ instead of the more usual q.

the impact surfaces $S^{(N)}$ are found, and listed in Table 5.5 together with the impact maps.

Phase	$S^{(N)}$	$\triangle^{(N)}$
Phase-1	$S^{(1)} := \{x^{(1)} \mid x_1^{(1)} = \frac{\pi}{6}\}$	$x_+^{(2)} = Ix_-^{(1)}$
Phase-2	$S^{(2)} := \{x^{(2)} \mid x_1^{(2)} = -\frac{\pi}{6}\}$	$x_+^{(3)} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_-^{(2)}$
Phase-3	$S^{(3)} := \{x^{(3)} \mid x_3^{(3)} = \frac{\pi}{6} + x_1^{(3)}\}$	$x_+^{(4)} = Ix_-^{(3)}$
Phase-4	$S^{(4)} := \{x^{(4)} \mid x_3^{(4)} = -\frac{\pi}{6} + x_1^{(4)}\}$	$x_+^{(5)} = Ix_-^{(4)}$
Phase-5	$S^{(5)} := \{x^{(5)} \mid x_3^{(5)} = -\frac{5\pi}{6} + x_1^{(5)}\}$	$x_+^{(6)} = Ix_-^{(5)}$
Phase-6	$S^{(6)} := \{x^{(6)} \mid x_3^{(6)} = -\frac{7\pi}{6} + x_1^{(6)}\}$	$x_+^{(7)} = Ix_-^{(6)}$
Phase-7	$S^{(7)} := \{x^{(7)} \mid x_1^{(7)} = \frac{5\pi}{6}\}$	$x_+^{(8)} = Ix_-^{(7)}$
Phase-8	$S^{(8)} := \{x^{(8)} \mid x_1^{(8)} = \frac{7\pi}{6}\}$	$x_+^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_-^{(8)} + \begin{bmatrix} -\frac{\pi}{3} \\ 0 \\ 0 \\ 0 \end{bmatrix}$

Table 5.5: Numerical values for the impact surfaces of the eight phases of the RM-WOI, $S^{(N)}$.

5.5 The Reduced System Dynamics

By imposing a virtual constraint function on the EOM of the models, the reduced system dynamics may be derived

$$\alpha(s)\ddot{s} + \beta(s)\dot{s}^2 + \gamma(s) = 0 \tag{5.26}$$

where s can be chosen equal to either θ_1 or θ_2, as described in Section 3.4.2. The structure of the constraint function $\varphi(s)$ are chosen as a Bezier curve. The definition of a Bezier curve is restated here for convenience.

$$B(t) = \sum_{i=0}^{n} \begin{pmatrix} n \\ i \end{pmatrix} (1-t)^{n-i} t^i P_i \ , \ t \in [0,1] \tag{5.27}$$

The Bezier curve evolves as the parameter t advances from 0 to 1. In this study, this parameter will be modified to be a function of s, as shown in the following

equation

$$t = \frac{s - s_b}{s_e - s_b} \tag{5.28}$$

where $s \in [s_b, s_e]$ is the span of the Bezier curve. Figure 5.8 shows an example of an arbitrary Bezier curve for the configuration space of Model-1.

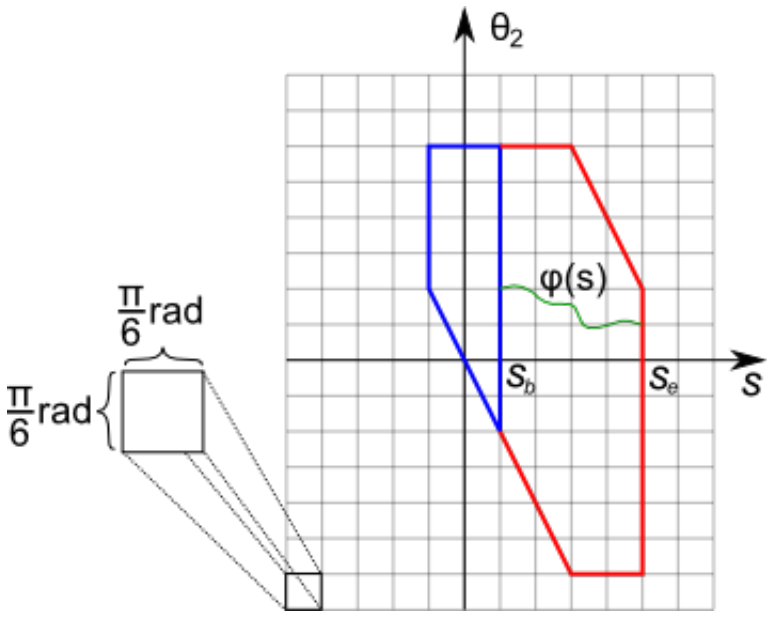

Figure 5.8: Example of Bezier curve in the valid configuration space of Model-1 where $\theta_2 = \varphi(s)$, $s = \theta_1$, $s_b = \frac{\pi}{6}$ and $s_e = \frac{5\pi}{6}$.

In the example in Figure 5.8, the virtual constraint function is chosen as $\theta_2 = \varphi(s)$. The function is valid for $s \in [s_b = \frac{\pi}{6}, s_e = \frac{5\pi}{6}]$. Figure 5.8 also shows that $\varphi(s_b) = \frac{\pi}{3}$ and $\varphi(s_e) = \frac{\pi}{6}$, which means that the control points $P_0 = [s_b, \varphi(s_b)]$ and $P_n = [s_e, \varphi(s_e)]$, where n denotes the order of the polynomial.

Now, if the virtual constraint depicted in Figure 5.8 where to be imposed on the EOM of Model-1, then the resulting system (the reduced dynamics system), would be forced to move along the path defined by $\varphi(s)$. This motion assumes, of course, that some generic control law is able to uphold the virtual constraint, $\varphi(s)$. In fact, there is no guarantee that the torque needed to uphold the constraint is feasible. This have to be checked manually, and is a vital part of the process of searching for virtual constraints.

5.5.1 Choice of Virtual Constraint

The virtual constraint function may be chosen as either $\theta_2 = \varphi(s = \theta_1)$ or $\theta_1 = \varphi(s = \theta_2)$. The difference between these to structures are best illustrated by an example.

Consider Figure 5.9.

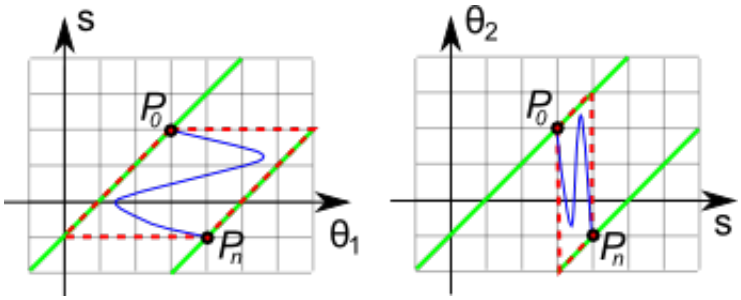

Figure 5.9: Possible virtual constraint functions, $\varphi(s)$. To the left, $s = \theta_2$. To the right $s = \theta_1$.

The to images illustrates a section of the configuration space of Model-5. P_0 and P_n indicates the first and last control point of a Bezier curve. The blue line indicates some Bezier curve, and the dotted red boxes indicates the allowable space for the Bezier curve. There are several aspects of this image to consider.

First of all, in the left-most image, the Bezier curve is allowed to move to the left of P_0. This may be a feature that is necessary for the Acrobot to move from configuration P_0 to P_n. Secondly, the area bounded by the dotted red boxes, are much larger in the left-most image, allowing the Bezier curves more flexibility. Imagine that P_n where to slide downwards along the green border line until it reaches the point just below P_0. For the right-most image, this would result in the dotted red box being reduced to a line, and the only allowable Bezier curve would be the straight line from P_0 to P_n. On the other hand, the dotted red box of the left-most image would increase its area, allowing for even more flexible Bezier curves. These facts indicate that in this case, the choice of constraint function should be $\theta_1 = \varphi(s = \theta_2)$.

