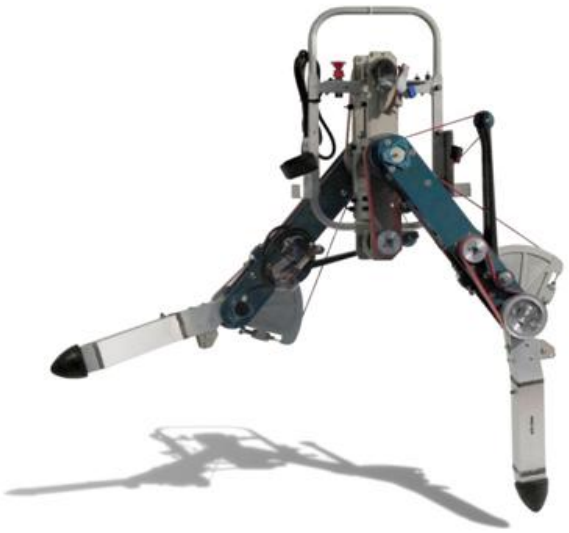
Graduation Report



Angle of Attack Control for Running Robots

TU Delft Delft University of Technology

Karin Griffioen
June 2011

Preface

The first time I saw a walking robot was on the open days in Delft, 7 years ago. It was one of the reasons I chose to study mechanical engineering. I could not have hoped or even imagined then that I would be writing my thesis report today about a running robot. This has been a challenging and very interesting project. I have been stuck at times, but I have loved (almost) every day of it. I feel lucky to have a thesis project that involved so many of the elements that interest me: theoretical and literature research, model simulations, and hands-on experience with a state of the art robot. To see Phides running at the end of this research is the best reward for all the months of hard work.

There are many people who have helped me make this possible. I would like to thank my family and friends, for putting up with me the past few weeks and months, and for listening to endless stories about limit cycles, faulty motors and simulation problems. I am very grateful to have loving and supporting parents, who have made it possible for me to enjoy my studies and student life to the fullest in the past years. Thanks to Martijn, for inspiring me to choose BioRobotics and for pushing me to always strive for my best possible work. Thanks to my fellow Running Robot Researchers, Leonard and Niels, for all the helpful and insightful discussions and for all the coffee breaks. Most of all I would like to thank my supervisor Daniël, for allowing me to find my own way in this research, but always being willing and able to guide me whenever I was lost.

The Hague, 15 June 2011
Karin Griffioen

Graduation Paper

Angle of Attack Control for Running Robots

Delft Biorobotics Laboratory, Mechanical Engineering, Delft University of Technology
Karin Griffioen, Advisor: Daniël Karssen
June 15, 2011

***Abstract*—For running robots, angle of attack control is an important part of the controller, as the angle of attack has a large influence on the state of the running robot. Over the last 25 years, many angle of attack controllers have been introduced. However, it is unknown what the relative performance is of these controllers. The goal of this study is to determine the best angle of attack controller in terms of disturbance rejection and robustness. In this study, we investigated the disturbance rejection and robustness of the six most used angle of attack controllers. We found that dead-beat control provides by far the best disturbance rejection. Furthermore, dead-beat control also performs very well in terms of robustness; its disturbance rejection is hardly affected by model and sensor errors. For these reasons, we consider dead-beat control the best angle of attack controller.**

***Index Terms*—running robot, SLIP model, angle of attack**

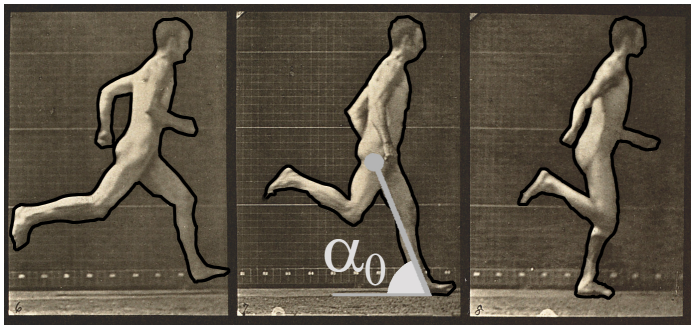


Figure 1: Photographs of a human running. The middle picture shows the moment of touchdown. The angle of attack α_0 is defined as the angle between the leg and the ground at the moment of touchdown. Adapted from [3].

I. Introduction

LEGGED robots have been of interest to the robotics community for quite a while. Many walking robots have been designed in the past years. Inspired by the successes in walking robots, attention has been starting to shift recently to running robots. The reason for studying running robots is twofold. First of all, legged robots provide more mobility than e.g. wheeled robots. It is much more difficult to navigate challenging terrain such as stairs, holes, etc. for a wheeled robot than it is for a legged robot. When a legged robot is walking, its speed is limited to $\sqrt{gl}$ [1], where l is the leg length. To increase the speed above this limit, the robot has to start running. Thus, a running robot can provide better mobility than a wheeled robot, at a higher speed than a walking robot. The second reason to study running robots is that they can be a model for human (and animal) running. The complex task of locomotion is performed easily by humans, but still poorly understood by science. A better understanding of the science of locomotion makes it easier to help people who now have trouble or are unable to walk and run. Knowledge on how humans run could for example help in the design of prostheses or rehabilitation techniques. As shown by famous 'blade runner' Oscar Pistorius, successful running prostheses do exist, but little is known as to why these prostheses work so well [2]. Having a running robot can facilitate research in this area.

When building a humanoid running robot it is important to consider how it can best be controlled. Since the early days of running robotics, control of a running robot has often been split in three parts [4]. First, the upper body posture is controlled by applying a hip torque. Second, leg thrust is applied during the stance phase or at lift-off to get the robot to a certain energy level. Finally, leg placement at touchdown controls the ratio between forward speed and height at the next hop. This is called angle of attack control.

The study presented in this paper focuses on angle of attack control. The angle of attack is defined as the angle between the leg (the line between hip and foot) and the ground at the moment of touchdown, see figure 1. A steep angle of attack will result in high forward speed but low height; a flat angle of attack will result in low forward speed but great height at the next hop.

Ideally, one would like to analytically solve the equations of motion of a running model and determine an ideal angle of attack from this solution. This ideal angle of attack will get the model exactly to a certain desired state, for example a desired forward speed or a desired hopping height. Unfortunately, even for the simplest running model, it is not possible to solve the equations of motion analytically. This has compelled many researchers to find a different way to control the angle of attack. Many angle of attack controllers have been designed to date (e.g. [4], [5], [6], [7]). All these controllers have been shown to provide stable running patterns. However, they have never been properly compared in terms of performance. This study will identify and compare the most researched angle of attack controllers, in order to determine the best angle of attack controller for certain performance indicators. An extensive literature study is performed to identify existing angle of attack controllers and to find possibilities for a new controller. All controllers are compared on a basic level to identify the most promising controllers. The promising controllers are then investigated in-depth, to determine the best angle of attack controller based on several performance indicators, namely disturbance rejection and robustness. These indicators show how well a robot performs in real-world situations, such as being

disturbed and having errors in sensor and/or model inputs.

The remainder of this paper is organized as follows: Section II introduces two important running robot models that will be used throughout this study, as well as a real running robot that will be used for experiments. Section III introduces all angle of attack controllers that are researched in this study, and identifies the most promising controllers. Section IV describes how the most promising controllers will be implemented on the models and real robot. Section V presents the disturbance rejection results of the most promising controllers, both on a simple and on a realistic model. Section VI presents the robustness of the controllers. Section VII presents a parameter study for the controllers. Section VIII presents results of experiments on the real running robot. Finally, discussions and conclusions of this study can be found in Sections IX and X respectively.

II. Running models and robot

For this study, several running models and a real running robot will be used. A simple running model, the Spring Loaded Inverted Pendulum (SLIP) model, is used to test the disturbance rejection and robustness of the angle of attack controllers. This model is described in section II-A. A realistic model, which closely resembles a running robot, is described in section II-B. This realistic model is used to verify that the conclusions from the SLIP model are still valid in a more realistic environment. To confirm that results obtained from the simulations are also valid in the real world, the promising controllers are implemented on a running robot. The running robot used in this study is described in section II-C.

A. SLIP model

An important model in running robotics research is the Spring Loaded Inverted Pendulum (SLIP) model. This model consists of a point mass m attached to a massless spring with spring constant k and rest length l_0, see figure 2. Even though this model is very simple, it is shown it is a good model for human and robot running, and that the gaits found with this model are similar to human and animal running gaits [8], [9].

For the SLIP model, equations of motion are divided in stance phase and flight phase equations of motion. The flight phase can be characterized as ballistic flight, leading to the following equations of motion:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \end{bmatrix}, \tag{1}$$

where:

$\ddot{x}$ is the horizontal acceleration of the point mass,
$\ddot{y}$ is the vertical acceleration of the point mass and
g is the gravitational acceleration.

For the stance phase, the equations of motion are:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} -\cos\alpha \\ \sin\alpha \end{bmatrix} \frac{F_s}{m} + \begin{bmatrix} 0 \\ -g \end{bmatrix}, \tag{2}$$

where:

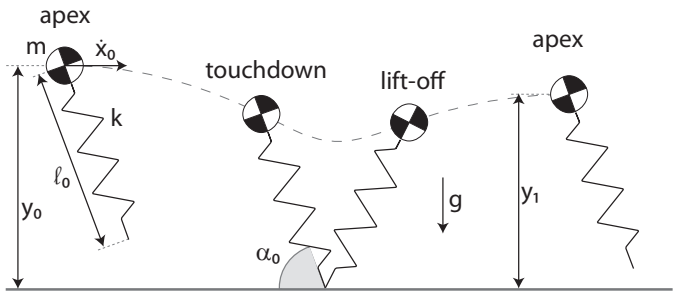


Figure 2: The Spring Loaded Inverted Pendulum (SLIP) model. This model consists of a point mass m, attached to a massless linear spring with spring constant k and rest length l_0. The model is subject to gravitational acceleration g. The highest point of the flight phase is called apex; the instances where the foot touches and leaves the ground are called touchdown and lift-off respectively. The angle of the leg with respect to the ground at the moment of touchdown is called the angle of attack α_0.

F_s is the spring force,
α is the angle between leg spring and ground and
m is the mass of the point mass.

In this study, a linear spring is used, so the spring force is:

$$F_s = k \cdot (l - l_0), \tag{3}$$

where:

k is the spring constant,
l is the spring length and
l_0 is the rest length of the spring.

The transition from flight phase to stance phase is called touchdown. This happens when the foot hits the ground, so when:

$$y = l_0 \sin\alpha_0, \tag{4}$$

where:

α_0 is the angle between leg spring and ground at touchdown, also called the angle of attack

The transition from stance phase to flight phase is called lift-off. This happens when the foot leaves the ground, which is when the leg spring returns to its rest length, so when:

$$l = l_0 \qquad \text{with} \qquad \dot{l} > 0. \tag{5}$$

During the flight phase, the state of the SLIP model consists of horizontal and vertical positions x and y and horizontal and vertical speeds $\dot{x}$ and $\dot{y}$. At the highest point of the flight phase, called the apex, the state of the model is completely described by the height y and the horizontal speed $\dot{x}$. This is because the horizontal position x is irrelevant and the vertical speed $\dot{y}$ is zero by definition, since the apex is the highest point of the flight phase. The system is conservative, since no energy can be lost or gained. There is no actuation system which can add energy to the model, and no impact losses can occur due to the fact that the leg has no mass. For such a conservative SLIP model, the state at the apex can even be described by height y alone, since the horizontal speed $\dot{x}$ is

then coupled to the height through the system energy E_s.

A step for the SLIP model is considered here as the motion from one apex to the next apex. To control a running robot one would like to find a stable *limit cycle* [10] for a step. A limit cycle is considered a series of steps where the state of the runner at the end of each step is exactly equal to the state of the runner at the beginning of the step. To analyze the existence of a limit cycle for a running robot, a mapping from one apex point to the next apex point can be used. Such a mapping is called a Poincaré or return map. The return map S gives the state at the next apex $\mathbf{v}_{i+1}$ as a function of the state at the current apex $\mathbf{v}_i$:

$$\mathbf{v}_{i+1} = S\left(\mathbf{v}_i\right). \tag{6}$$

A limit cycle exists if the return map S has a fixed point $\mathbf{v}^*$ where:

$$\mathbf{v}^* = S\left(\mathbf{v}^*\right). \tag{7}$$

In this study, model parameters for the SLIP model are set at $m = 80kg$, $l_0 = 1m$, $k = 20kN/m$, $g = 9.81m/s^2$, both because of conformance to human values, as well as to facilitate comparison to literature results, where these values are somewhat standard values.

B. Realistic model

The realistic model (see figure 3) used in this study closely resembles a robot runner. Sideways motion of the model is not possible, making it a two-dimensional model. It has two legs, both consisting of an upper and a lower leg, and an upper body. All elements have a distributed mass and rotational inertia. The elasticity of the legs is provided by a torsion spring in the knees. This torsion spring has a non-linear stiffness profile, which is chosen such that it has the same effect as a linear spring between hip and toe. The simulated linear spring stiffness is chosen in accordance with the real robot at $k_{knee,linear} = 5.1kN/m$. The spring in the knee is only active when the relevant leg is in the stance phase. All parameters of the realistic model are chosen in accordance with the parameters of the real robot used in this study, in order to facilitate easy comparison of experimental results. The values of these parameters can be found in table I.

The upper body of the model is attached to the world in such a way that rotation of the upper body is impossible, to eliminate the need to control the upper body posture, which is the subject of a separate study.

The realistic model has four motors: one in each knee and one in each hip. The knee motors are placed parallel to the knee springs. The motors can exert torques in order to position all elements in the desired state. PD-controllers are used to move the respective joint to its desired state. Fifth order splines are used to determine a smooth desired trajectory from the current state to the desired state. Since the realistic model is not conservative like the SLIP model, a control system must ensure that the energy of the model stays at the desired level. In this study, to keep this control as simple as possible, the choice is made to use a constant knee push-off during the second part of the stance phase. This means that the stance knee motor exerts a constant torque when the stance knee angle has a positive angular speed. The value of the torque is determined empirically, so that a resulting limit cycle will have both the desired apex height and speed. To prevent slipping of the stance foot at touchdown, and to prevent the occurrence of liftoff before the stance leg is at its rest length, the torque in the stance hip motor is limited so that there can never be pulling forces between the ground and the stance foot.

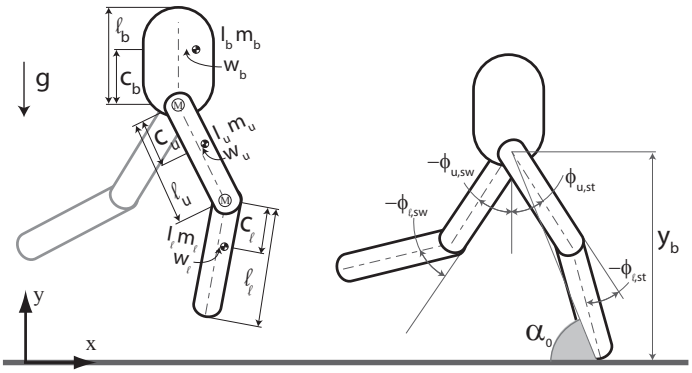


Figure 3: A realistic model of a runner. The model has two upper and two lower legs, and an upper body. All elements have distributed masses and rotational inertia. The upper body is attached to the world so that rotation is not possible. The model is subject to gravitational acceleration g. The rotational springs in the knees are only active when the respective leg is in the stance phase. There are motors in both knees and both hips. For the complex model, the angle of attack α_0 is defined as the angle between the hip-foot line and the ground at the moment of touchdown. The left figure shows the model parameters, parameter values are given in table I. The right figure shows the definition and direction of state parameters.