It is important, however, to realize that there is no obvious way of knowing the optimal path through Model-5. For all we know, the best path could still be the one depicted in the right-most image. In the searching process, it is important to keep an open mind and not rely to heavily on ones intuition.

5.6 The Search Function

The previous sections of this chapter showed how to derive the reduced system dynamics by imposing virtual constraints. This section is concerned with the search for such virtual constraint functions that will realize the rolling motion without impact.

Before the search function is formally stated, there will be a brief discussion of how to generate a useful first-guess of Bezier curves for the eight phases.

5.6.1 Producing the First-Guess Bezier Curves

The RMWOI is a complex motion. It is not easy to intuitively guess its path in configuration space. The search algorithm needs an initial, closed path in configuration space, and this path should fairly good. The process of producing such a path, is as follows

1. Apply heuristic PD-controllers to the EOM of the models. Each phase should have its own PD-controller.

2. Implement the impact map.

3. Simulate the system numerically, while tuning the controllers by utilizing visual feedback from the Graphical Simulator, until the result is satisfactory close to one cycle of RMWOI.

4. Fit Bezier curves to the resulting path of the simulation. Each phase should have its own Bezier curve.

5. Adjust the final control point of the second Phase-1 ($P_n^{(1)}$) to be equal to the first control point of the first Phase-2 ($P_0^{(2)}$).sketches

The first step is to apply PD-controllers to the EOM of Equation 5.22, which results in the following equation.

$$\ddot{\theta} = -M^{-1}(C(\theta,\dot{\theta})\dot{\theta} - G(\theta) - K_p(\theta - \theta_\star) - K_d(\dot{\theta} - \dot{\theta}_\star)) \tag{5.29}$$

where $\theta_\star$ and $\dot{\theta}_\star$ is the desired value of θ and $\dot{\theta}$, respectively. Only the relative angle between the links (θ_2) may be controlled by external torque. Therefore, the structure of K_p and K_d are as follows

$$K_p = \begin{bmatrix} 0 & 0 \\ 0 & P \end{bmatrix},\ K_d = \begin{bmatrix} 0 & 0 \\ 0 & D \end{bmatrix} \tag{5.30}$$

where P and D are scalar constants.

The second step is to implement the impact map. This may be achieved by continuously checking the state vector of the system while it is being simulated numerically. If the state vector coincides with the switching surface, the numerical simulation of the current phase is stopped, and the state vector is stored. The values of the stored state vector are then mapped using the corresponding impact map function, and used as initial conditions for the model corresponding to the next phase.

The third step is to tune the PD-controllers until a satisfactory motion is achieved. In this study, this task was undertaken by using visual feedback from the Graphical Simulator. The Graphical Simulator can produce an animation of the numerical simulation, and this aspect made it easier for the author to intuitively guess values for the controller gains. The simulation should start at Phase-1, go through all the eight phases, and then return to Phase-1 a second time. The simulation should be stopped just before it enters Phase-2 the second time.

When a satisfactory path has been produced in step three, the fourth step is to fit Bezier curves to this path. The path is divided into nine path-segments, where each path segment corresponds to the phase defined by the configuration space (see Figure 5.7). There will be two path-segments corresponding to Phase-1, and one path-segment for each of the remaining phases. The path-segment of the first Phase-1 is discarded. Then, each of the eight remaining path-segments are fitted to distinct Bezier curves. In this study, this was done by using the least-squares method.

The final step is to adjust the final control point of the second Phase-1 to be equal to the first control point of Phase-2. This is done to produce a set of path-segments that, as a whole, defines a closed path in the configuration space[4].

Notes on this process The reader may wonder why this simulation process starts in Phase-1, and not Phase-2, when the path-segment of the first Phase-1 is discarded. In the simulation studies of this study, when initializing the simulation in Phase-1, the Acrobt was standing up-right, and slightly tipped to the right. Both $\dot{\theta}_1$ and $\dot{\theta}_2$ was initialized to zero. These choices of initial conditions are clearly possible. By initializing the Acrobot in such a feasible state, it seemed reasonable that the motions produced would be valid.

During this process, the Bezier control points are stored. The vector $\theta = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]^T$ at the end of each phase is also stored, and the reason for this will be explained when defining the search function.

[4]This step is not needed if one should be able to produce a closed path from simulation. This, however, is highly unlikely.

5.6.2 Defining the Search Function

Having shown how to produce a feasible first-guess of a closed path in configuration space, the formal definition of the search function may be stated.

$$\begin{matrix} min\, f(x) \\ x \in \mathbb{R}^n \end{matrix} \quad subject\, to\, c_i(x) \geq 0, \;\; i \in I \tag{5.31}$$

where x is the vector of optimization parameters, n is the number of optimization parameters, $c_i(x)$ is the lower- and upper bounds of x and $f(x)$ is the objective function. The parameter vector x is composed of the following items

- Bezier control points.
- The initial condition $\dot{s}_b$ of Phase-2.
- A vector of values of s at the end of each phase ($s_e^{(N)}$).

5.6.2.1 The Parameter Vector

Remark In the following text, the parameter vector and the search procedure will be described in detail. The reader should be aware that in this section, the search procedure is assumed to start at Phase-1. In the actual simulation studies, the initial phase is instead Phase-2. It should also be mentioned that the different simulation studies does not necessarily incorporate all the elements of the parameter vector which are described here.

Explaining the Bezier control points The composition of x is important and will need further investigation, beginning with the Bezier control points. Figure 5.10 shows a closed path in configuration space and some of its Bezier control points.