Table I: Realistic model and Phides parameters.

	body	upper leg	lower leg
Mass m [kg]	7.41	2.54	0.51
Moment of inertia I [kgm^2]	0.080	0.036	0.005
Length l [m]	0.3	0.3	0.3
Vertical offset CoM c [m]	0	0.183	0.139
Horizontal offset CoM w [m]	0	0	0

As for the SLIP model, equations of motion are divided between stance phase and flight phase. During the flight phase, the leg that is supposed to hit the ground next is considered the stance leg; the other one is the flight leg. The flight phase ends when the stance foot hits the ground. At this point, impact equations are calculated and the stance phase commences. The stance phase ends when the leg is at its rest length, simulating an end-stop in the knee. Impact equations are calculated, the stance leg then becomes the flight leg and vice versa, and the next flight phase begins.

C. Running robot Phides

For this study, we use a running robot called Phides (see figure 4). Phides is about 0.6m high from foot to hip. It consists of an upper body and two legs, both consisting of an upper

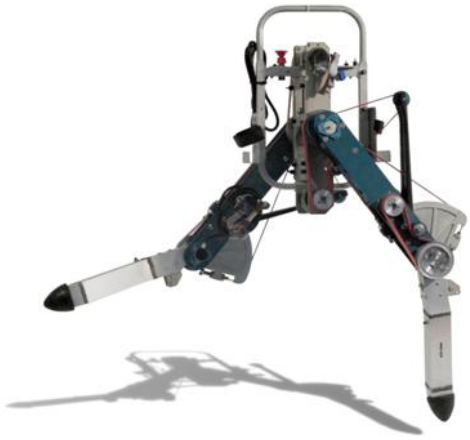


Figure 4: Running robot Phides.

leg and a lower leg. The elasticity of the legs is provided by a rotational spring in the knee. For the parameter values of Phides see table I. Phides has four motors: two in the knees, and two in the hips. All motors are Maxon RE35 30V motors with Maxon gearboxes with a 1:55 gear ratio. The knee motors are placed parallel to the knee springs. Furthermore, the knee motors are placed in series with a torsion bar, so that a series elastic actuator [11] is formed, ensuring smooth torque control in the joints. At the hip, Phides is attached to a boom construction, which prevents sideways motion of the robot, in essence making it a two-dimensional robot. The robot can run in circles on the ground, or it can run on a treadmill. A construction is in place at the hip, which makes it possible to prevent angular movement of the upper body. This is activated during this study.

III. Angle of attack controllers

A literature study is performed, which identifies the five most researched types of angle of attack controllers, as well as an opportunity for a new type of angle of attack control. These controllers are compared on a basic level, to identify the most promising controllers. In this comparison, all controllers are compared to an ideal angle of attack, over a range of initial conditions. All angle of attack controllers are described in sections III-A-III-F. The basic comparison of the controllers is described in section III-G.

A. Constant angle of attack controller

The simplest angle of attack controller that exists is the constant angle of attack controller [5]. This controller sets the angle of attack to a constant value, independent of any input:

$$\alpha_0 = \text{const} \tag{8}$$

where:

α_0 is the angle between the leg and the ground at touchdown, also called the angle of attack.

Seyfarth et al. [5] investigated whether such a simple control scheme can generate stable running patterns. They performed a simulation of the SLIP model with constant angle of attack, which shows that with a proper combination of model parameters and initial conditions, constant angle of attack control can produce a stable running motion.

B. Raibert controller

To control the angle of attack, Raibert introduced the concept of *neutral point* [4]. The neutral point is defined as 'the unique foot position that results in zero net forward acceleration', where *net forward acceleration* is defined as the difference between forward speed at touchdown and at lift-off. Due to the non-integrability of the equations of motion of running models, the neutral point cannot be determined exactly. Raibert estimated the neutral point based on forward speed and stance time of the previous step:

$$x_{f0} = \frac{\dot{x} T_s}{2}, \tag{9}$$

where:

x_{f0} is the horizontal distance between the foot and center of mass for the neutral point,
$\dot{x}$ is the forward speed and
T_s is the stance time of the previous stance phase.

This estimation of the neutral point can be understood as follows: Assuming that the forward speed during stance is constant and that the stance time stays the same for all stance phases, $\dot{x}T_s$ gives the distance traveled during the stance phase. Assuming furthermore that the stance phase is symmetric, $\frac{\dot{x}T_s}{2}$ gives the neutral point.

It may not always be the objective to place the foot at the neutral point, which gives zero net forward acceleration. Therefore, the foot can be displaced from the neutral point according to the difference between current and desired forward speed. This leads to the following equation for foot placement:

$$x_f = \frac{\dot{x} T_s}{2} + k_{\dot{x}}(\dot{x} - \dot{x}_d). \tag{10}$$

where:

x_f is the desired horizontal distance between the foot and the center of mass,
$\dot{x}_d$ is the desired forward speed and
$k_{\dot{x}}$ is a control gain.

C. Swing-leg retraction

In human and animal running, it is observed that the front leg rotates backwards just before touchdown [3] (see figure 5). This is called swing-leg retraction.

Swing-leg retraction reduces the horizontal velocity of the foot with respect to the ground at touchdown, and therefore reduces landing impact losses [12]. Seyfarth et al. have shown that, apart from these energy benefits, swing-leg retraction can also improve stability [6]. The reason why swing-leg retraction

Figure 5: Photographs of a human running. Retraction of the swing-leg is observed in these photographs. Adapted from [3] and based on [6].

control can improve stability is that the angle of attack is automatically adapted to a change in height, because of the swing-leg action. An increased hopping height of a runner will result in a longer flight time. This will in turn result in a steeper angle of attack. For a decreased height, the flight time will be shorter, resulting in a flatter angle of attack.

D. Passive dynamic running

Passive dynamically walking robots, which are powered only by gravity, have been researched extensively. McGeer showed that, given the right initial conditions, two-legged robots are able to perform stable walking on a shallow slope, without active control or energy supply other than gravity [13]. Inspired by the success in walking robots, researchers started looking at passive dynamic running. In passive dynamic running, a spring is inserted in the hips of the runner. The energy stored in this spring during the stance phase will move the legs during the flight phase. The idea behind passive dynamic running is that, with proper selection of the hip spring constant, the legs will automatically move to an angle of attack which will result in stable running motions. McGeer looked extensively at the possibilities of passive dynamic running [14]. He found that, while some modes exhibit inherent stability, others need to be stabilized actively. Several controllers have been devised that stabilize passive dynamic running (e.g. [7] [15] [16]).

E. Approximate return map control

To control a running robot, it would be ideal to find an exact return map of the state of the robot. A return map would give the state at the next hop $\mathbf{v}_{i+1}$ as a function of the state at the current hop $\mathbf{v}_i$ and the angle of attack α_0:

$$\mathbf{v}_{i+1} = f\left(\mathbf{v}_i, \alpha_0\right). \tag{11}$$

To find such a return map, the equations of motion of the model of the robot would have to be integrated. Unfortunately, it is impossible to analytically integrate the equations of motion of even a very simple running model. This means that it is not possible to find an exact form of the return map. Many researchers have made approximations of the equations of motion of a simple running model (e.g. [17] [18] [19]), in order to find an approximate return map:

$$\mathbf{v}_{i+1} \approx \hat{f}\left(\mathbf{v}_i, \alpha_0\right). \tag{12}$$

Having an approximate return map, this equation can also be inverted to obtain the necessary angle of attack α_0 as a function of the current state $\mathbf{v}_i$ and the desired state $\mathbf{v}_{i+1,d}$. The inverted return map can then be used as an angle of attack controller.

F. Dead-beat control

One of the problems of controlling a running robot is that, even for the simplest running model, the equations of motion cannot be integrated analytically, and so no explicit solution to the equations of motion can be found. However, the equations of motion can be integrated numerically, which means that it is possible to calculate an ideal angle of attack, which gets the runner as close as possible to its desired state at the next step. A look-up table can then be created, containing the ideal angle of attack for many combinations of current and desired states.

Dead-beat control uses exactly such a look-up table. For computational reasons, this look-up table is constructed using the equations of motion of the SLIP model. Since the SLIP model is considered a good model for human and robot running [8], [9], it is expected that the data from this model will transfer reasonably well to a more realistic model and to a real robot. The look-up table contains all combinations of current states and angles of attack (in a certain range and with a certain grid size), and outputs the resulting state at the next step. Thus, for a given desired state at the next step, the ideal angle of attack can be found.

To the best of our knowledge, dead-beat angle of attack control has never been used as an angle of attack controller for a running robot. It will be investigated in-depth in this study.

G. Controller comparison

All controllers that were described in the previous sections are now compared on a basic level. This comparison is performed on the SLIP model. The model is started in a desired state of $y_{apex} = 1m$, $\dot{x}_{apex} = 5m/s$, and disturbed at apex. Only energy-neutral disturbances are applied. The resulting angles of attack of all controllers are compared to a numerically calculated ideal angle of attack, which returns the model exactly to the desired state at the next step. The results are shown in figure 6.

Figure 6 shows that two controllers give angles of attack very close to the ideal angle of attack, namely swing-leg retraction and approximate return map control. Constant control gives the ideal angle of attack only at the desired height of $1m$. The crude assumptions made by the Raibert controller result in angles quite far from the ideal angle of attack. Passive dynamic running on average gives angles of attack farthest from the ideal angle of attack. For the SLIP model, dead-beat control will always give angles equal to the ideal angle of attack, since its look-up table is constructed from the exact equations of motion of the model.

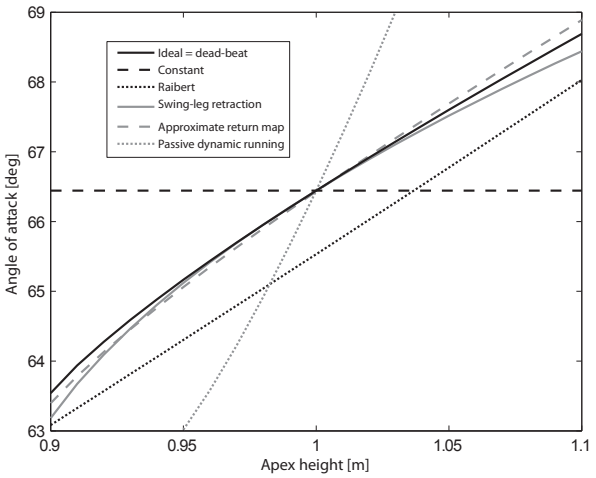


Figure 6: Comparison of the angles of attack given by the controllers to a numerically determined ideal angle of attack for the SLIP model. Controllers are started in a desired state and disturbed at apex. The desired height is $1m$. Angles given by dead-beat control are exactly equal to the ideal angle of attack, as it uses a look-up table constructed on the SLIP model.

From figure 6 it can be concluded that swing-leg retraction, approximate return map control and dead-beat control are promising controllers. However, approximate return map control requires the inversion of the approximate return map, see equation 12. Since this has to be done numerically, it cannot be done in real-time on a running robot, and thus a look-up table would have to be constructed. If a look-up table is constructed, it makes more sense to do this using the exact equations of motion, instead of an approximation of those. This is exactly what dead-beat control does. Therefore, only swing-leg retraction and dead-beat control are considered promising controllers, and they are investigated in-depth in this study. Constant angle of attack control is used in this study as a reference controller, as it is the most studied angle of attack controller.

IV. Controller implementation

In section III, two promising angle of attack controllers were identified, namely swing-leg retraction and dead-beat control. In this study we also investigate constant control as a reference controller, as it is the most studied angle of attack controller. All three controllers were discussed on a basic theoretical level. Implementation of these controllers on the SLIP model may be obvious at certain points, but implementation on the realistic model is not so obvious. This section will discuss how all three controllers are implemented on the two models and the robot used in this study.

A. Constant angle of attack control

For the SLIP model, the constant angle of attack necessary for a limit cycle with certain desired initial conditions can be calculated by numerically integrating the equations of motion. To find a limit cycle for the realistic model with certain desired initial conditions, the model is started with the angle of attack as found for the SLIP model. Often, this results in a limit cycle with conditions slightly off from the desired conditions, since the angle of attack value from the SLIP model is not exactly equal to the necessary angle of attack for the realistic model. The angle of attack is tweaked manually, together with the push-off torque value during the stance phase, until a limit cycle is found that has the desired apex height and forward speed.

B. Swing-leg retraction

In this study, swing-leg retraction is implemented with a constant retraction rate ω_R. The swing-leg starts at a certain retraction angle α_R at apex, and is retracted with the constant retraction rate from apex until touchdown. The resulting angle of attack is:

$$\alpha_0 = \alpha_R + \omega_R t, \tag{13}$$

where:

t is the flight time from apex until touchdown.

Karssen et al. [20] have shown that the optimal retraction rate is only dependent on the horizontal velocity of the runner. In this study the optimal retraction rate in terms of disturbance rejection as found by Karssen et al. will be used.

For the SLIP model, the retraction angle α_R and the retraction rate ω_R are chosen such that when the model is started in a certain desired limit cycle, the resulting angle of attack will keep the model in this limit cycle. If for a certain limit cycle the ideal angle of attack (which keeps the model exactly in the limit cycle), the desired retraction rate, and the apex height of the model are known, the retraction angle can be solved from the following formula:

$$y_{apex} = l_0 \sin(\alpha_0) + \frac{g}{2}\left(\frac{\alpha_0 - \alpha_R}{\omega_R}\right)^2 \tag{14}$$

For the realistic model, the retraction angle cannot be determined through a formula, since the hip does not follow a ballistic flight trajectory. To find a limit cycle for swing-leg retraction on the realistic model, the following method is used. The value for the angle of attack that was found for the constant controller is chosen as an initial estimate for the swing-leg retraction angle of attack. The retraction rate is determined based on the horizontal velocity, as mentioned in [20]. The flight time from apex to touchdown is determined from the limit cycle that was found for the constant angle of attack limit controller. An initial estimate for the retraction angle is then calculated with the following formula:

$$\alpha_R = \alpha_{0,constant} - \omega_R t_{constant} \tag{15}$$

where:

$\alpha_{0,constant}$ is the angle of attack in the constant angle of attack limit cycle and

$t_{constant}$ is the flight time from apex until touchdown for the constant angle of attack limit cycle.

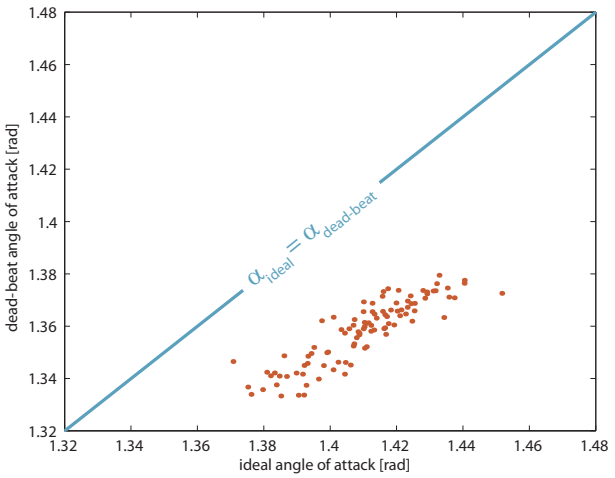


Figure 7: Comparison of dead-beat controller angles of attack to ideal angles of attack for realistic model. For 100 limit cycles near a desired limit cycle the ideal angle of attack was calculated and is compared to the angle of attack given by CoM dead-beat control. The line $\alpha_{ideal} = \alpha_{dead-beat}$ is also indicated.

With these control parameters, and initial conditions as found in the constant angle of attack limit cycle, the model is run for several steps until it converges to a limit cycle.

C. Dead-beat control

Dead-beat control uses a look-up table that has both the apex height at the next hop as well as the apex forward speed at the next hop as outcomes. Since these state parameters are coupled for the SLIP model, only one of the two can be used as desired state. The choice is made here to use forward speed as the leading desired state, with the side note that the height at the next step should not go below the rest length of the leg. This will ensure that controller is not limited in the choices of angles of attack due to limited ground clearance.

To use dead-beat control on the realistic model, an assessment needs to be made on how to interpret the SLIP model. For the SLIP model, the location of the Center of Mass (CoM) of the entire model and the location of the hip coincide. For the realistic model, this is not the case. This begs the question whether the dead-beat control should use values for the CoM apex height and speed, or for the hip apex height and speed. This question is investigated in-depth in appendix A. It is shown that hip dead-beat control gives angles of attack slightly closer to the ideal angle of attack than CoM dead-beat control for the realistic model. However, practical considerations, such as the availability and accuracy of certain sensors, will often be paramount in determining which type of control is used. For this study, the values for the CoM are used.

The dead-beat control look-up table is constructed using the SLIP-model equations of motion. This look-up table will also be used for the realistic model and the real robot. Although the SLIP-model is a good model for human and robot running [8], it is unknown whether the ideal angles of attack stored in the SLIP-model look-up table will transfer well to the realistic model. To investigate this, 100 combinations of initial conditions are taken. The initial conditions are all within 10% of the initial conditions a certain desired limit cycle. For all 100 combinations of initial conditions, the ideal angle of attack is calculated. This is the angle of attack for which the conditions after one step are the closest to its initial conditions. Also, the angle of attack that is given by the CoM dead-beat controller is calculated. Figure 7 shows the results. In this figure it can be seen that CoM dead-beat control consistently gives angles of attack slightly lower than the ideal angle of attack. However, the average difference is about $0.05rad$, or about 3 degrees. Differences this small indicate that the ideal angles of attack found on the SLIP model transfer remarkably well to the realistic model. It is thus expected that the performance of the dead-beat controller on the realistic model will resemble its performance on the SLIP model. This is investigated further in the next sections.

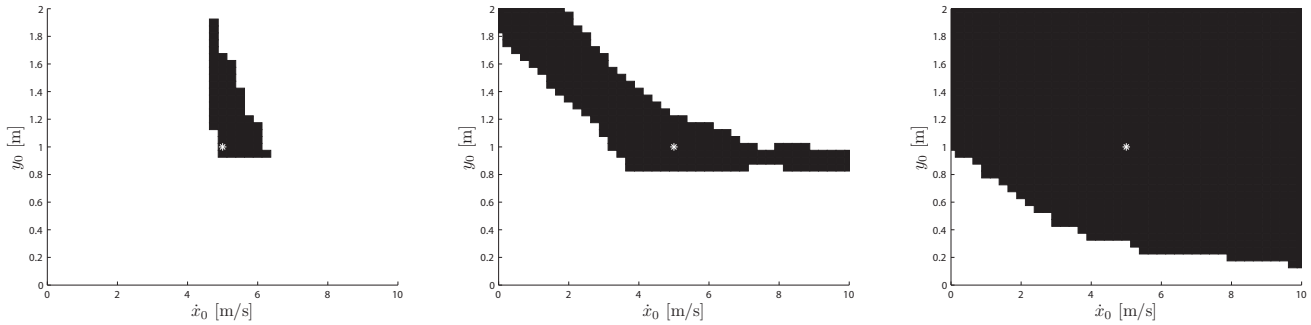
V. Disturbance Rejection

The main comparison method that will be used in this study is disturbance rejection. This method shows how well a robot can deal with unexpected disturbances. It is often used in robotics as a performance measure, and resembles real-life situations, where the robot is running along in its limit cycle and is suddenly disturbed. Such a disturbance can be caused by many factors, such as uneven terrain or wind factors when running outside. First, the size of the disturbances that the controllers can handle will be investigated, by looking at the Basin of Attraction for all controllers on the SLIP model (section V-A), and the maximum allowable disturbance for all controllers on the realistic model (section V-B). Then, the response of the controllers to disturbances is investigated in section V-C.

A. Basin of Attraction

For the SLIP model, a Basin of Attraction (BoA) is created. A BoA shows for all combinations of initial conditions, whether the model keeps running for a certain number of steps or falls down. In this study, a threshold of 24 steps is used. The SLIP model is considered to have fallen when the point mass is below the ground. Desired conditions are kept at the same level throughout the test, since it is assumed that the model is running in its desired limit cycle before it is disturbed, and will want to return to this desired limit cycle regardless of the disturbance. The SLIP model will be disturbed at apex, which means that only disturbances in apex height and forward speed need to be taken into account, as the state of the SLIP model is completely determined by height and forward speed at apex. The desired apex state is set at $\dot{x}_{apex,des} = 5m/s$ and $y_{apex,des} = 1m$.

Figure 8 shows the resulting basins of attraction for all three controllers on the SLIP model. Figure 8a shows that, as expected, constant angle of attack control has a small BoA around the desired state of $\dot{x}_{apex,des} = 5m/s$ and $y_{apex,des} = 1m$. Figure 8b shows that swing-leg retraction already has a much larger BoA than constant angle of attack control. Finally, figure 8c shows that dead-beat control has by far the largest BoA. The BoA of dead-beat control covers



(a) Constant control (b) Swing-leg retraction (c) Dead-beat control

Figure 8: Basin of attraction for all controllers on SLIP model. Model parameters $m = 80kg, k = 20 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$. The white star indicates the desired state.

almost the entire investigated region. The initial states that are not part of the dead-beat BoA all have apex heights lower than 1m. If the initial apex height is lower than the rest length of the leg, the ground clearance of the model limits the choices for the angle of attack. These limits on the possible angles of attack may cause the model to fall, regardless of the controller.

For the realistic model, it is computationally infeasible to create a real BoA, since at any moment during the cycle its state is defined by at least nine variables.

B. Maximum allowable disturbance

Since it is computationally infeasible to create a BoA for the realistic model, the maximum disturbance is calculated for this model, which takes into account disturbances in one direction only. Furthermore, only disturbances in apex hip height and hip forward speed are investigated, since these can be compared to the SLIP model results. To make comparison to the SLIP model disturbance rejection even easier, the maximum disturbances of the SLIP model are distilled from the BoA. These maximum disturbances are simply horizontal and vertical slices around the desired state of the BoA shown in figure 8. Maximum disturbance for all three controllers on the SLIP model can be seen in figure 9.

For the realistic model, the runner is considered to have fallen when the hip is below the ground. If the swing foot is below the ground, the model is not considered to have fallen, for this can be solved by changing controller parameters of control parts that are independent on the angle of attack controller. However, the results will note if the swing-foot was below the ground during a certain running cycle. The desired state for the realistic model is not set at the same (scaled) value as for the SLIP model, as running robot Phides is not designed to run at these high speeds. The desired state for the realistic model is set at $y_{apex,des} = 0.58m$ (equal to the rest length of the leg) and $\dot{x}_{apex,des} = 1.5m/s$. The maximum disturbances for all three controllers on the realistic model can be seen in figure 10.

Although figures 9 and 10 are made for different models and different desired states, they show roughly the same results for most controllers. Most importantly, they both show that dead-beat control always outperforms or equals the other controllers, regardless of the type of disturbance applied. This confirms the results already found on the SLIP model that dead-beat control has the best disturbance rejection of the angle of attack controllers.

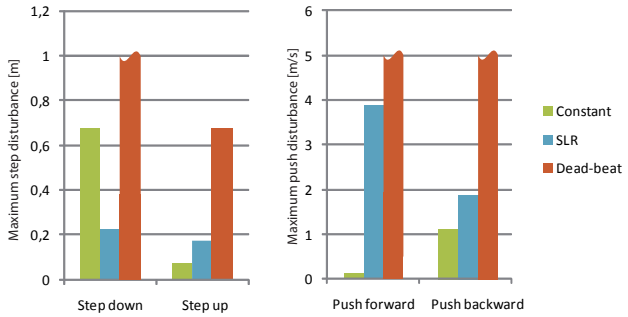


Figure 9: Maximum disturbance for controllers on the SLIP model. These results are distilled from the Basin of Attraction of the SLIP model of figure 8, around the desired states of $\dot{x}_{apex,des} = 5m/s$ and $y_{apex,des} = 1m$. Waves at the end of the bars indicate that the maximum disturbance may be even higher, but is not calculated further for computational reasons. Model parameters $m = 80kg, k = 20 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$

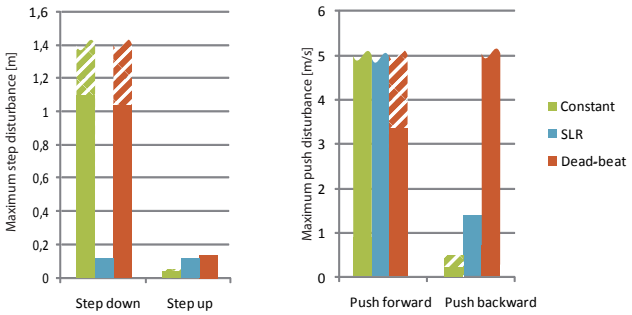


Figure 10: Maximum disturbance for controllers on the realistic model. Waves at the end of the bars indicate that the maximum disturbance may be even higher, but is not calculated further for computational reasons. Stripes in the bar indicate that at these disturbances, the swing-foot is below the ground at some point during the running cycle. The desired state is $\dot{x}_{apex,des} = 1.5m/s$ and $y_{apex,des} = 0.58m$. For model parameters see table I.

Figures 9 and 10 further show that swing-leg retraction cannot handle step disturbances very well, but can handle push disturbances very well, especially forward pushes. Dead-beat control deals very well with step down disturbances, and not so well with step up disturbances. The last is true for all controllers, due to limited ground clearance. There is a slight discrepancy in the results for the SLIP model and the realistic model for the constant controller. Both models show that constant control cannot deal well with step down and backward push disturbances. However, the realistic model indicates that constant control can handle step down and forward push disturbances better than expected by the SLIP model results. An explanation for this discrepancy may be found in the coupling of apex height and speed for the SLIP model, which is no longer present in the realistic model.

C. Disturbance respons

The previous sections showed, for both models, the maximum disturbance the controllers can handle. However, the maximum disturbance does not give any information as to the model's response to the disturbance; how fast does the model return to a limit cycle, and which limit cycle does it return to? To show the disturbance response, a variation of the gait sensitivity norm [21] is used in this study. Instead of summing the gait indicators over the steps of interest, as in [21], this study uses plots of the gait indicators over time. The gait indicator used for both the SLIP model and the realistic model is step time. The models are initialized in their desired limit cycles. After 10 steps, the apex height of the models is disturbed. The forward speed of the models at apex is changed accordingly, so that the overall energy of the models stays the same. The applied disturbance is the same for all controllers and is as large as possible, keeping in mind that all controllers should be able to keep running after the disturbance is applied. For the SLIP-model this means that the apex height is increased by 10%; for the realistic model the apex height is increased by 5%.

Figure 11 shows the disturbance response of the controllers on the SLIP model. In this figure it can be seen that for dead-beat control, the step time increases the first step after the disturbance, but is back to the limit cycle level the second step after the disturbance. For swing-leg retraction, the settling-time is a bit longer, although the initial increase in step time is a bit lower. Constant control has by far the worst disturbance response of the three controllers. It has an enormous increase in step time right after the disturbance, and it takes about 30 steps before the step time is back to its initial level.

Figure 12 shows the disturbance response of all controllers on the realistic model. Since all models have slightly different initial limit cycles, the initial step time levels are also different. Dead-beat control shows somewhat similar behavior on the realistic model as it did on the SLIP model. Immediately after the disturbance, the step time is increased. However, it takes dead-beat control a few more steps to settle back into its original step time on the realistic model, whereas for the SLIP model this happened already the second step after the disturbance. This is explained by the fact that dead-beat control

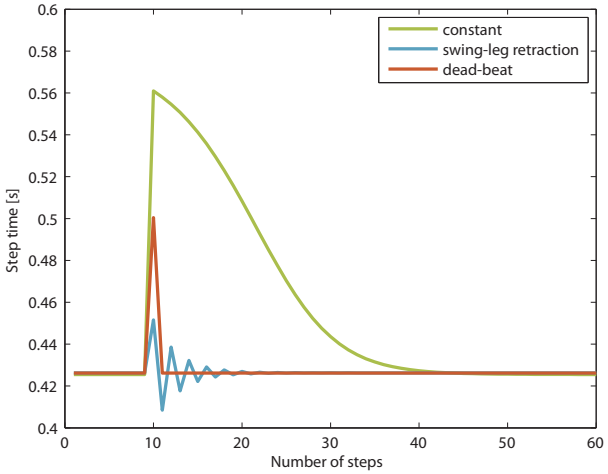


Figure 11: Disturbance response of controllers on SLIP model. After 10 steps, the controller is disturbed, in such a way that the overall energy is kept constant. Model parameters $m = 80kg, k = 20 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$. Desired state $\dot{x}_{apex,des} = 5m/s$ and $y_{apex,des} = 1m$.

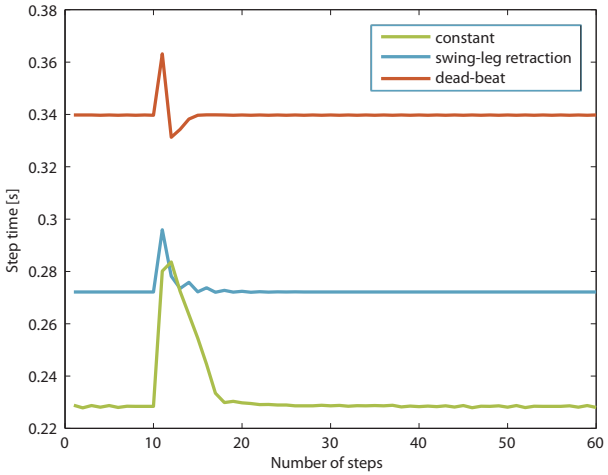
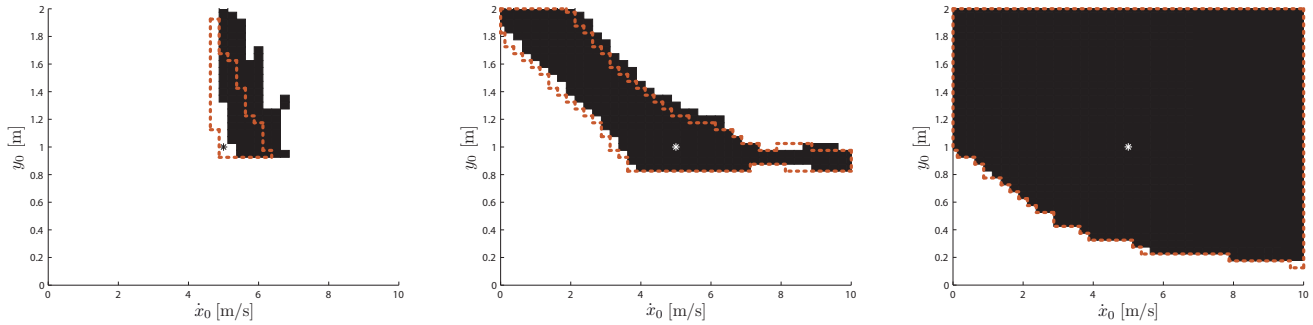


Figure 12: Disturbance response of controllers on the realistic model. After 10 steps, the controller is disturbed, in such a way that the overall energy is kept constant.

does not give *exactly* the ideal angle of attack for the realistic model, unlike for the SLIP model. Swing-leg retraction also shows a similar disturbance response for both models. The step time is increased immediately after the disturbance, and it takes about 10 steps for the step time to return to its original level. Finally, constant control settles down to its original step time a bit faster on the realistic model than on the SLIP model, but it still has the worst disturbance response of all controllers. The initial increase in step time right after the disturbance is much larger than for the other two controllers.