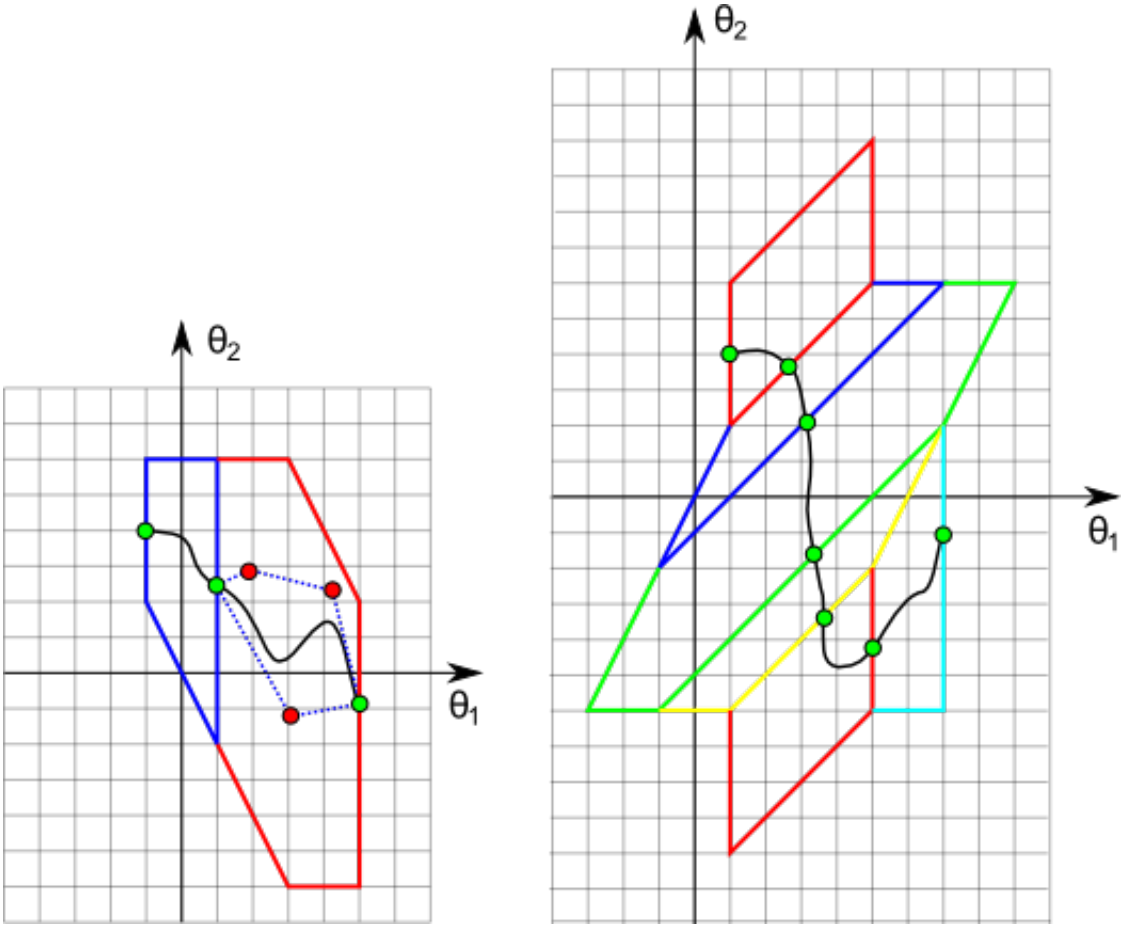

Figure 5.10: Example of a closed path in configuration space. The green dots are Bezier control points at the switching surfaces.

The control points of a fourth degree Bezier curve are shown in the configuration space of Model-1. These control points make up a convex hull, in which the curve is wholly contained, and each curve starts at- and ends at a green dot. The green dots are Bezier control points at the switching surfaces, and the red dots (shown only for Model-1) are intermediate control points. The eight black curves formes a closed path in the total configuration space. Except for the configuration space of Model-1, all intermediate control points have been omitted. The green dots however, are included for each configuration space. They indicate the start- and endpoint of all eight curves. Now, assume that the five control points of Phase-1 (the three red dots and the two green dots) are used in the parameter vector x. Obviously, there are

some constraints as to where to put these points. The right-most green dot should lie on the right-most boundary of the configuration space of Model-1, and the left-most green dot should lie on the left-most boundary of the configuration space of Model-1. The red dots are also constrained, although their placement enjoys more flexibility. They can not be placed outside the valid configuration space of Model-1, because this will result in the convex hull being outside the boundaries. Theoretically, this may lead to the curve going outside of the valid configuration space.

Having investigated the placement of the control points of Phase-1, we consider Phase-2. First of all, the reader is reminded that jumps in configuration space is not allowed except for jumps that are governed by impact laws. A jump in configuration space means, for the case of the Acrobot, that there is a jump in θ_1 or θ_2. This is obviously not physically possible. Therefore, the starting point of the curve in Phase-2 must over-lap the endpoint of the curve in Phase-1, meaning that their numeric values must be indentical. So for Phase-2 (given that the Bezier order is four), only four control points are free variables, compared to five control points in Phase-1. This is true for all the remaining phases of the motion, even for the switches where the impact maps are non-trivial. The endpoint of Phase-2 is transformed according to impact map $\triangle_{(2)}$ (for reference, see Table 5.5) and the resulting values defines the starting control point of Phase-3. It is important to note that for Phase-8, both the starting point and endpoint are already defined[5]. The endpoint has to overlap with the starting point of Phase-1.

As mentioned, there can be no jumps in the configuration space by obvious reasons. There can neither be jumps in the time-derivatives, $\dot{\theta}_1$ or $\dot{\theta}_2$, because this would demand unlimited acceleration. Therefore, the curve must be smooth and connected at all points. This fact makes it possible the reduce the number of free variables even more. This is not necessary to do however, and the reason will be explained later in the text.

Explaining the vector of values of s at the end of each phase ($s_e^{(N)}$). Consider Figure 5.10. Assume that Bezier curves have been generated for both Phase-1 (Model-1) and Phase-2 (Model-2). Assume also that the choice of virtual constraint function for these two phases have been $\theta_2 = \varphi(s = \theta_1)$. The generation of the Bezier curve is according to Equation 5.27, where $t = \frac{s - s_b}{s_e - s_b}$. When the Bezier curve is generated, the values of s_b and s_e must therefore be known. For Phase-1 and Phase-2 this is trivial, because these points are defined by the bondaries of the configuration spaces. These values can be read from Figure 5.10, and are stated below

[5]This is not the case for Simulation-1 of Section 5.7.

$$\begin{aligned} s_b^{(1)} &= \tfrac{5}{6}\pi \\ s_e^{(1)} &= \tfrac{1}{6}\pi \\ s_b^{(2)} &= s_e^{(1)} \\ s_e^{(2)} &= -\tfrac{1}{6}\pi \end{aligned} \tag{5.32}$$

For Phase-3 however, it is not that simple. If the virtual constraint function is $\theta_2 = \varphi(s = \theta_1)$ for Phase-3, then $s_e^{(3)}$ must be on the line defined by $\theta_2 = \frac{1}{6}\pi + s,\ s \in [\frac{1}{6}\pi, \frac{5}{6}\pi]$. Therefore, the value of $s_e^{(3)}$ has to be known before-hand. These values must also be known for Phase-4, Phase-5 and Phase-6, basically all the phases where the switching surface is not a vertical line. When producing the first-guess closed path in Section 5.6.1, these values were stored, and are used as parameters.

Explaining the initial condition $\dot{s}_b$ of Phase-2. When the EOM have been reduced to the reduced system dynamics, the system depends only on s and $\dot{s}$. To initialize the simulation, the initial conditions s_b and $\dot{s}_b$ has to be provided to numerical solver. The simulation may start in any phase, but in this study the choice have been made to always start in Phase-2. One advantage with this, is that $s_b^{(2)}$ is readily given by the configuration space boundaries of Phase-2. The other initial condition, $\dot{s}_b^{(2)}$, may vary. The first value of $\dot{s}_b^{(2)}$ are found from the first-guess process of Section 5.6.1.

These values makes up the basic parameter vector x. During the searching process, this vector may be subject to change. If for example, the order of the Bezier curves is changed from four to six, the parameter vector must contain more parameters. However, the three elements discussed in this section will still be present.

5.6.2.2 The Inequality Constraints

The inequality constraints of the search function (Equation 5.31) is denoted $c_i(x)$. These functions are basically lower- and upper bounds for the parameter vector x. The Bezier control points obviously have upper- and lower bounds, which are the boundaries of the configurations spaces. The boundaries of the $s_e^{(N)}$ is also defined by the configuration spaces. The boundaries of $\dot{s}_b^{(2)}$ are based on the physical constraints of the actuator at the joint.

Notes on Lower- and Upper Bounds In this study, these contraint functions plays only a minor role. Although the configuration spaces defines the boundaries

exactly, it proved to be difficult and time-consuming to define these constraints. The problem arises from the fact that the boundary-values depend on the control points placement on the s-axis. The control points of Phase-1 (Figure 5.10) have different upper- and lower boundaries, which are determined by a function of s. Instead of finding these functions, a less conservative approach was utilized, where constant values were used. In theory, these boundary-values could exclude valid- and include invalid places of control points. This problem was avoided in the design of the objective function.

5.6.2.3 The Objective Function

Having discussed the parameter vector and the inequality constraints, the objective function $f(x)$ may be described. The objective function is by far the most complex aspect of the serach function, and will be explained in detail. Although the objective function changes slightly in the different simulation studies, the structure remains the same.

The basic structure of the objective function is sketched in Figure 5.11.