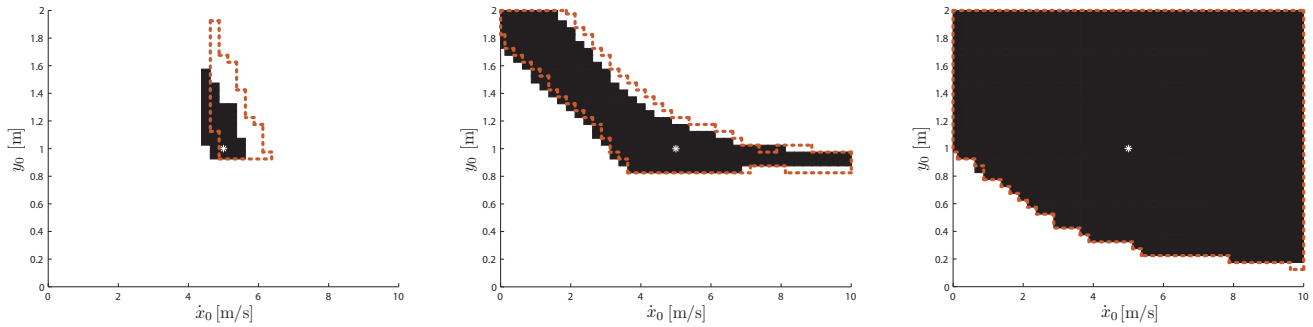
VI. Robustness

In the previous sections the angle of attack controllers were compared in terms of disturbance rejection. It was assumed that all model parameters and controller inputs were measured



(a) Robustness constant (b) Robustness SLR (c) Robustness dead-beat control

Figure 13: Robustness of controllers on SLIP model. Model parameters $m = 80kg, k = 20 * 10^3 N/m, L0 = 1m, g = 9.81m/s^2$; Controller input parameter k is estimated falsely at $k = 18.18 * 10^3 N/m$. Dotted lines indicate the original basins of attraction. The white star indicates the desired state.



(a) Robustness constant (b) Robustness SLR (c) Robustness dead-beat control

Figure 14: Robustness of controllers on SLIP model. Model parameters $m = 80kg, k = 20 * 10^3 N/m, L0 = 1m, g = 9.81m/s^2$; Controller input parameter k is estimated falsely at $k = 22 * 10^3 N/m$. Dotted lines indicate the original basins of attraction. The white star indicates the desired state.

correctly, whereas in real life model errors and sensor errors are likely to occur. Therefore, this section investigates the robustness of the controllers. Robustness indicates how well a controller can handle model and sensor errors. For computational reasons, robustness of the controllers is examined only on the SLIP model.

A. Model errors

To test the effect of model errors, the Basin of Attraction (BoA) of figure 8 is recreated, this time with an error in the model parameters. Out of the four model parameters of the SLIP model (m, k, l_0 and g), only one has to be varied, because the equations of motion are scalable according to Buckingham's PI theorem [22]. The spring stiffness that is used by the controllers will be varied, while the model spring stiffness that is used in the equations of motion will be kept the same. This way, any changes in the BoA will be attributable to the model error only, and not to actual changes in the model parameters. Effects of a 10% underestimation of the spring stiffness can be found in figure 13. Effects of a 10% overestimation of the spring stiffness can be found in figure 14.

Figure 13a shows that for constant angle of attack control, an underestimation of the spring stiffness will result in a slightly shifted and slightly larger BoA, where the desired state of $\dot{x}_{apex,des} = 5m/s$ and $y_{apex,des} = 1m$ is not even part of the BoA anymore. Figure 14a shows that an overestimation of spring stiffness also results in a slightly shifted, but smaller BoA. The shift can be explained by the fact that an error in the model parameters will result in a wrong estimation of the constant angle of attack needed to stay in the desired limit cycle. Figure 13b and figure 14b show that swing-leg retraction is hardly affected by underestimation or overestimation of the spring stiffness. The same goes for dead-beat control; although there is a slight change along the edges of the BoA, the general shape and size of the BoA stay the same.

B. Sensor errors

There are three different sensors that are relevant to the different controllers. Constant angle of attack control needs no sensors, and is therefore not affected by sensor errors. Swing-leg retraction needs to know when the apex occurs, which is the point at which the retraction of the swing-leg starts. Thus it is affected by apex time errors. Dead-beat control is affected by errors in apex state, so both apex height and apex forward speed.

The effect of sensory errors on the controllers is investigated as follows: A BoA is created, but at every step the relevant

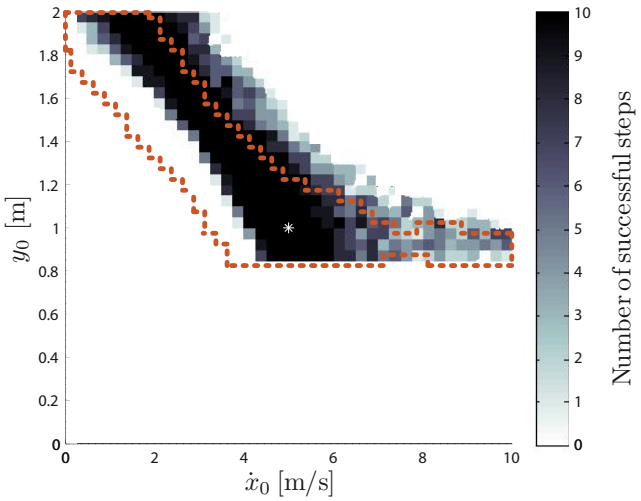


Figure 15: Robustness of swing-leg retraction controller on the SLIP model with apex time sensor errors plus/min 5%. Other controllers are not affected by this type of error. The dotted line indicates the original basin of attraction. The white star indicates the desired state.

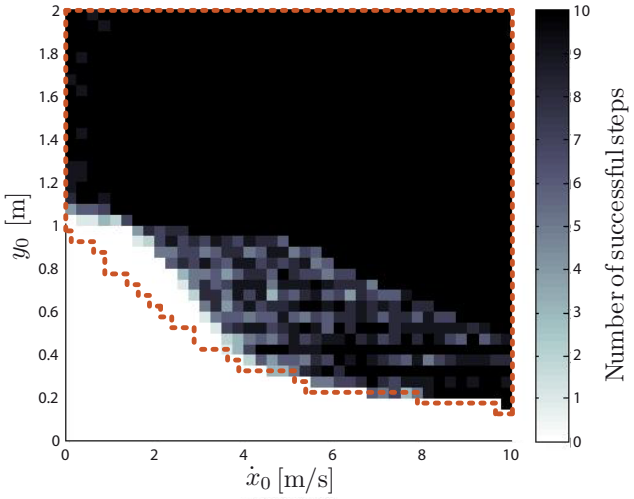


Figure 16: Robustness of dead-beat controller on the SLIP model with apex height sensor errors plus/min 5%. Other controllers are not affected by this type of error. The dotted line indicates the original basin of attraction. The white star indicates the desired state.

sensor information is varied randomly, with a uniform distribution between plus and minus 5% of the actual value. For apex time, the value that is varied is the time between liftoff and apex. This process is repeated ten times for each sensor, and the 10 resulting BoAs are summed.

First, apex time sensory errors are investigated. The only controller affected by this is swing-leg retraction, since the other controllers do not need to know the time at which apex occurs. The results for apex time errors can be found in figure 15. Although the resulting BoA for swing-leg retraction becomes somewhat smaller due to the sensory errors, the general shape stays the same, and swing-leg retraction is found not to be affected too much by this type of error.

Results for apex height errors can be found in figure 16. The only controller affected by this type of error is dead-beat control. As can be seen in figure 16, dead-beat control is affected by apex height sensory errors when the starting height is lower than 1m. For starting heights lower than 1m, the dead-beat controller may select an angle of attack that places the foot below the ground at apex. This happens when the measured height is more than the actual height and will automatically result in a fall. This does not happen for starting heights over $1m$ high, since the leg length is only $1m$, and the foot can thus never be placed below the ground.

Finally, apex forward speed errors are considered. The results can be found in figure 17. Again, only dead-beat control is considered, since the other controllers do not use apex forward speed as an input, and are thus not affected by these errors. The figure shows that dead-beat control is hardly affected by apex forward speed errors.

Apart from the random sensory errors that were discussed here, persistent sensor errors can also occur. Persistent sensor error effects on the controllers show the same tendency as given here for random sensor errors. Extensive results for persistent sensor errors can be found in appendix B.

For all three controllers it can be concluded that they are

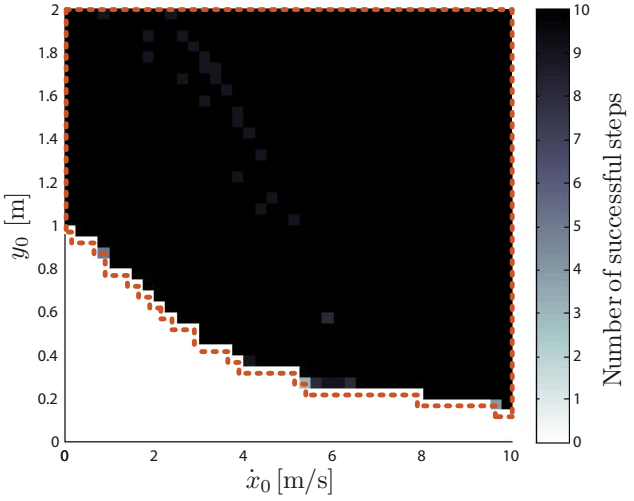


Figure 17: Robustness of dead-beat controller on the SLIP model with apex forward speed sensor errors plus/min 5%. Other controllers are not affected by this type of error. The dotted line indicates the original basin of attraction. The white star indicates the desired state.

very robust. Although the runner may end up in a slightly different limit cycle than desired, for almost all of the points in the initial BoA it will still keep on running with errors present. The only type of error that has a significant effect is the apex height error, which reduces the BoA of dead-beat control for apex heights lower than the rest length of the leg.

VII. Parameter Study

In the previous sections, several angle of attack controllers were tested on disturbance rejection and robustness, for certain parameter sets. This section will investigate whether the conclusions that can be drawn from this are still valid for other parameter sets. This parameter study is performed on the SLIP model. Again, only one of the four model parameters

has to be varied because the equations of motion are scalable. Furthermore, the desired forward speed is also varied.

First the model parameter spring stiffness will be varied. In the previously constructed Basins of Attraction(BoAs) the spring stiffness was set at $k = 20 * 10^3 N/m$; here a spring stiffness of $k = 10 * 10^3 N/m$ and $k = 40 * 10^3 N/m$ is used. Figure 18 shows the resulting BoAs for a spring stiffness of $k = 10 * 10^3 N/m$. Figure 19 shows the resulting BoAs for a spring stiffness of $k = 40 * 10^3 N/m$. Although there are slight differences between these BoAs and the BoAs with a spring stiffness of $k = 20 * 10^3 N/m$, the general size and shape of the BoAs stay the same regardless of the parameter change.

The second parameter that is changed is the desired forward speed at apex. Previously the desired forward speed was set at $5m/s$; here it will be set at $2.5m/s$ and $7.5m/s$.

From figures 20 and 21 it can be seen that changing the desired forward speed has a slightly larger effect than changing the spring stiffness. This is especially evident for the constant controller, which is not even able to run with desired speeds of $2.5m/s$. Swing-leg retraction also has a smaller BoA at this desired speed. However, the trends that were found in section V-A are still valid: Constant angle of attack control has the smallest BoA; Swing-leg retraction performs better than constant control in terms of disturbance rejection, and dead-beat control has by far the largest BoA, and thus the best disturbance rejection. Therefore, it can be concluded that results are independent of parameter values.

VIII. Running robot experiments

Both dead-beat control and swing-leg retraction are implemented on running robot Phides. However, the robot is not yet able to run with these controllers due to problems with sensors and motors. The robot is able to hop with a constant angle of attack controller, given the right initial conditions. Results from the constant angle of attack controller show one of the problems that occur for the other controllers. The main problem is that the motors do not provide accurate angle of attack control. Figure 22 shows the actual and desired angle of attack for 16 consecutive steps of the robot. It can be seen in this figure that actual angles of attack are sometimes quite far from the desired angle of attack, up to 0.08 rad. This problem is especially evident for the left leg of the robot. Possible causes of this problem are the fact that while running the robot operates near the actuator limits, and that the measured torques provided by the motors is often not equal to the actual torques that are applied.

We plan to resolve these problems in the near future, so that both dead-beat control and swing-leg retraction can be tested on the robot.

IX. Discussion

In this paper two promising angle of attack controllers have been compared in terms of disturbance rejection and robustness. On the SLIP model, the Basin of Attraction (BoA) of the dead-beat control was far larger than that of swing-leg retraction. These findings are corroborated by the realistic

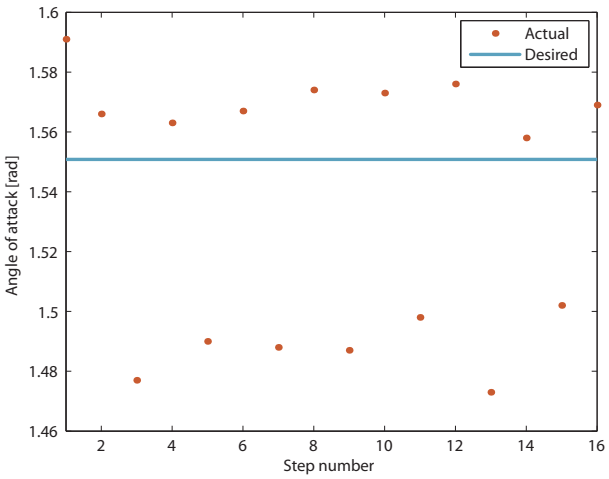
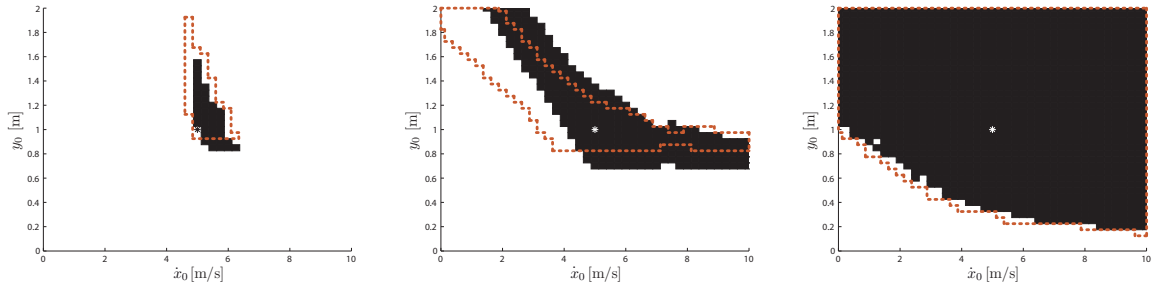


Figure 22: Actual and desired angle of attack for running cycle on robot Phides with constant angle of attack controller. Uneven step numbers correspond to the stance phase of the left leg; Even step numbers correspond to the stance phase of the right leg.

model results, where dead-beat control is able to withstand very large disturbances, except for step up disturbances. Step up disturbance rejection is limited for any controller due to limited ground clearance. Swing-leg retraction showed a much smaller BoA on the SLIP model than dead-beat control. Again, the trends found for the SLIP model were confirmed by results of the realistic model. These showed that swing-leg retraction cannot handle very large step disturbances - both positive and negative - but can handle push disturbances quite well. Constant control was used as a reference, since this is a very simple and much studied form of angle of attack control. Indeed, the BoA of constant control found for the SLIP model was quite small. Surprisingly, the disturbance rejection of constant angle of attack control on the realistic model was larger than for the SLIP model, for step down and forward push disturbances. This may be explained by the decoupling of forward speed and height at apex for the realistic model, due to the possibilities of energy dissipation. This allows the realistic model to end up in a limit cycle with a different energy level, which is impossible for the SLIP model.