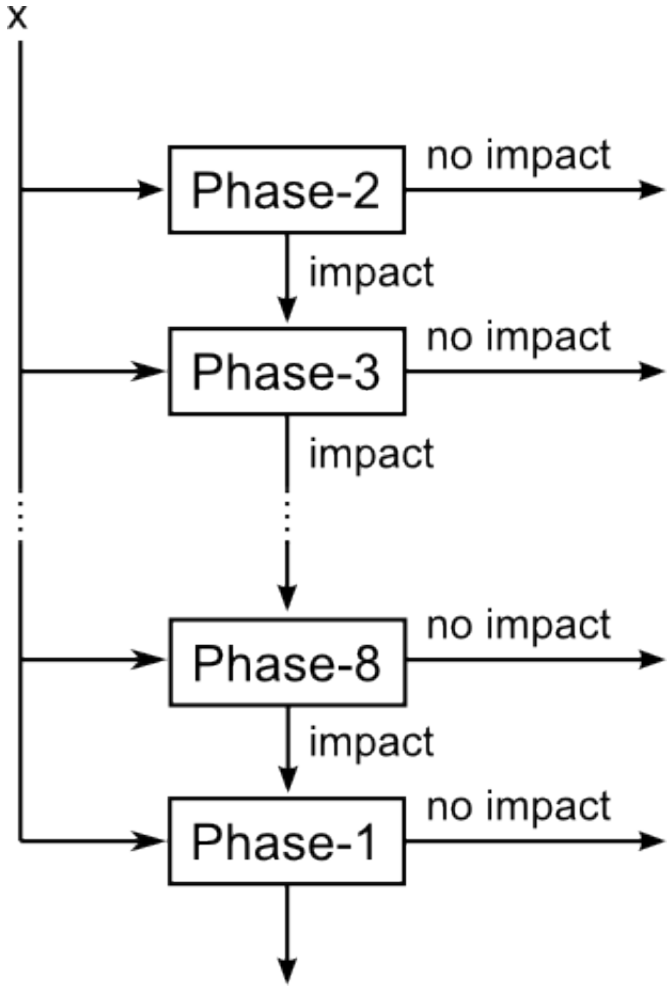

Figure 5.11: Basic structure of the objective function, $f(x)$.

The parameter vector x is somehow passed to the objective function. The elements of x which are related to Phase-2, are picked up, and a Bezier curve based on these elements is produced. The numerical simulation of the reduced dynamics system of Phase-2 begins with initial conditions $\dot{s}_b^{(2)}$ (which is an element of x) and $s_e^{(1)} = \frac{1}{6}\pi$ (which is constant).

The simulation of Phase-2 has two possible outcomes. The simulation hits the impact surface, or it does not. If the impact surface has been hit, the impact map transforms the last state of the solution vector, and passes this new vector to Phase-3. If the result of the simulation is no impact, the function terminates. There may be several reasons for this, depending on different implementations of objective function.

However, as the function terminates, it should always return some scalar function value. This value is produced by the cost function, and is a measure for how well this particular parameter vector did.

Assuming that an impact occured, the same process is then repeated for Phase-3, Phase-4 etc, until it reaches Phase-1. The simulation of Phase-1 also has to outcomes, although slightly different than for the other phases. If there is no impact, the function terminates as usual. If the simulation impacts, then the search function has found a closed path in the configuration space. It has not, however, necessarily found a closed trajectory, which is the goal of the search. To achieve a closed trajectory, the value $\dot{s}_e^{(1)}$ must be equal to $\dot{s}_b^{(2)}$. If this condition is satisfied, the Acrobot model has returned to the state it was in at the beginning of the function. Therefore the cost function at this point should always be

$$cost(\dot{s}_b^{(2)}, \dot{s}_e^{(1)}) = \| \dot{s}_b^{(2)} - \dot{s}_e^{(1)} \|^2 \tag{5.33}$$

where $\| \cdot \|$ is some norm-function. If this function is zero, the search function has successfully found a closed trajectory for the RMWOI. It should be noted that this function is not used for Simulation Study-1 introduced in the next sextion.

5.7 The Simulation Studies

The search process carried out in this study, was mainly based on intuitive guesses and trial and error. Due to the complexity of the mathematical model and the rolling motion without impact, it is hard to theoretically determine the ideal order of the Bezier curves and the ideal cost function. For example, if the path in configuration space was assumed to be a straight line, then a first order Bezier curve might be sufficient. If the path however, was assumed to be a complex sinusoidal function, then of course, a higher order Bezier curve should be utilized. These observations are logical, although they are only based on intuition. No great efforts were made to find optimal Bezier curve orders by use of theoretical analysis, mainly because such an approach was expect to be more time-consuming than useful.

There was some experimenting with the order of the Bezier curve, but the results of this action seemed irrelevantly small and therefore all simulation studies described in this section have Bezier curves of order five. This choice is justified by the results of the initial first-guess of Section 5.6.1. One typical first-guess path is shown in Figure 5.12.

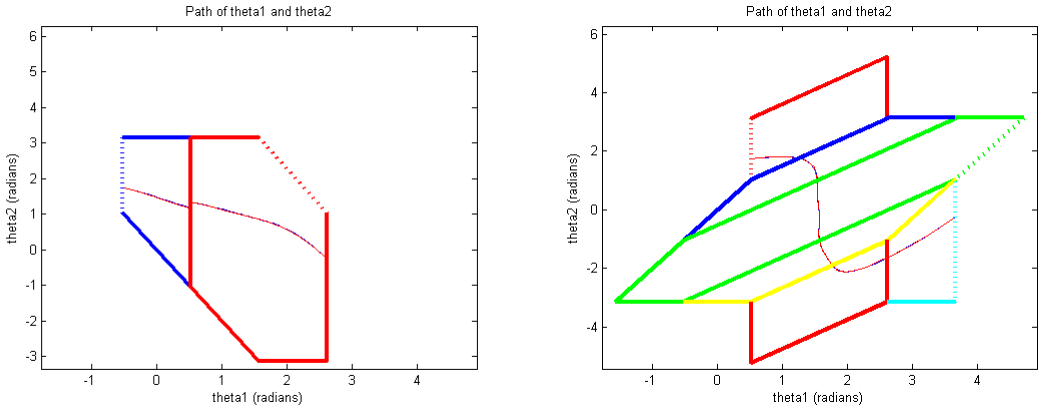

Figure 5.12: Typical initial first-guess of the path of the RMWOI.

From this figure, it can be seen that the behaviour of the paths in the different phases are similar to second- or third order polynomial functions. In fact, third order Bezier curves were tried out. The results of this turned out to capture most of the Acrobots dynamics, although the decision to use fifth order curves were made just to be safe. Although this choice calls for longer simulations, the versatility of the Bezier curve is greatly improved. Having decided that the Bezier curves should always be of order five, it remains to decide on implementations of the objective function and the cost function.

The search process may be divided into four different simulation studies with respect to the choice of objective- and cost function. These simulation studies are listed and described below.

Simulation-1 In the first simulation study, the connection of the path was a part of the cost function, and not ensured by the objective function. The cost function would return a high integer (i.e. 999999) if the simulation of each phase did not hit its impact surface. Also, the boundaries of each individual configuration space were not accounted for in the objective function.

Simulation-2 In this simulation study, the objective function ensured that the produced path would be closed. The boundaries of each individual configuration space were not accounted for in the objective function. The cost function is based on travelled distance. The idea is that a simulation that is able to reach Phase-5 should be less expensive than a path that only reaches Phase-4. If one whole cycle of the motion is produced the cost function is reduced to Equation 5.33.

Simulation-3 This simulation studies is equal to Simulation-2, except for the addition of configuration space boundaries for each phase. This ensures that only legal paths are produced.

Simulation-4 This simulation studies is equal to Simulation-3, but the virtual constraint functions for Phase-5 and Phase-6 are changed. For the first three simulation studies, the virtual constraint functions were $\theta_2 = \varphi(s = \theta_1)$ for all phases. In this simulation study it was changed to $\theta_1 = \varphi(s = \theta_2)$ for Phase-5 and Phase-6.

The next sub sections will describe each of these simulation studies in detail, and discuss the results.