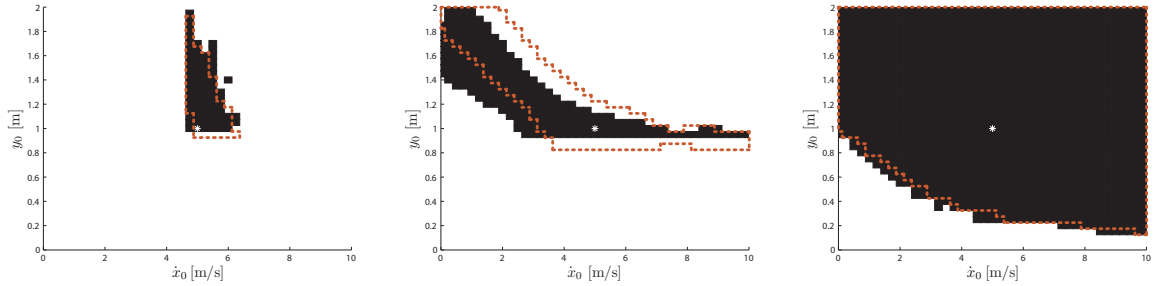
It was shown in this paper that dead-beat control performs very well on the realistic model, even though the angles of attack it uses are calculated on the SLIP model. However, for a certain parameter set, it was shown that dead-beat control consistently gives angles of attack lower than the ideal angle of attack for the realistic model. It may be possible to correct for this slight difference by adapting the angle of attack given by dead-beat control. This is not done in this study, so that the intrinsic properties of dead-beat control could be better investigated. Further research is needed to determine whether such a correction is possible and how the size of the correction depends on certain parameters.

Results in this study show that dead-beat angle of attack control performs better in terms of disturbance rejection than swing-leg retraction. This begs the question why humans



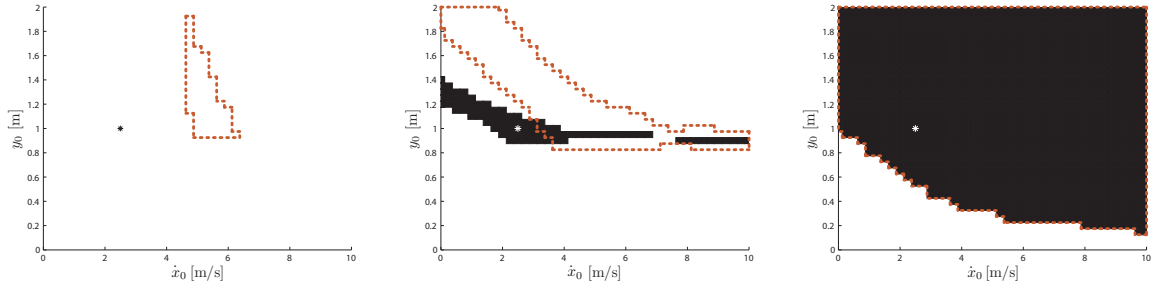
(a) Constant control (b) Swing-leg retraction (c) Dead-beat control

Figure 18: Basin of attraction for all controllers on SLIP model. Model parameters $m = 80kg, k = 10 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$. The dotted line indicates the original basin of attraction. The white star indicates the desired state.



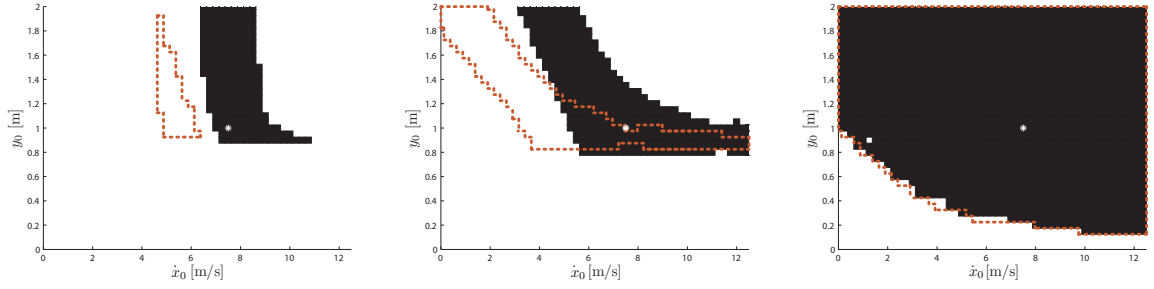
(a) Constant control (b) Swing-leg retraction (c) Dead-beat control

Figure 19: Basin of attraction for all controllers on SLIP model. Model parameters $m = 80kg, k = 40 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$. The dotted line indicates the original basin of attraction. The white star indicates the desired state.



(a) Constant control (b) Swing-leg retraction (c) Dead-beat control

Figure 20: Basin of attraction for all controllers on SLIP model. Model parameters $m = 80kg, k = 20 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$, desired forward speed $\dot{x}_{apex,des} = 2.5m/s$. The dotted line indicates the original basin of attraction. The white star indicates the desired state.



(a) Constant control (b) Swing-leg retraction (c) Dead-beat control

Figure 21: Basin of attraction for all controllers on SLIP model. Model parameters $m = 80kg, k = 20 * 10^3 N/m, l0 = 1m, g = 9.81m/s^2$, desired forward speed $\dot{x}_{apex,des} = 7.5m/s$. The dotted line indicates the original basin of attraction. The white star indicates the desired state.

and animals do retract their legs prior to touchdown when they run, as can be seen in the photographs of Muybridge [3]. The answer may be found in the fact that swing-leg retraction reduces the foot speed relative to the ground at touchdown. As explained by Karssen et al. [20], this has three benefits. First of all, it reduces impact losses and thus reduces the specific energy costs of running. Secondly, it reduces the chance of slipping. Finally, it reduces the impact forces at touchdown, thereby reducing the chance of damage to the runner. These benefits were not taken into account in this study, and may explain the running behavior of humans.

In this paper, swing-leg retraction and dead-beat control were investigated as two separate angle of attack controllers. However, it is also possible to combine the two types of control. This will result in a controller where the desired angle of attack is found in a lookup table, as for dead-beat control, but instead of keeping the leg at this angle constantly until touchdown, the leg is swung backwards around this angle of attack as happens in swing-leg retraction. This way, the angle of attack is still chosen based on the dead-beat control that has been proven to work well, but will also have the benefits of swing-leg retraction: it will compensate for height sensor errors and will reduce impact losses and forces as well as the likelihood of slipping. All the tests performed on dead-beat control and swing-leg retraction in this study have also been performed on the combined control, for full results see appendix C. On the SLIP model, in most of the cases, combined control performs exactly equal to dead-beat control, as is expected when no errors are present. When sensor height errors are present, combined control does indeed improve the disturbance rejection compared to dead-beat control. More prominent improvements were expected for the realistic model. However, the realistic model also shows the largest flaw of combined control. For the combined control to work, the exact moment of touchdown has to be known, so that the leg indeed swings around the desired angle of attack at that moment. If the touchdown time is estimated wrongly, the high retraction rates will result in angles of attack very far off the desired angle of attack. Unfortunately, there is no easy way to determine the exact moment of touchdown for the realistic model. Therefore, it was impossible to find a limit cycle using combined control on the realistic model around the desired state. Unless a better way is found to determine the touchdown time, a combination of dead-beat control and swing-leg retraction will not perform well as an angle of attack controller.

Finally, a note should be made here as to one of the initial assumptions made in this study, namely the decoupling of the three controllers for a running robot: the angle of attack controller, the energy controller (push-off during the stance phase), and the upper body posture controller. Although almost all studies on running robotics use this assumption, the coupling of these controllers has never been properly investigated. For the SLIP model, only angle of attack control is relevant, since the energy level is constant, and the upper body is a point mass. For the realistic model, this study used a fixed upper body, eliminating the need for an upper body posture controller, and used the simplest form of energy control. Other studies are ongoing at the Delft Biorobotics Laboratory that study the other controllers needed for a running robot; when these are finished it is recommended that the effects that these controllers have on one-another is studied as well, before assuming that what is best if only of the controllers is considered is also best when complete control is considered.

X. Conclusions

In this paper two promising angle of attack controllers were compared in terms of disturbance rejection, stability and robustness. From this study, it can be concluded that:

- Dead-beat control outperforms swing-leg retraction in terms of disturbance rejection and robustness.
- Both dead-beat control and swing-leg retraction can deal well with push disturbances. Dead-beat control can handle step disturbances a lot better than swing-leg retraction.
- Dead-beat control is hardly affected by model and sensory errors.
- The ideal angles of attack that are used by dead-beat control are calculated on a simple model, but transfer well to a more realistic model.

References

[1] R. Alexander. Three uses for springs in legged locomotion. *The International Journal of Robotics Research*, 9(2):53, 1990.

[2] P.G. Weyand, M.W. Bundle, C.P. McGowan, A. Grabowski, M.B. Brown, R. Kram, and H. Herr. The fastest runner on artificial legs: different limbs, similar function? *Journal of Applied Physiology*, 107(3):903, 2009.

[3] E. Muybridge. *The human figure in motion*. Dover Publications, 1955.

[4] M.H. Raibert. *Legged robots that balance*. MIT press Cambridge, MA, 1986.

[5] A. Seyfarth, H. Geyer, M. Gunther, and R. Blickhan. A movement criterion for running. *Journal of Biomechanics*, 35(5):649–655, 2002.

[6] A. Seyfarth, H. Geyer, and H. Herr. Swing-leg retraction: a simple control model for stable running. *Journal of Experimental Biology*, 206(15):2547, 2003.

[7] M. Ahmadi and M. Buehler. A control strategy for stable passive running. In *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings*, volume 3, 1995.

[8] R. Blickhan. The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11-12):1217–1227, 1989.

[9] R. Blickhan and RJ Full. Similarity in multilegged locomotion: Bouncing like a monopode. *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology*, 173(5):509–517, 1993.

[10] D.G.E. Hobbelen and M. Wisse. Limit cycle walking. *Humanoid Robots*, pages 451–468, 2007.

[11] G.A. Pratt and M.M. Williamson. Series elastic actuators. In *Proceedings of IROS 95,*, 1995.

[12] B. De Wit, D. De Clercq, and P. Aerts. Biomechanical analysis of the stance phase during barefoot and shod running. *Journal of Biomechanics*, 33(3):269–278, 2000.

[13] T. McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9(2):62, 1990.

[14] T. McGeer. Passive bipedal running. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 240(1297):107–134, 1990.

[15] C. Francois and C. Samson. A new approach to the control of the planar one-legged hopper. *The International Journal of Robotics Research*, 17(11):1150, 1998.

[16] SH Hyon and T. Emura. Energy-preserving control of a passive one-legged running robot. *Advanced Robotics*, 18(4):357–381, 2004.

[17] WJ Schwind and DE Koditschek. Approximating the stance map of a 2-DOF monoped runner. *Journal of Nonlinear Science*, 10(5):533–568, 2000.

[18] H. Geyer, A. Seyfarth, and R. Blickhan. Spring-mass running: simple approximate solution and application to gait stability. *Journal of theoretical biology*, 232(3):315–328, 2005.

[19] O. Arslan, U. Saranli, and O. Morgul. An approximate stance map of the spring mass hopper with gravityorrection for nonsymmetric locomotions. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1829–1834. Institute of Electrical and Electronics Engineers Inc., The, 2009.

[20] J.G.D. Karssen, M. Haberland, M. Wisse, and S. Kim. The optimal swing-leg retraction rate for running. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011.

[21] D.G.E. Hobbelen and M. Wisse. A disturbance rejection measure for limit cycle walkers: The gait sensitivity norm. *Robotics, IEEE Transactions on*, 23(6):1213–1224, 2007.

[22] E. Buckingham. On physically similar systems; illustrations of the use of dimensional equations. *Physical Review*, 4(4):345–376, 1914.

APPENDIX A
COMPARISON OF CENTER OF MASS OR HIP VALUES FOR DEAD-BEAT CONTROL

In this study, dead-beat control is proposed as an angle of attack controller. Using the equations of motion of the SLIP model, a look-up table is constructed that contains an ideal angle of attack for all combinations of current and desired states (apex height and forward speed). However, using this look-up table on the realistic model, it is unknown whether values for the hip apex height and forward speed, or for the Center of Mass (CoM) apex height and forward speed should be used.

To investigate this, the following method is used. First, a limit cycle is found for the realistic model with a constant angle of attack controller. Then, the initial conditions of this limit cycle are varied randomly, so that all initial conditions are within +/-10% of their limit cycle value. For the new initial conditions, an ideal angle of attack is calculated which gets the model as close as possible to its initial conditions after one step. Close is defined here as having a small sum of all differences between conditions after one step and initial conditions, so:

$$e = \sum_{i=1}^{12} abs\left(V_{n+1}(i) - V_n(i)\right) \tag{16}$$

where:

e is the error that is to be minimized,
V is the initial conditions vector and
n is the step number.

For the same initial conditions, the angle of attack given by the dead-beat controller is also calculated, both using values of the CoM and using values of the hip. This process is repeated 100 times. Figure 23 shows a boxplot of the difference between the dead-beat angle of attack and the ideal angle of attack, for both options.

From figure 23 it can be seen that using values of the hip to look up an angle of attack performs slightly better, that is it gives angles of attack closer to the ideal angle of attack. There are also some practical considerations however when choosing for CoM or hip dead-beat control. For the CoM type of dead-beat control, the length from Center of Mass to foot at touchdown is needed. Theoretically, it is impossible to know this length before touchdown, as the location of the Center of Mass is not yet known. However, when looking at the 100 trials as used before, it turns out that the length between CoM and foot at touchdown is almost a constant value. For the 100 trials used before, the CoM-foot length has an average value of 0.49m with a standard deviation of 0.001m. This is thus chosen as the standard value of the CoM-foot length.

For the hip type of dead-beat control, a practical consideration is that at liftoff the apex height and speed of the hip is unknown, since the hip does not follow a ballistic flight trajectory. This means that the desired angle of attack cannot be calculated yet, and has to be estimated, so that the leg can already be moved near its desired angle. If this is not done until apex, there may not be enough time to move the leg to its desired angle before touchdown. A first option to estimate the desired angle of attack is to estimate the apex height and speed of the hip by assuming that it *does* follow a ballistic flight trajectory. Using this on the 100 trajectories as calculated before, it is found that this gives an error in apex height error of only 2%, but an apex speed error of 52%. This will not give a realistic estimate of the angle of attack, so instead another estimate is used. Another possibility is to use the angle of attack that was found at the previous apex as an estimate for the angle of attack between liftoff and apex. For conditions close to a limit cycle, this will be a very good estimate; for conditions further from a limit cycle this may also be a bit further off the desired angle of attack, but is close enough that after apex there is enough time to move the leg to the desired angle of attack.

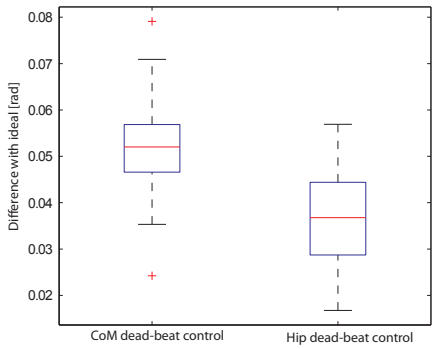


Figure 23: Boxplot of the difference between the ideal angle of attack and the angles of attack given by the two types of dead-beat control, for 100 different sets of initial conditions. On each box, the middle line indicates the median, and the two outer lines indicate the 25th and 75th percentiles. The whiskers indicate the farthest points that are not considered outliers; outliers are indicated by crosses.