5.7.1 Simulation-1

Simulation-1 is structurally different from the other simulation studies. This is because the objective function of Simulation-1 does not ensure that a terminating search produces a closed path. For the path to be closed, the last control point of the last phase must overlap the first control point of the initial phase. Thus, in the other simulation studies, the last control point of the last phase is calculated in the objective function rather than being part of the parameter vector. In this simulation study however, this control point is part of the parameter vector, and the cost function is left with the responsibility of achieving a closed path.

Figure 5.13 illustrates the cost function of this simulation study.

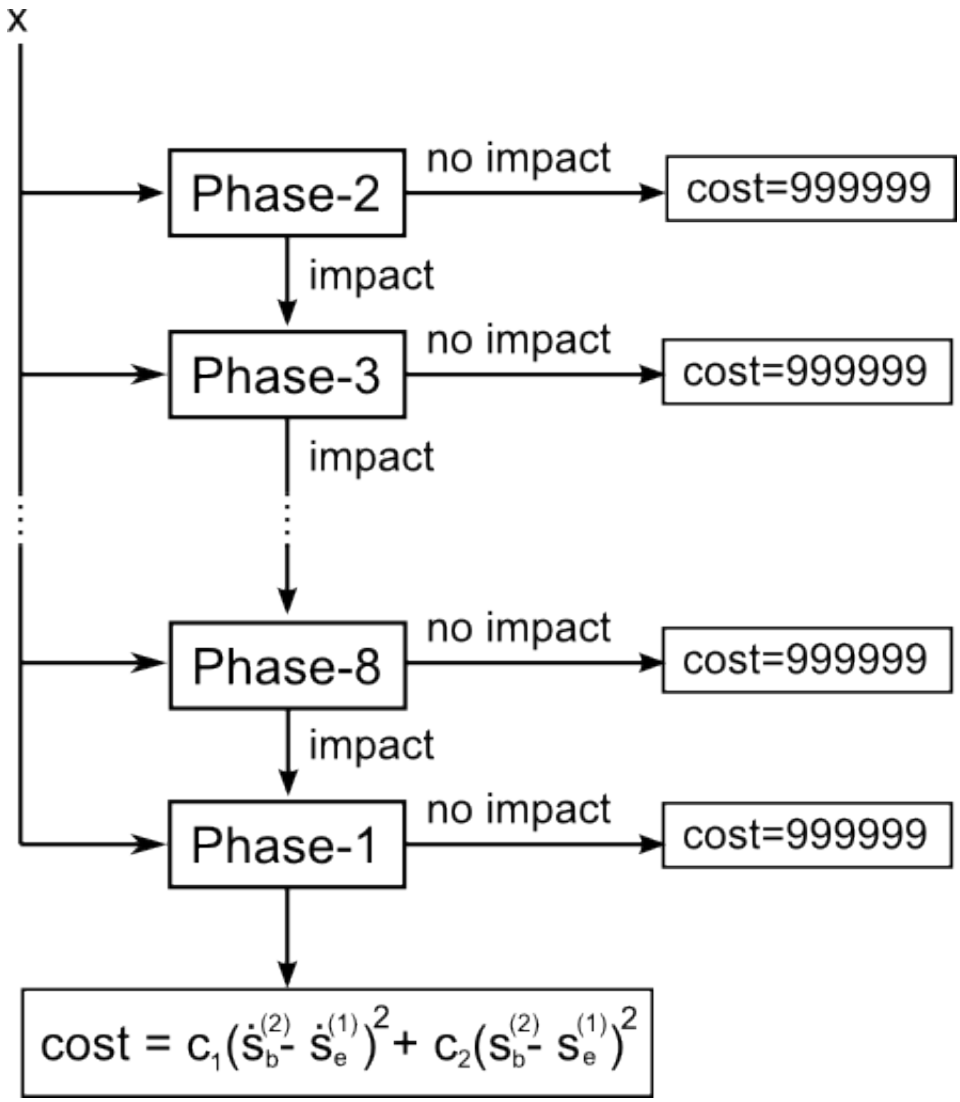

Figure 5.13: Illustration of the cost function for Simulation-1.

First of all, it should be noted that the cost function returns a value of 999999 for every phase if the simulation does not reach the impact surface. The idea was to severly punish any path that did not reach all the phases. The problem here is, of course, that a path reaching Phase-1 (but not hitting the impact surface) is identical in cost to a path that reaches any previous phase. This is obviously problematic, because a path that moves all the way to Phase-1 is clearly closer to realizing the RMWOI than a path that terminates at Phase-2 (the first phase). This approach

was tried out because it was believed that by producing an initial first guess that was very close to one cycle of the RMWOI, the search algorithm would always manage to complete one cycle, meaning that it would always return to Phase-2.

The terminating cost function is

$$cost = c_1(\dot{s}_b^{(2)} - \dot{s}_e^{(1)})^2 + c_2(s_b^{(2)} - s_e^{(1)})^2 \tag{5.34}$$

where $c_1$and c_2 are positive scaling constants. Simulation-1 is the only simulation study which utilizes this structure of the cost function. Because the objective function does not ensure the path to be closed, it has instead been implemented in the cost function by the term $c_2(s_b^{(2)} - s_e^{(1)})^2$. Since the first priority of the search is to produce a closed path, the constant c_2 is typically much larger than c_1.

Discussion of Simulation-1 The results from this simulation study was poor. The search was not even able to produce a closed path. The objective function was oblivious to the confined borders of the configuration space, and as a result the generated paths was sometimes largely outside these boundaries. However, the simulation study in itself is interesting for another reason. It illustrates the initial ignorance and arrogance of the author, and shows how such a search process is largely based on trial and error. This is simulation study was easy to implement, having disregarded the configuration space and by using an extremely simple cost function. By conducting this simulation study, the author realized that a more sophisticated cost function was needed, and that the closedness of the path should probably be implemented as a constraint in the objective function.

5.7.2 Simulation-2

Simulation-2 was motivated by the poor results of Simulation-1. In Simulation-1, the search function was not even able to produce a closed path in the configuration space. This simulation study should first and foremost ensure that a closed path was found before the cost function started searching for the closed trajectory. This is achieved by calculating the last control point of the last phase in the objective function, rather than it being in the parameter vector. Figure 5.14 illustrates the cost function of this simulation study.

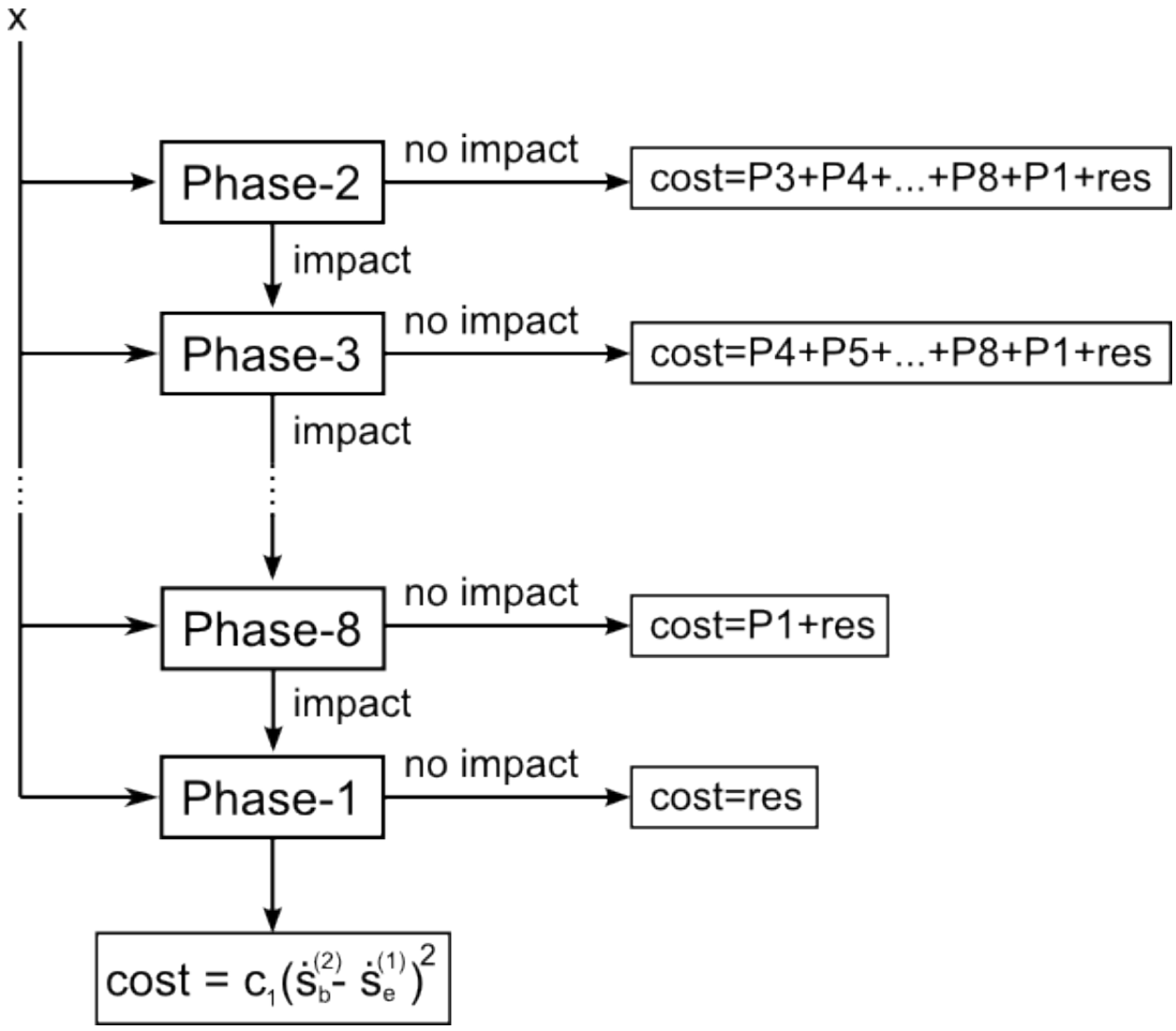

Figure 5.14: Illustration of the cost function for Simulation-2.

Consider Figure 5.14. Assume that some arbitrary values of the parameter vector x produces a path that does not reach the impact surface of Phase-2. Then the objective function terminates with the cost $cost = P3 + P4 + P5 + P6 + P7 + P8 + P1 + res$. These values (except for res) are travel distances corresponding to the different phases. Assuming that all these travel distances are equal to 10, the cost function becomes $cost = 10 + 10 + 10 + 10 + 10 + 10 + 10 + res = 70 + res$. The variable res consists of two parts. The first part is called the residue of the phase.

This is the length of the shortest path from the point on the path which is closest to the impact surface, to the impact surface. To see this, let θ_n be some arbitrary point on the path. The shortest distance from θ_n to the impact surface is along a line perpendicular to the impact surface. Let $d_n = sd(\theta_n)$ be the shortest distance from θ_n to the impact surface. Each point θ_n on the path have a corresponding shortest distance d_n. Let D be the set containing all these distances d_n. The residue is

$$residue = min(D) \tag{5.35}$$

where $min(X)$ is a function that returns the smallest element of the set X. The other part of res is some positive constant value called $offset$. This constant is introduced to ensure that the cost function always returns a larger value than the worst case of the terminating cost function. This is illustrated in Figure 5.15. This figure shows that the cost function is gradually decreasing as the path moves through the phases. The reader may notice that the offset is always larger than the worst case of the terminating cost function. If there was no $offset$, the cost function would have been identically zero when the closed path was achieved. The $residue$ is illustrated by triangles, and shows how the cost function gradually decreases as the the path gets closer to the proceeding phase.

The reader may also notice that there is a small jump in the cost function at the impact surface between phases. This is a deliberate choice. Consider Phase-2 of Figure 5.15. Assume that the constant value $P3$ is gradually shrinking such that the $offset$-box and the $residue$-triangle are moving to the left. If $P3$ shrinks enough, then the worst case of the cost function for Phase-3 would return a larger value than best case of Phase-2. In other words, the search function believes that it is better to terminate at the end of Phase-2 than at the start of Phase-3. This problem must be avoided by clever choices of the constants. With this in mind however, there is no reason for all the P-constants to be of equal size.

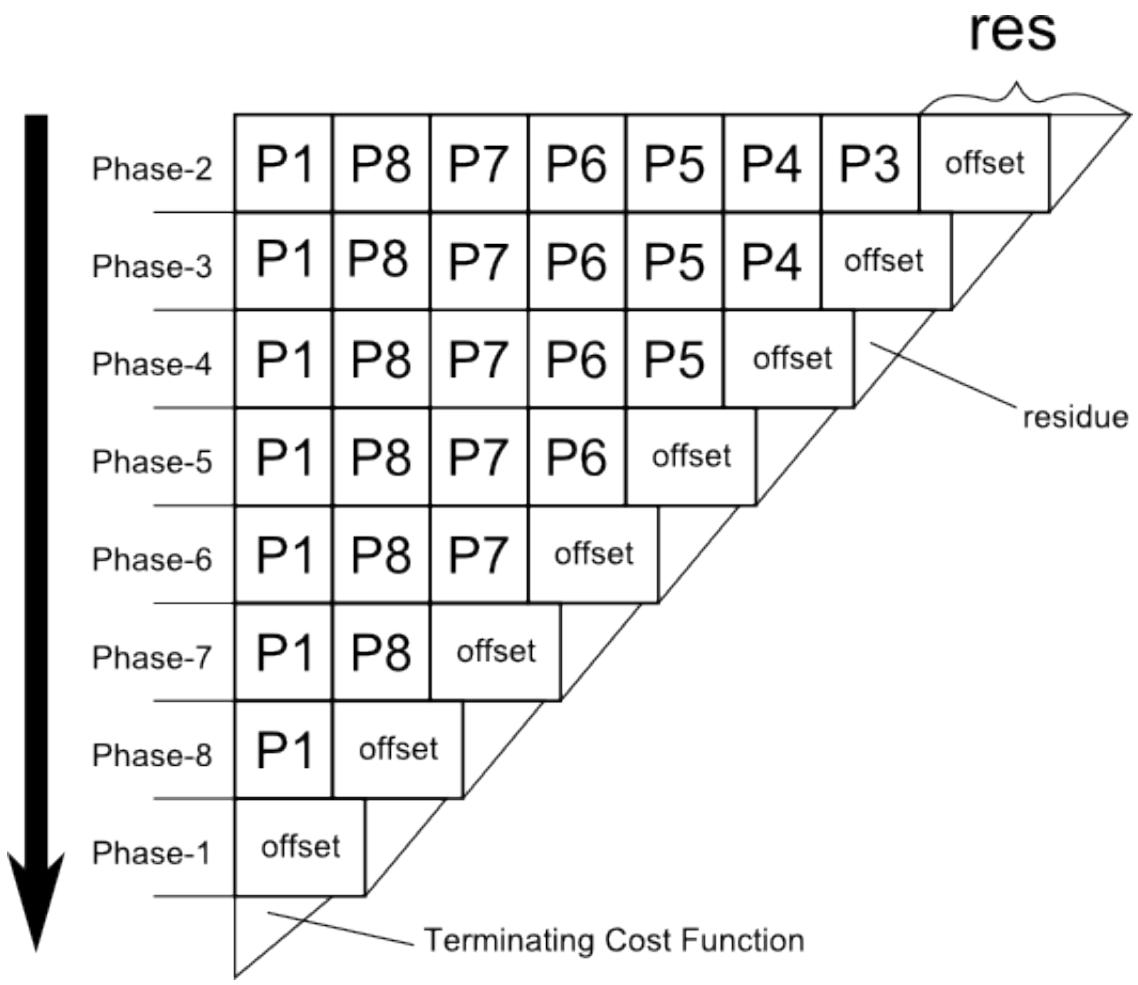

Figure 5.15: Diagram of the cost function of Simulation-2, Simulation-3 and Simulation-4.