Since the hip apex height and speed can not be determined at liftoff, an apex sensor is needed for the hip dead-beat control. This is another practical consideration, since a sensor that can accurately measure apex time may not be present in all robots. For the CoM dead-beat control this is not a problem, since the center of mass does follow a ballistic flight trajectory during the flight phase. This means that CoM positions and velocities at liftoff can be used to calculate the CoM apex height and forward speed. Measuring liftoff is usually easier than measuring apex, since ground contact sensors are usually present in running robots. The absence of an accurate way to measure apex is the reason why CoM dead-beat control was chosen in this study.

Finally, a practical consideration to choose hip dead-beat control is the amount of calculations and inputs needed for CoM dead-beat control. To calculate the position and velocity of the center of mass, the positions and velocities of all state parameters have to be measured. The large amount of inputs needed may lead to larger errors in the CoM apex height and forward velocity, which will in turn result in a larger error in the ideal angle of attack. This is not the case for hip dead-beat control, which only needs to know the hip height and forward speed.

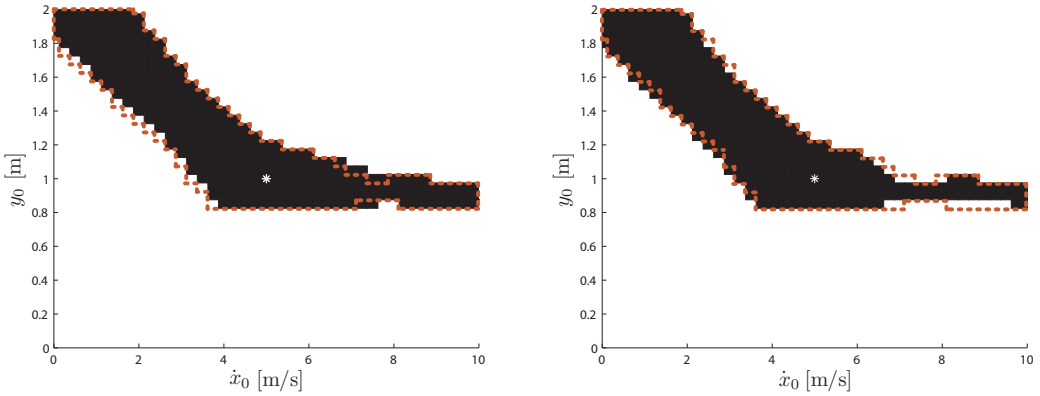
APPENDIX B
EFFECTS OF PERSISTENT SENSORY ERRORS ON ANGLE OF ATTACK CONTROLLER DISTURBANCE REJECTION

In section VI-B the effects of random sensor errors on the angle of attack controllers was shown. Another type of sensor errors that can occur is a persistent sensor error, where the error is a certain constant value. Here the effects of these persistent sensor errors are investigated. As for the random errors, swing-leg retraction is only affected by apex time errors and dead-beat control is affected by apex height and apex forward speed errors. Constant angle of attack control is not affected by sensor errors. Persistent errors investigated here will have a value of plus or minus 5% of the actual value.

The effects of persistent apex time errors on swing-leg retraction can be found in figure 24. This figure shows that persistent apex time errors have only a very small effect on swing-leg retraction.

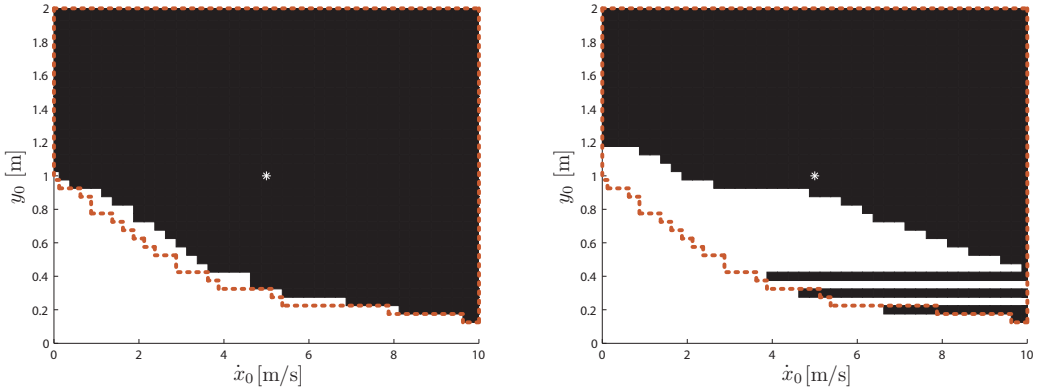
The effects of persistent apex height errors on dead-beat control can be found in figure 25. It can be seen in this figure that underestimating the apex height hardly affects the dead-beat controller; however, overestimating the apex height decreases the BoA of the dead-beat controller a lot, especially for initial apex heights lower than 1m. This is because overestimating the apex height leads the controller to believe there is more ground clearance than there actually is, which can cause the controller to select an angle of attack for which the foot will always be below the ground. This always results in a fall.

The effects of persistent apex forward speed errors on dead-beat control can be found in figure 26. In this figure it can be seen that apex forward speed errors hardly have an effect on dead-beat control.



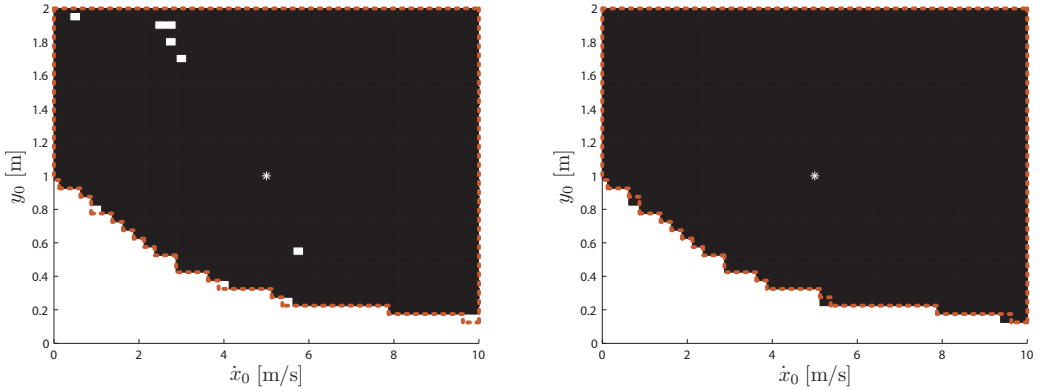
(a) Persistent apex time errors of -5% of the actual value (b) Persistent apex time errors of +5% of the actual value

Figure 24: Basin of Attraction for swing-leg retraction with persistent apex time errors.



(a) Persistent apex height errors of -5% of the actual value (b) Persistent apex height errors of +5% of the actual value

Figure 25: Basin of Attraction for dead-beat control with persistent apex height errors.



(a) Persistent apex forward speed errors of -5% of the actual value (b) Persistent apex forward speed errors of +5% of the actual value

Figure 26: Basin of Attraction for dead-beat control with persistent apex forward speed errors.

APPENDIX C
RESULTS FOR COMBINATION OF DEAD-BEAT CONTROL AND SWING-LEG RETRACTION

In this study, swing-leg retraction and dead-beat control were investigated as two separate controllers. However, it is also possible to combine the controllers. Dead-beat control will determine the desired angle of attack at touchdown, but the leg will swing backwards around this point just before touchdown. This may have both energy and robustness benefits over dead-beat control alone. Energy benefits are explained by the reduction of the foot velocity relative to the ground, in the same way as for swing-leg retraction. Robustness benefits are explained by the fact that swing-leg retraction automatically adapts the angle of attack to a change in apex height, due to changes in flight time. This means that the swing-leg action can (partially) compensate for an apex height sensor error if the right retraction rate is selected.

For the SLIP model this means that, if all sensor information is correct, the leg will still hit the ground at the angle given by the dead-beat controller. Thus, most results on the SLIP model are exactly the same for the combined control as they are for the dead-beat controller as discussed in this study. Combined control can improve the robustness of the controller for apex height sensor errors. To show this, the same process is used as in section VI-B: The apex height sensor will give values uniformly distributed between plus and minus 5% of the actual value at every step. Ten Basins of Attraction (BoAs) are constructed this way, and are summed. The result can be seen in figure 27. Figure 27 shows that, although the combined controller is slightly affected by apex height errors, the combined controller does indeed increase the robustness to apex height sensor errors, as compared to the dead-beat controller in figure 16.

Since combined control uses swing-leg retraction, it can also be affected by apex height errors. To investigate this, ten BoAs are constructed and summed, having a measured apex time uniformly distributed between plus and minus 5% of the actual value at every step. The result can be seen in figure 28. This figure shows that combined control is very much affected by apex time errors, especially for apex speeds above 5m/s.

Unfortunately, implementation of combined control on the realistic model is not easy. To use combined control, the exact touchdown time has to be known, so that the touchdown angle will indeed be (close to) the angle of attack determined by the dead-beat controller, assuming no errors in the sensor inputs. However, there is no easy way to determine the touchdown time. Two methods were attempted to estimate the touchdown time. The first method assumed that the hip follows a ballistic flight trajectory. The second method assumed that the flight time of the second part of the flight phase - from apex until touchdown - is equal to the flight time from liftoff to apex. Both assumptions are too crude to use as a touchdown time estimator, and it was not possible to find a limit cycle for the combined controller on the realistic model using these assumptions. To be able to use the combined controller, a different method of estimating the touchdown time should be used.

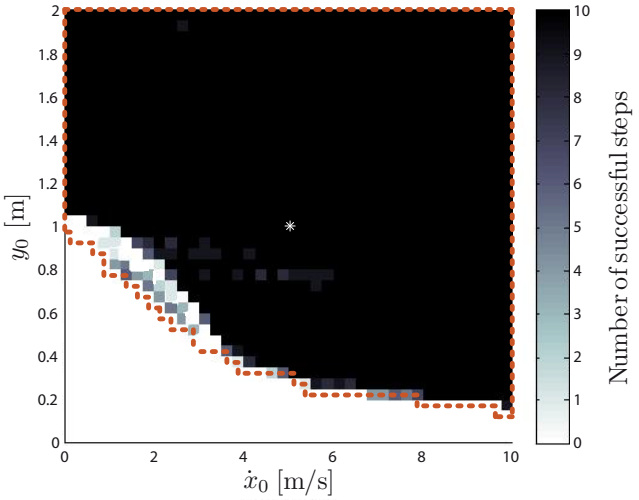


Figure 27: Robustness of combined controller on SLIP model. Apex height sensor errors plus/min 5%. The dotted line indicates the original basin of attraction.

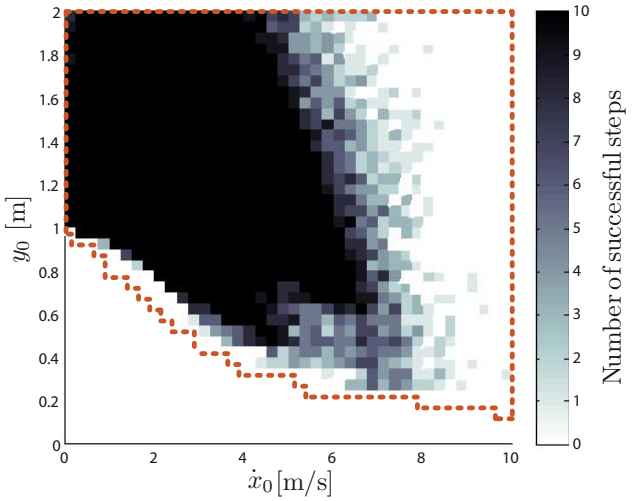


Figure 28: Robustness of combined controller on SLIP model. Apex time sensor errors plus/min 5%. The dotted line indicates the original basin of attraction.

Literature study

Angle of Attack Controllers for Running Robots

Delft Biorobotics Laboratory, Mechanical Engineering, Delft University of Technology
Karin Griffioen, Advisor: Daniël Karssen
November 10, 2010

***Abstract*—In the control of running robots, one of the most important aspects is the placement of the leg, or the angle of attack control. The angle of attack is defined as the angle between the leg of the robot and the ground, at the moment of touchdown. By controlling the angle of attack, the ratio between height and forward speed at the next hop of the robot is controlled. Due to the non-integrability of the equations of motion of a running robot model, an ideal angle of attack cannot be determined analytically. Several different angle of attack controllers have been proposed in literature, however a clear comparison of all controllers has never been made. The goal of this study is to determine which angle of attack controllers already exist and to compare their performance on a simple model of a running robot. An extensive literature study shows that five different types of angle of attack controllers exist. Controllers are compared both quantitatively, comparing resulting angles of attack to a numerically computed ideal angle of attack, and qualitatively, looking at the number of sensors a controller needs and energy benefits the controller may have. This study shows that two types of controllers have very good overall results: swing-leg retraction and approximate return map control. Swing-leg retraction swings the leg backwards during the flight phase, thereby adapting the angle of attack to the state of the robot. Approximate return map control simplifies the equations of motion, so that an estimate of the return map is found, which is used as an angle of attack controller. The other three types of controller produce angles of attack so far from the ideal angle that they should not be considered as controllers for running robots.**

***Index Terms*—running robot, SLIP, angle of attack**

I. Introduction

THE Delft Biorobotics Laboratory (DBL) has been building bipedal robots for many years. Recent DBL robots include walking robots Mike [1], Max [2], Denise [3], [4] and Flame [5] and soccer robot TUlip [5]. Inspired by the successes in walking robotics, the DBL's new goal is to build a running robot.

The first mention in literature of a running or hopping robot was in 1969, when a feasibility study was performed for a lunar hopping transporter [6]. Since then, research on running robots has progressed, and today running robots include hexapods (e.g. [7], [8]), quadrupeds (e.g. [9], [10]), bipeds (e.g. [11], [12]) and monopods (e.g. [13]–[15]).

The reason for studying running robots is twofold. First of all, legged robots provide more mobility than e.g. wheeled robots. It is much more difficult to navigate challenging terrain such as stairs, holes, etc. for a wheeled robot than it is for a legged robot. When a legged robot is walking, its speed is limited to $\sqrt{gl}$ [16]. To increase the speed above this limit, the robot has to start running. Thus, a running robot can provide better mobility than a wheeled robot, at a higher speed than a walking robot. The second reason to study running robots is that they can be a model for human (and animal) running. The complex task of locomotion is performed easily by humans, but still poorly understood by science. A better understanding of the science of locomotion makes it easier to help people who now have trouble or are unable to walk and run.

While design of the DBL running robot has already started, it is important to consider how it can best be controlled. To control a running robot, three basic actions can be taken. First, the upper body can be controlled by applying a hip torque. Second, leg thrust can be applied during the stance phase or at lift-off. Finally, leg placement at touchdown can be controlled by controlling the angle of attack. This study will focus on angle of attack control. Put simply, this control determines at what angle the leg should hit the ground at touchdown. Changing the angle of attack will determine the ratio between forward speed and height at the next hop. A steep angle of attack will result in high forward speed but low height; a flat angle of attack will result in low forward speed but great height at the next hop.

To decide on the best angle of attack controller for the DBL running robot, a literature study on existing controllers is performed. Controllers found in literature are compared on a basic level; promising controllers will be studied in-depth at a later stage. Results of the literature study and comparison of the controllers can be found in this paper.

The remainder of this paper is organized as follows: Section II introduces the SLIP-model: an important model for running robotics, which will be used throughout this paper. Section III describes the angle of attack controllers found in literature. In Section IV, the controllers are compared in a quantitative manner, whereas in Section V they are compared qualitatively. Some points of discussion are found in Section VI and conclusions are found in Section VII. Finally, plans for future work are presented in Section VIII.

II. The SLIP-model

An important model in running robotics is the Spring Loaded Inverted Pendulum (SLIP) model. This model consists of a point mass m attached to a massless spring with spring constant k and rest length l_0, see figure 1. Even though this model is so simple, research suggest that it is a good model for human and robot running, and that the gaits found with this model are similar to human and animal running gaits [17], [18]. According to Full and Koditschek [19], the SLIP-model can be considered a *template* for running, where a template is

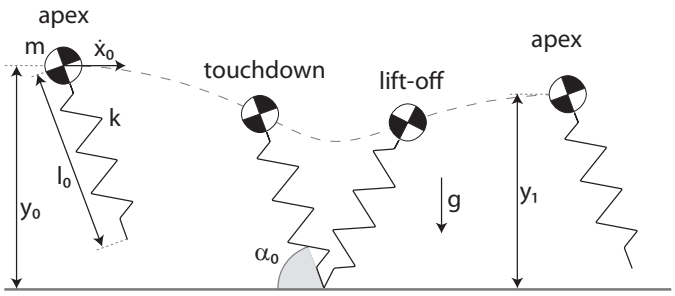


Figure 1: The Spring Loaded Inverted Pendulum (SLIP) model. This model consists of a point mass m, attached to a massless linear spring, with spring constant k and rest length l_0. The model is subject to gravity g. The highest point of the flight phase is called apex; the instances where the foot touches and leaves the ground are called touchdown and lift-off respectively. The angle of the leg with respect to the ground at the moment of touchdown is called the angle of attack α_0.

defined as 'the simplest model (least number of variables and parameters) that exhibits a targeted behavior'.

For the SLIP-model, equations of motion are divided in stance phase and flight phase equations of motion. The flight phase can be characterized as ballistic flight, leading to the following equations of motion:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \end{bmatrix}, \tag{1}$$

where:

$\ddot{x}$ is the horizontal acceleration of the point mass,
$\ddot{y}$ is the vertical acceleration of the point mass and
g is the gravitation constant.

For the stance phase, the equations of motion are:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} -\cos\alpha \\ \sin\alpha \end{bmatrix} \frac{F_s}{m} + \begin{bmatrix} 0 \\ -g \end{bmatrix}, \tag{2}$$

where:

F_s is the spring force,
α is the angle between leg spring and ground and
m is the mass of the point mass.

In this study a linear spring is used, so the spring force becomes:

$$F_s = k \cdot (l - l_0), \tag{3}$$

where:

k is the spring constant,
l is the spring length and
l_0 is the rest length of the spring.

The transition from flight phase to stance phase is called touchdown. This happens when the foot hits the ground, so when:

$$y = l_0 \sin\alpha_0, \tag{4}$$

where:

α_0 is the angle between leg spring and ground at touchdown, also called the angle of attack

The transition from stance phase to flight phase is called lift-off. This happens when the foot leaves the ground, which is when the leg spring returns to its rest length, so when:

$$l = l_0 \qquad \text{and} \qquad \dot{l} > 0. \tag{5}$$

The state of the SLIP-model consists of horizontal and vertical positions x and y and horizontal and vertical speeds $\dot{x}$ and $\dot{y}$. At the highest point of the flight phase, called the apex, the model is completely described by the height y and the horizontal speed $\dot{x}$. This is because the horizontal position x is irrelevant and the vertical speed $\dot{y}$ is zero by definition, since the apex is the highest point of the flight phase. If the SLIP-model is undisturbed and the ground level stays the same, the system is conservative, since no energy can be lost or gained. For such a conservative SLIP-model the state at the apex can even be described by height y alone, since the horizontal speed $\dot{x}$ is then coupled to the height through the system energy E_s.

A step for the SLIP-model is considered the motion from one apex to the next apex. To control a running robot one would like to find a stable *limit cycle* for a step. A limit cycle is defined in [21] as 'a series of exact repetitions of a closed trajectory in state space'.

To analyze the existence and stability of a limit cycle for a running robot, a mapping from one apex point to the next apex point can be used. Such a mapping is called a Poincaré or return map. A return map gives us the state at the next apex $\mathbf{v}_{i+1}$ as a function of the state at the current apex $\mathbf{v}_i$:

$$\mathbf{v}_{i+1} = S(\mathbf{v}_i). \tag{6}$$

The function S is called the stride function. The periodic motion required for a limit cycle exists if a fixed point $\mathbf{v}^*$ exists where:

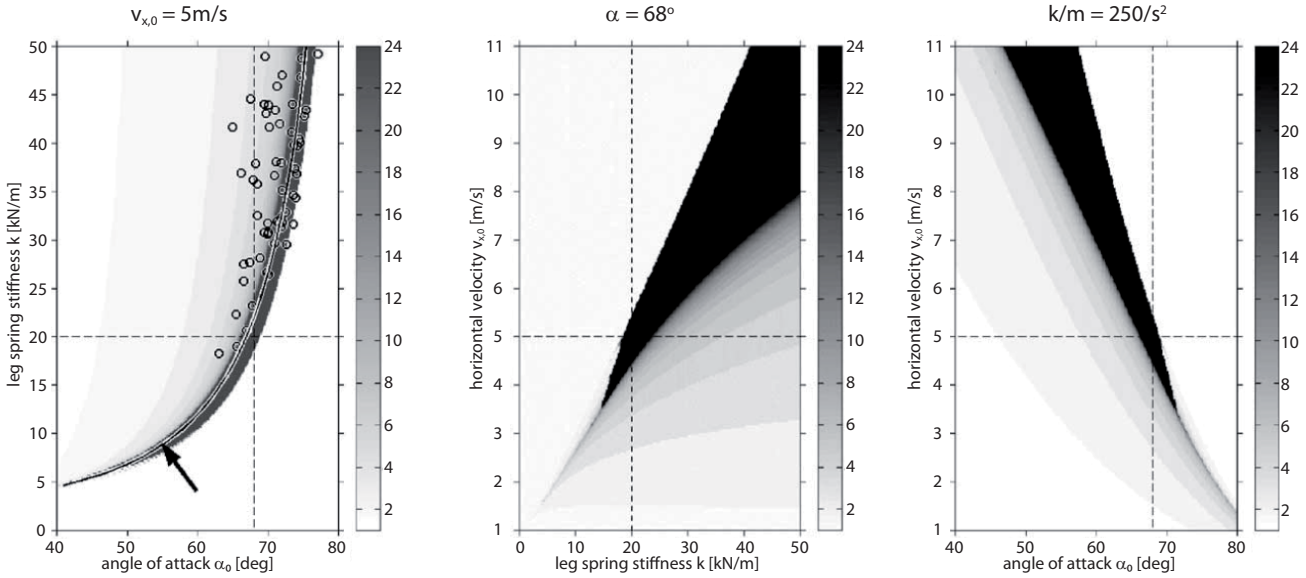
$$\mathbf{v}^* = S(\mathbf{v}^*). \tag{7}$$

The stability of the limit cycle can be analyzed by looking at the eigenvalues of the partial derivative of the stride function S to the state $\mathbf{v}$. If the eigenvalues are within the unit circle in the complex plane, the stability of the trajectory is guaranteed and a limit cycle exists.

Existence and stability of a limit cycle can also be checked graphically, if a plot is made of the return map [22]. A limit cycle exists if there is a point on the return map which is located on the line $y_{i+1} = y_i$. Stability of this limit cycle is guaranteed if the slope at this point has an absolute value smaller than 1.

III. Angle of attack controllers

Generally, to find the desired angle of attack for a running robot, one would integrate the equations of motion of the



(a) Steps to fall for constant horizontal velocity (b) Steps to fall for constant angle of attack (c) Steps to fall for constant leg stiffness

Figure 2: A proper combination of leg stiffness k, angle of attack α_0 and forward speed $v_{x,0}$ can produce stable running. This figure shows data from a simulation on the SLIP-model. The simulation is run until either the point mass falls on the ground ($y = 0$) or until 24 steps are performed. The grayscale legends to the right of each figure indicate the number of steps that the SLIP-model performed. Each figure keeps one parameter constant. In (a) horizontal velocity is kept constant at a value of $5m/s$. Experimental data of a robot running at a horizontal velocity of $4.6 \pm 0.5m/s$ are also shown with small circles. The arrow points to a solid line with function $k \cdot (1 - \sin\alpha_0) = 1600N/m$. In (b) the angle of attack is kept constant at a value of 68°. In (c) the leg stiffness scaled to the body mass (i.e k/m) is kept constant at a value of $250/s^2$. Initial conditions used: $y_0 = 1m$, $v_{y,0} = 0m/s$ Model parameters used: $m = 80kg$, $l_0 = 1m$. From [20].

SLIP-model, insert the current state and desired state, and solve for the angle of attack. Unfortunately, the SLIP equations of motion are non-integrable, forcing researchers to find a different way to control the angle of attack. Through extensive literature research, five types of angle of attack controllers are found:

A. Constant angle of attack controller

The most basic controller that was found is the constant angle of attack controller [20], [23], [24]. This controller sets the angle of attack to a constant value, independent of any input:

$$\alpha_0 = \text{const} \tag{8}$$

In [20], Seyfarth et al. investigate whether such a simple control scheme can generate stable running patterns. They perform a simulation of a running SLIP-model with constant angle of attack, which shows that a proper combination of leg stiffness k, angle of attack α_0 and forward speed $v_{x,0}$ can indeed produce stable running, see figure 2. Figure 2a shows that for a given initial apex speed ($v_{x,0} = 5m/s$) and height ($y_0 = 1m$), stable running patterns are found for combinations of leg stiffness k and angle of attack α_0 that lie in a J-shaped region of the (α_0, k)-plane. Seyfarth et al. fit a function through this region:

$$k \cdot (1 - \sin\alpha_0) = \text{const}. \tag{9}$$

The constant is found to have a value of $1600N/m$ for the specific set of parameters used. Figures 2b and 2c show the influence of horizontal speed on stable running. From these figures, it can be concluded that running with constant angle of attack control requires a certain minimum horizontal velocity, about 3.5 m/s for the parameter set used by Seyfarth et al.. Furthermore it can be concluded that increasing the horizontal velocity results in a larger region of stable running in the (α_0, k)-plane. This increased horizontal velocity will require a higher leg stiffness and/or a flatter angle of attack.

B. Raibert controller

Marc Raibert is considered one of the pioneers in the field of running robotics. Over 20 years ago, he designed a simple but very effective controller [25]. Raibert decomposes control of a robot in three distinct parts: hopping height, forward speed and upper body attitude. Of these three parts, forward speed is controlled by the angle of attack. To control the angle of attack, Raibert introduces the concept of *neutral point*. The neutral point is described as 'the unique foot position that results in zero net forward acceleration', where *net forward acceleration* is defined as the difference between forward speed at touchdown and at lift-off. For the SLIP-model, placing the foot at the neutral point will result in an apex height and speed equal to the previous apex height and speed. Due to the non-integrability of the equations of motion, the neutral point cannot be determined exactly. Raibert has made

an estimation of the neutral point based on forward speed and stance time of the previous step:

$$x_{f0} = \frac{\dot{x}T_s}{2}, \tag{10}$$

where:

x_{f0}	is the horizontal distance between the foot and center of mass for the neutral point,
$\dot{x}$	is the forward speed and
T_s	is the stance time of the previous stance phase.

This estimation of the neutral point can be understood as follows: Assuming that the forward speed during stance is constant and that the stance time stays the same for all stance phases, $\dot{x}T_s$ gives the distance traveled during the stance phase. Assuming furthermore that the stance phase is symmetric - which is true if the apex states before and after the stance phase are equal - then $\frac{\dot{x}T_s}{2}$ gives the neutral point.

Of course, it may not always be the objective to place the foot at the neutral point, which gives zero net forward acceleration. To accelerate, the foot is displaced from the neutral point, according to the following formula:

$$x_{f\Delta} = k_{\dot{x}} \left(\dot{x} - \dot{x}_d \right), \tag{11}$$

where:

$x_{f\Delta}$	is the horizontal displacement of the foot from the neutral point,
$\dot{x}_d$	is the desired forward speed and
$k_{\dot{x}}$	is a feedback gain.

This leads to the following complete formula for foot placement:

$$x_f = \frac{\dot{x}T_s}{2} + k_{\dot{x}}(\dot{x} - \dot{x}_d). \tag{12}$$

where:

x_f	is the horizontal distance between the foot and the center of mass.

Knowing x_f, it is easy to calculate the necessary angle of attack:

$$\alpha_0 = \arccos\left(\frac{\dot{x}T_s}{2l_0} + \frac{k_{\dot{x}} \left(\dot{x} - \dot{x}_d \right)}{l_0} \right) \tag{13}$$

The feedback gain $k_{\dot{x}}$ is determined empirically. For Raiberts one-legged hopping machine $k_{\dot{x}}$ has a value of 0.035m/(m/s) [26].

C. Swing-leg retraction controller

In human and animal running, it can be observed that the swing-leg is retracted with respect to the ground just before touchdown [27] (see figure 3). This is called swing-leg retraction. The idea of retracting the swing-leg before touchdown was already mentioned in 1986 by Raibert [25]. Raibert called this *ground speed matching*, and although he did not implement it on his hopping machines at the time, he believed it could improve their performance.

Figure 3: Photographs of a human running. Retraction of the swing-leg is observed in these photographs. Adapted from [27].

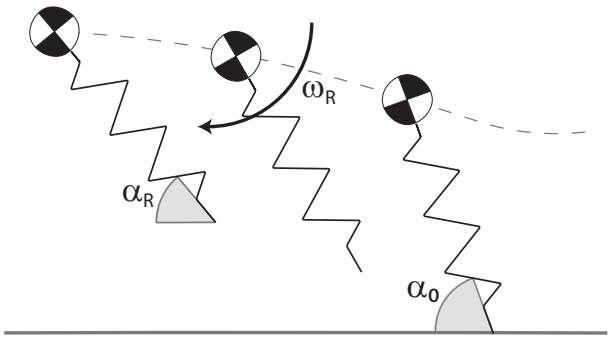


Figure 4: Swing-leg retraction for SLIP-model. The angle of attack α_0 is dependent on the retraction angle at apex α_R and the retraction rate ω_R.

As is expected, swing-leg retraction reduces the horizontal velocity of the foot with respect to the ground at touchdown, and therefore reduces landing impact [28]. Many years since Raibert first mentioned the possibility of *ground speed matching*, Seyfarth et al. started investigating this further [22]. They showed that, apart from the energy benefits, *swing-leg retraction*, as they call it, also improves stability. Seyfarth et al. consider swing-leg retraction with a constant retraction rate, where the retraction angle at apex is α_R and the swing-leg retracts from apex until touchdown with a constant retraction rate ω_R (see figure 4).

Applying swing-leg retraction, the angle of attack becomes:

$$\alpha_0 = \alpha_R + \omega_R t, \tag{14}$$

where:

t	is the flight time from apex until touchdown.

The benefit of swing-leg retraction control is that the angle of attack is automatically adapted to a change in apex height, because of the swing-leg action. An increased height of a runner at apex, will result in a longer flight time. This will in turn result in a steeper angle of attack, as can be seen from equation 14. For a decreased height at apex, the flight time will be shorter, resulting in a flatter angle of attack. Proper

selection of the apex retraction angle and retraction rate can improve the stability of a running robot [22].

D. Approximate return map controller

To control a running robot, it would be ideal to find an exact return map of the state of the robot at the apex. For the SLIP-model with constant system energy, the state at the apex is completely determined by the apex height. Thus, a return map would give the height at the next apex y_{i+1} as a function of the height at the previous apex y_i and the angle of attack α_0:

$$y_{i+1} = f\left(y_i, \alpha_0\right). \tag{15}$$

To find such a return map, the equations of motion of the model of the robot would have to be integrated. Unfortunately, it is impossible to integrate the equations of motion of even a model as simple as the SLIP-model. This means it is not possible to find an exact form of the return map. Many researchers have made approximations of the equations of motion for the SLIP-model, in order to find an approximate return map:

$$y_{i+1} \approx \hat{f}\left(y_i, \alpha_0\right). \tag{16}$$

Having a (approximate) return map of y_{i+1} as a function of y_i and the angle of attack α_0, this equation can also be rewritten to obtain the necessary angle of attack α_0 as a function of the current state y_i and the desired state $y_{i+1,d}$:

$$\alpha_0 = h\left(y_i, y_{i+1,d}\right). \tag{17}$$

This equation can be used as an angle of attack controller.