Discussion of Simulation-2 Simulation-2 yielded some interesting results. It was discovered that small changes in the initial first-guess path, could make or break the results. A search based on some initial first-guess would result in a path that at best made it to Phase-6. Then, by slightly modifying the first-guess path (by altering the controller values) the search achieved the closed path. If the search managed to find a closed path, it would almost always find a closed trajectory. This observation was surprising to the author. It was believed that finding a closed path would be easy, but that a closed trajectory would be hard to find. Simulation-2 proved this hypothesis to be wrong. It proved to be quite hard to find a closed path, but having

found this path, a closed trajectory would almost always be achieved.

The closed trajectory found by this search function were all very different, but they had one important property in common; they all violated the configuration space in some way. This was also surprising to the author. Preceeding the search, it was believed that the path found by the initial first-guess algorithm, would be the path of least resistance for achieving the RMWOI. The results however, were clearly suggesting that this was not the case. These observations motivated Simulation-3, which adds the constraints of the configuration space to the objective function.

5.7.3 Simulation-3

Simulation-3 employed the same cost function as Simulation-2. The only difference is that the objective function was modified to ensure that the path was wholly contained inside the boundaries of the configuration space. This was done in the following manner.

For each new phase, the objective function initializes a new numerical simulation (in MATLAB). This simulation is initialized with initial values corresponding to the impact laws. If the simulation terminates without hitting the impact phase, the boundaries of the configuration space are ignored as before, and returns a value according to the cost function. If however, the impact surface is reached, the simulation terminates prematurely, and an algorithm checks if all the points of the path is contained within the configuration space of that phase. If this is the case, the search continues as normal. If this is not the case, the search is terminated and returns the value of the cost function.

Discussion of Simulation-3 Simulation-3 revealed a new problem. The search algorithm would terminate with an error originating from the numerical integrator (ODE45 in MATLAB). A solution to this problem became evident after inspecting the path of the initial first-guess. The path through Phase-5 (Model-5) seemed to be almost parallell to the θ_2-axis. According to Section 5.5.1, choosing the virtual constraint function as $\theta_2 = \varphi(s = \theta_1)$, would inhibit the flexibility of the path.

5.7.4 Simulation-4

Motivated by the observations from Simulation-3, the virtual constraint functions for Phase-5 (Model-5) and Phase-6 (Model-6) were changed to $\theta_1 = \varphi(s = \theta_2)$. The idea was to allow for more flexibility in the path of these to phases. The virtual constraint of the other phases remained as $\theta_2 = \varphi(s = \theta_1)$.

Discussion of Simulation-4 The fourth and final simulation study was initially thought to be successful. The resulting closed path is shown in Figure 5.16. The path in Phase-5 is particularly interesting. Figure 5.17 shows this path in more detail.

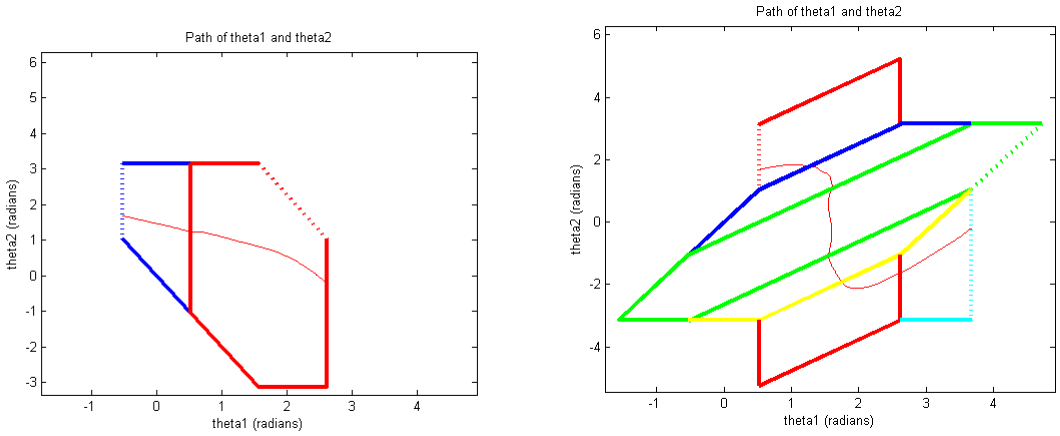

Figure 5.16: Path of the rolling motion without impact.

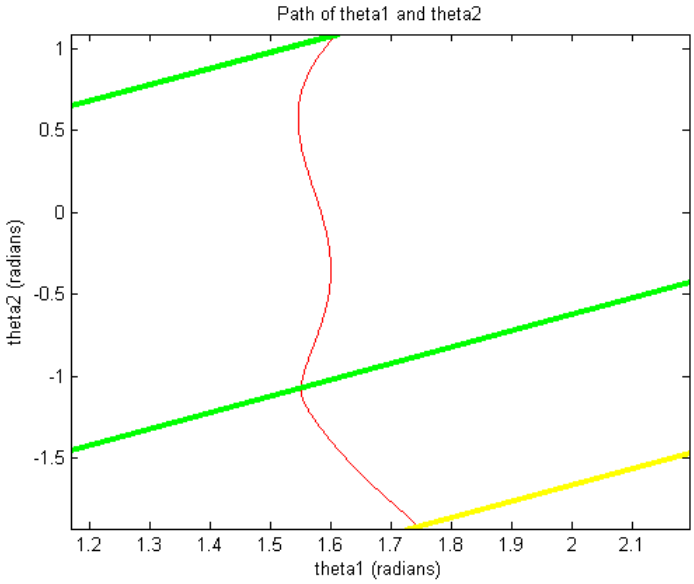

Figure 5.17: Path of Phase-5 and Phase-6.

Figure 5.17 illustrates the importance of changing the virtual constraint function

for Phase-5. The path in Phase-5 looks like a third- or fourth order polynomial function in θ_2. This behaviour could not be achieved if the the constraint function was $\theta_2 = \varphi(s = \theta_1)$.

On closer inspection of the evolution of the variables however, it becomes clear that the result is not a closed trajectory. Figure 5.18 shows the evolution of θ_1, θ_2, $\dot{\theta}_1$ and $\dot{\theta}_2$ for Simulation-4.

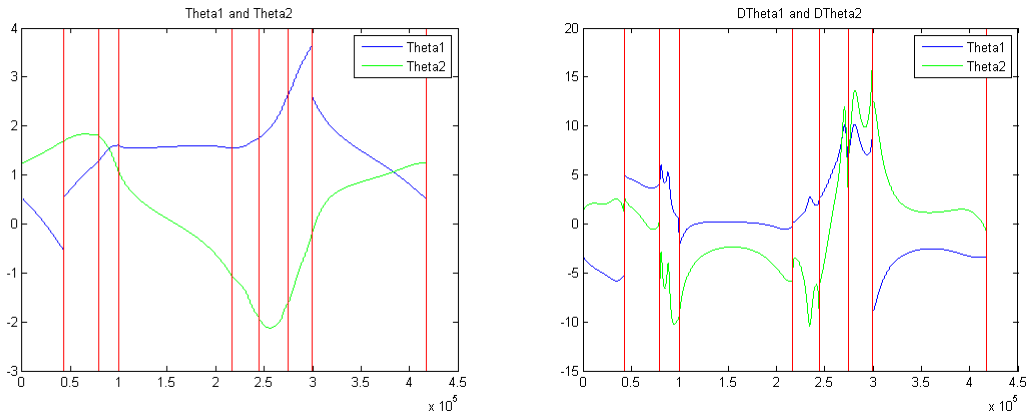

Figure 5.18: The evolution of θ_1, θ_2, $\dot{\theta}_1$ and $\dot{\theta}_2$ for Simulation-4. The vertical red lines indicate the switching surfaces. The left-most red line is the switching surface between Phase-2 and Phase-3.

The left-most image of Figure 5.18 shows the evolution of θ_1 and θ_2. The green curve is continous, and the blue curve is piecewise continous except for at the switching surfaces wich are governed by non-trivial update laws. This indicates that the path in configuration space is closed, which is a necessary condition for the desired closed trajectory.

The right-most image however, shows unwanted behaviour of the angular velocities. There are clearly jumps in both $\dot{\theta}_1$ and $\dot{\theta}_2$. There should, according to the update laws, be jumps in $\dot{\theta}_1$ at the switching surfaces between Phase-2 and Phase-3, and Phase-7 and Phase-8. There should however, not be any jumps in $\dot{\theta}_2$. Figure 5.19 shows in detail the switching surface between Phase-2 and Phase-3.

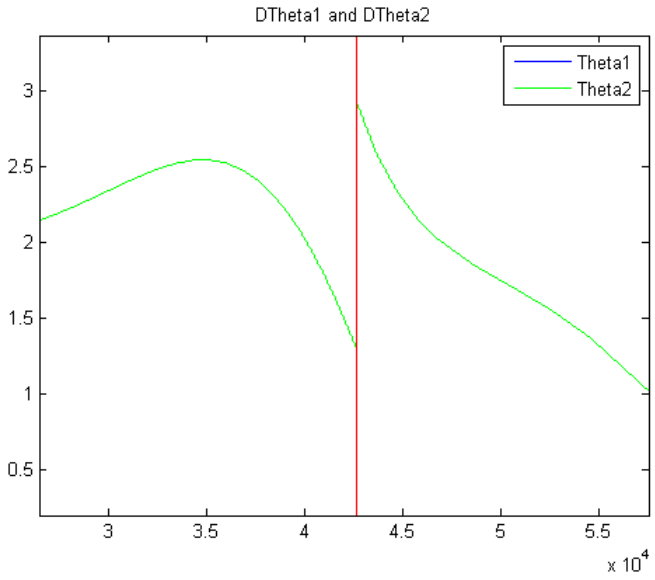

Figure 5.19: Discontinuity in $\dot{\theta}_2$ at switching surface between Phase-2 and Phase-3 of Simulation-4.

Such a discontinuity would require an infinite torque, which is clearly not physically possible. The reason for this discontinuity is the fact that the objective function does not force the angular velocities to be continous. It only ensures continuity in the path variables.

After this discovery, an attempt to enforce continuity on the angular velocities was made. The idea was to make the second control point of each Bezier curve fixed in a way such that continuity in the angular velocities was enforced. The results were very poor, and are not included in this report. There might be many reasons for this poor result, but the most likely reason is the ever growing complexity of the computer code. If an index in the code is wrong, the search routine would still run, but the results would be ridiculous. A search with a duration of one hour could therefore run without any problem, and only the resulting path would indicate an existing bug. This makes the debugging of the code extremely time-consuming.

Chapter 6

Main Result and Conclusion

6.1 Results and Future Work

The objective of this study has been to find a closed trajectory of the Acrobot with curved links which realizes a rolling motion without impact. To do this, a hybrid system consisting of eight different phases was introduced, as well as a set of update laws. Virtual holonomic constraints were then introduced to reduce the order of the system, such that analytical methods for control design could be applied. Then a numerical search was done to find a closed trajectory. This proved to be a challenging objective, and satisfying results were not achieved. A seemingly closed trajectory was found, but closer inspection revealed that there existed unfortunate discontinuities in the angular velocities. These discontinuities would demand infinite motor torques to realize the motion, and the result was therefore not satisfying. The motions produced by the empirical PD-controllers however, strongly suggests that a rolling motion without impact could be possible to find.

This study has produced a lot of basic knowledge for the future work with the Acrobot with curved links. This achievements are listed below.

1. Mathematical models have been produced, and thouroughly checked.

2. Configuration spaces for these models have been defined, as well as the corresponding update laws.

3. A reasonable strucuture for a path-searching routine has been suggested, and possible sources of error has been discovered. The complexity of the code for the search routine is perhaps the most likely source of error.

These are all factors that might contribute to the future work with the Acrobot, and for details on their derivation, please consult Section 5. The following list suggests objectives for future work.

1. Ensure continuity of the angular velocities. It is highly unlikely that a search routine would "accidentally" end up with continous angular velocities.

2. Structurize the computer code of the search routine. The author has, through this study, discovered the critical need of a structured code. The nature of the search routine is highly complex, and is highly dependent on passing parameters between functions, and indexing large vectors and matrices. Such a code becomes unclear as it grows. The author would suggest to use an object oriented language (i.e. C++) to write the search routine, instead of the more sequential MATLAB-script which has been used in this study.

3. Search for simpler motions. The rolling motion without impact which is considered in this study, is very complex. It may be beneficial to find other motions first, which consists of fewer phases. The realization of such motions might help to gain insight in the complex dynamics of the Acrobot.

6.2 Conclusion

The aim of this study has been to combine the properties of legged locomotion robots and wheeled robots. The subject of this matter has been the Acrobot, which exploits the passive rolling motion of wheeled robots as well as the versatility of legs. The biggest source of energy dissipation for a legged robot, is due to the impacts with the ground. This has been avoided in this study, by designing a continous rolling motion without impact.

The results of this study indicates that the design of locomotion gaits for combined locomotion robots is possible. It remains to see however, whether the energy cost of such robots can compete with the energy cost of either legged- or wheeled robots.

Bibliography

[1] A. Robertsson A. Sandberg A. Shiriaev, J. Perram. Periodic motion planning for virtually constrained eulerlagrange systems, systems and control letters 55, 2006.

[2] Yannick Aoustin Franck Plestan E.R. Westervelt Carlos Canudas-de-Wit Christine Chevallereau, Gabriel Abba and J.W. Grizzle. Rabbit: A testbed for advanced control theory. ieee control systems magazine, vol. 23, no. 5, pp. 57 to 79, 2003.

[3] Christine Chevallereau Jun Ho Choi Benjamin Morris Eric R. Westervelt, Jessy W. Grizzle. Feedback control of dynamic bipedal robot locomotion. boca raton, fl: Crc press, taylor and francis group, 2007.

[4] Daan G.E. Hobbelen and Martijn Wisse. Limit cycle walking. delft university of technology. the netherlands, 2007.

[5] Stephen J. Wright Jorge Nocedal. Numerical optimization. 2. ed. springer, 2006.

[6] Hassan K. Khalil. Nonlinear systems. prentice hall, 2002.

[7] A.D. Kuo. Choosing your steps carefully: Trade-offs between economy and versatility in dynamic walking bipedal robots. ieee robotics and automation magazine, vol. 14, no. 2, pp. 18 to 29, 2007.

[8] M. Vidyasagar Mark W. Spong, Seth Hutchinson. Robot modelling and control. wiley, 2006.

[9] et al. Michael H. Dickinson. How animals move: An integrative view. science 288, 100 (2000). doi: 10.1126/science.288.5463.100, 2000.

[10] Anton Shiriaev Pedro X. La Hera, Leonid B. Freidovich and Uwe Mettin. New approach for swinging up the futura pendulum: Theory and experiments. journal homepage: www.elsevier.com/locate/mechatronics. mechatronics vol. 19, 2009.

[11] Shigeki Nakaura Shimpei Isobe and Mitsuji Sampei. Continuous rolling motion control for the acrobot composed of rounded links. 47th ieee conference on decision and control cancun, mexico, dec. 9-11, 2008.

[12] Mark W. Spong. The swing up control problem for the acrobot. february, 1995.