In 2000, Schwind and Koditschek proposed an approximate return map that uses an iterated application of the mean value theorem for integral operators [29]. Both performance and complexity of the return map increase with the number of iterations.

In 2005, Geyer et al. proposed a much simpler approximation of the return map, assuming small angular sweep and small spring compression during stance [30].

Both solutions described above assume steps that are symmetric, so that gravity can either be ignored or linearized during the stance phase. This inspired Arslan et al. to contrive a return map with gravity correction, which also takes into account steps that are non-symmetric [31].

Arslan et al. have also compared the three return maps described above [31]. For certain initial conditions, they computed the next apex height by numerically integrating the SLIP-model numerically. Then they computed the prediction of the next apex height for all three return maps. This was done for all angles no more than 0.4rad from the neutral angle of attack. The neutral angle of attack is the angle which results in a symmetric step, so that the next apex height equals the previous apex height. The results of the comparison of the three return map controllers can be seen in figure 5.

As seen in figure 5, none of the controllers outperform the others for every touchdown angle, so no definite conclusion can be reached as to which will perform best in the comparison

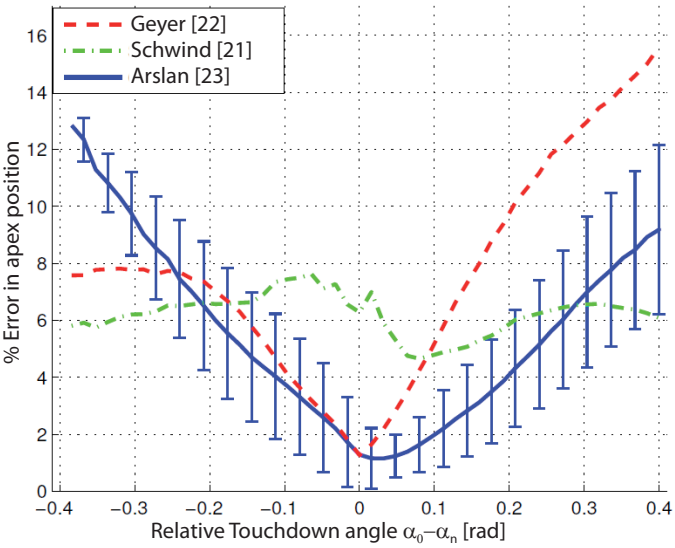


Figure 5: Mean apex position percentage error versus relative touchdown angle. From [31].

that will be done in this research. For simplicity reasons, the comparison of the different types of controllers will be performed on the approximate return map by Geyer et al. [30]. If results are promising, the different versions of approximate return map control will be studied further.

E. Passive dynamic running

Passive dynamically walking robots, which are powered only by gravity, have been researched extensively. McGeer showed that, given the right initial conditions, two-legged robots are able to perform stable walking on a shallow slope, without active control or energy supply other than gravity [32]. Collins et al. built a three-dimensional walker [33] that showed the passive-dynamic stability described by McGeer. Inspired by the success in walking robots, researchers started looking at passive dynamic running. McGeer looked extensively at the possibilities of passive dynamic running [34]. He found that, while some modes exhibit inherent stability, others need to be stabilized actively. Thompson and Raibert also investigated the stability of passive dynamic running and optimized initial conditions and parameters to find trajectories that are nearly reentrant [35].

Since passive dynamic running is not inherently stable, several people have devised controllers to stabilize the trajectories. Ahmadi and Buehler propose a controller based on ‘online calculations of the desired passive dynamic motion’, which they call Controlled Passive Dynamic Running, or CPDR [36], [37]. They have implemented this controller successfully on their ARL-Monopod II [38].

Francois and Samson derive 1-periodic solutions of a simplified integrable model of a hopper, which are stabilized using impulsive control inputs [39].

In [40], Hyon and Emura propose an energy-preserving control strategy. They expect that preserving the system energy will let the system generate natural periodic gaits autonomously.

All three controllers described above have been shown to stabilize passive dynamic running. The comparison of the controllers in the next sections will be based on completely passive dynamic running, without stabilizing control, to investigate the basic characteristics of passive dynamic running. If results are promising, further research will be done on the control of passive dynamic running.

For passive dynamic running, a design choice needs to be made to select an appropriate hip oscillation frequency ω_h. This hip oscillation frequency depends amongst others on the choice of the hip spring stiffness k_h. The leg angle α is related to the hip oscillation frequency through the following formula:

$$\ddot{\alpha} = {\omega_h}^2 \left(\frac{1}{2}\pi - \alpha \right). \tag{18}$$

Thus, α becomes a function of time and the hip oscillation frequency:

$$\alpha = f\left(t, \omega_h\right). \tag{19}$$

For a given hip oscillation frequency the angle of attack becomes:

$$\alpha_0 = \alpha\left(t = t_{touchdown}\right). \tag{20}$$

IV. Quantitative comparison

Now that the five types of existing angle of attack controllers are introduced, it is important to know how they compare to each other. To get a first impression of the performance of the controllers, a simple simulation is performed. This simulation will show how close the controllers get to the ideal angle of attack for a certain situation. The methods for this simulation will be discussed in Section IV-A. The results of the simulation will be discussed in Section IV-B.

A. Method

The simulation is performed on the SLIP-model as described in Section II. The equations of motion of the SLIP-model are integrated using Matlabs ode45 integrator. Parameters are chosen at $m = 80kg$, $l_0 = 1m$, $k = 20kN/m$, $g = 9.81m/s^2$, both because of conformance to human values, and to facilitate comparison to literature results, where these values are becoming somewhat standard values. The model is started in a limit cycle with $y_{apex} = 1m$ and $\dot{x}_{apex} = 5m/s$. At apex, the model is disturbed, in such a way that the total energy of the system is conserved. This means that if the height at apex is increased by the disturbance, the horizontal speed at apex is lowered accordingly so that the total system energy stays the same. The disturbances are up to 10% of the original apex height, so that the apex height ranges between 0.9m and 1.1m and the apex horizontal speed ranges accordingly between 5.2m/s and 4.8m/s. The goal is to go back to the original limit cycle, so the desired height at the next apex is 1m. The model is run for all the initial apex values with all the controllers, to see what angle of attack the controllers give. These angles of attack are then compared to the *ideal angle of attack*. The ideal angle of attack is the angle which gets the model exactly to its desired state at the next apex. This ideal angle of attack cannot be determined analyticaly, but it can be calculated numerically.

Most controllers have control and design parameters that need to be set. If possible, the control parameters are chosen so that if started in the desired limit cycle, the controller will give an angle of attack equal to the ideal angle of attack. This means that when the controller is started in this limit cycle, it will stay in the desired limit cycle. If this still leaves choices for other parameters (as is the case for the Raibert controller and swing-leg retraction), these parameters are optimized numerically, so that the resulting angles of attack over the entire range of starting heights are as close to the ideal angle of attack as possible.

As a measure of performance, the average distance between angle of attack given by the controller and the ideal angle of attack over the entire range of starting heights is taken.

B. Results

Results of the simulations can be seen in figure 6. First of all, this figure shows the ideal angle of attack for apex heights ranging from 0.9m to 1.1m. Starting at a given apex height, this ideal angle of attack gets you exactly to your desired apex height at the next apex (in this case an apex height of 1m). Figure 6 also shows the results for all the controllers discussed in Section III. Looking at the results for the constant angle of attack controller, it is immediately obvious that this controller indeed keeps the angle of attack constant, as the resulting angle of attack is a straight line. The value of the constant angle of attack is chosen so that when started in the desired state, the model will stay in the desired state. This is clear from figure 6, where the constant angle of attack crosses the ideal angle of attack at an apex height of 1m (which is the desired apex height).

What stands out for the Raibert controller is that the resulting angle of attack is never equal to the ideal angle of attack. As discussed in Section III-B the Raibert controller makes some assumptions to determine the neutral point; namely that the forward speed stays constant throughout the stance phase, that the stance time in the next step will be equal to the stance time of the previous step and finally that the stance phase is symmetrical. Since these assumptions are never exactly correct, the calculated neutral point is never exactly right. This is most obvious when the model is started at an apex height of 1 meter. Since this is the desired apex height, there is no P-action and the only control is from the estimated neutral point. The difference with the ideal angle of attack shows that the neutral angle of attack is off by about 1^o at this point. Calculating back, the estimated neutral point (the distance between hip and toe) is off by about 1.5cm, or 4%. In this simulation, the feedback gain for the Raibert controller $k_{\dot{x}}$ is kept constant. The value of the gain is optimized numerically, in such a way that the average difference with the ideal angle of attack over the entire range of starting apex heights is minimized. This results in an optimal value for the feedback gain of 0.11.

Figure 6 also shows that the resulting angle of attack for swing-leg retraction is very close to the ideal angle of attack.

Swing-leg retraction control has two parameters which can be chosen freely, the apex retraction angle α_R and the retraction rate ω_R. As in constant angle of attack control, it is required that, when started at the desired apex height, the model will return exactly to this desired apex height at the next flight phase. In other words, the angle of attack at the desired apex height should be equal to the ideal angle of attack. This leaves only one of the original two free parameters. This final free parameter is optimized numerically using the same criterion as for the Raibert controller, namely that the average difference with the ideal angle of attack over the entire range of starting apex heights should be minimized. This procedure gives an optimal apex retraction angle of 62^o and a retraction rate of 36^o/s. With these well-chosen apex retraction angle and retraction rate, resulting angles of attack are very close to the ideal angle of attack, reinforcing the choice for a constant retraction rate.

The resulting angle of attack given by approximate return map control is very close to the ideal angle of attack. What stands out most is that, even though no control choices could be made, the resulting angle of attack at the desired apex height of 1m is (practically) equal to the ideal angle of attack. This can be explained by the fact that the approximations to the return map are valid for symmetrical steps. If the states at the beginning and end of the step are equal, the step must be symmetric and the approximate return map controller will give the ideal angle of attack.

Finally, the results for passive dynamic running are also seen in figure 6. For some initial apex heights, the resulting angle of attack is so far off the ideal angle of attack that results lie outside the boundaries of the figure. An explanation for the large difference between passive dynamic running and the ideal angle of attack can be found in the fact that passive dynamic running has no control parameters that can be set. The hip oscillation frequency can be set once during the design stage of the robot, but after that control of the robot is completely passive, not even the desired height can be adapted. While this has the advantage that no sensors are needed, it also results in angles of attack quite far off the ideal angle of attack, as seen in figure 6. Furthermore, not all initial apex heights have a resulting angle of attack. This is because for apex heights lower than 0.95m, the foot starts below the ground and the swing action of the leg is never able to get the foot above the ground. For passive dynamic running, there is a design choice to be made for the hip spring constant. In this case the same condition was used as for constant angle of attack control and swing-leg retraction control, namely that when started at the desired height of 1m, the model should still be at the desired height at the next apex. This results in an angle of attack for passive dynamic running that is equal to the ideal angle of attack for an initial apex height of 1m.

To get a feel for the effects that the resulting angles of attack given by the controllers have on the resulting apex height, figure 7 shows a return map of all the controllers. This figure shows the resulting height at the next apex y_1 for the range of starting apex heights y_0 for all controllers. The starting apex height determines the resulting angle of attack for the controller, which in turn determines the height at the next apex. The ideal angle of attack by definition has the desired height at the next apex, so this is a horizontal straight line through $y_1 = 1m$. The two controllers that have the smallest deviation from the ideal angle of attack, swing-leg retraction and approximate return map control, also result in apex heights that are very close to the desired apex height. The other controllers have angles of attack that result in apex heights much further from the desired apex height of 1m. Both the constant angle of attack controller and passive dynamic running do not have resulting apex heights for some initial apex heights. This is because for both controllers, at certain initial apex heights, the foot starts below the ground and never gets above the ground.

For all controllers, the difference between the resulting angle of attack and the ideal angle of attack is calculated and averaged over the entire range of starting apex heights. The results can be seen in table I. For passive dynamic running, only those starting apex heights where a resulting angle of attack was found are included in the result. As expected from figure 6, swing-leg retraction and approximate return map control have the least difference with the ideal angle of attack and passive dynamic running has the largest difference with the ideal angle of attack.

V. Qualitative comparison

Now that the quantitative results of the controllers are known, this section will investigate the qualitative aspects of the controllers. First the number of sensors necessary for each controller will be discussed. Then, some energy benefits that certain controllers have will be addressed.

A. Sensors

The number of sensors required for a certain controller is important because sensors in a real system will lead to sensor errors. Eventually one of these controllers will be implemented on a real robot. On this real robot, having many errors in the sensor data will not be beneficial for the stability of the robot. Therefore, the fewer sensors a controller needs, the better. Some sensors may produce more errors than others. In general measuring with respect to the ground (instead of internally) will be more error-prone.

For the constant controller, only one sensor is needed, namely a leg angle sensor, which measures the angle between the leg and the ground.

Looking at the Raibert control equation (13) it can easily be seen that two inputs are needed to calculate the desired angle of attack: horizontal speed $\dot{x}$ and stance time at the previous stance phase T_s. To know the stance time a sensor to measure whether there is contact between the foot and the ground is needed. This is quite easy to measure and large errors are not expected here. Finally, a sensor is needed to measure the leg angle. All together three sensors are needed: a speed sensor (or position sensor, with results differentiated), a contact sensor and a leg angle sensor.

Swing-leg retraction control only needs two inputs, as can be seen from (14). The time since the apex is required, which means a sensor to know when the instant of apex occurs is

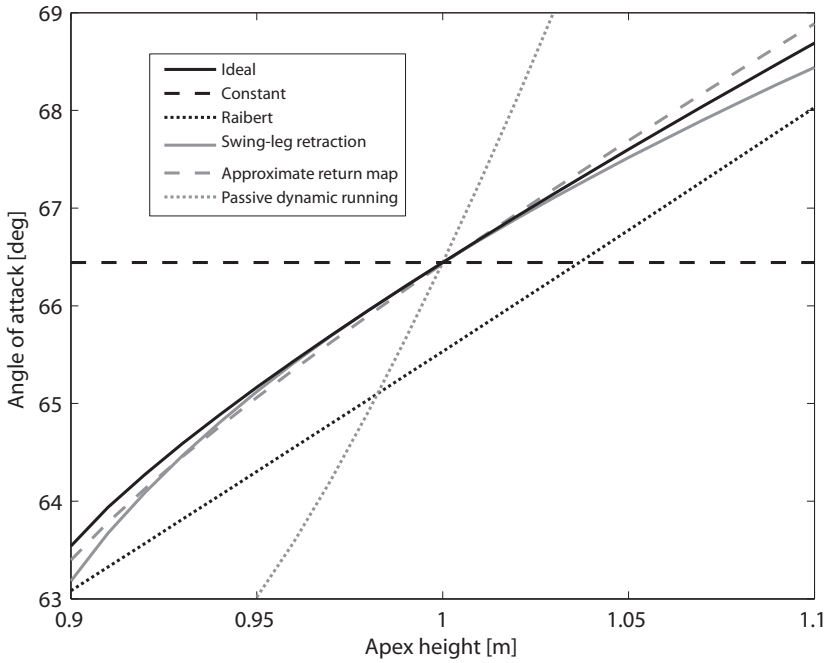


Figure 6: Resulting angle of attack for all controllers found in literature, compared to the ideal angle of attack. All controllers are started from initial apex heights ranging from 0.9 to 1.1m and resulting angles of attack are determined. The ideal angle of attack, which results exactly in the desired height of 1m at the next apex, is determined numerically.

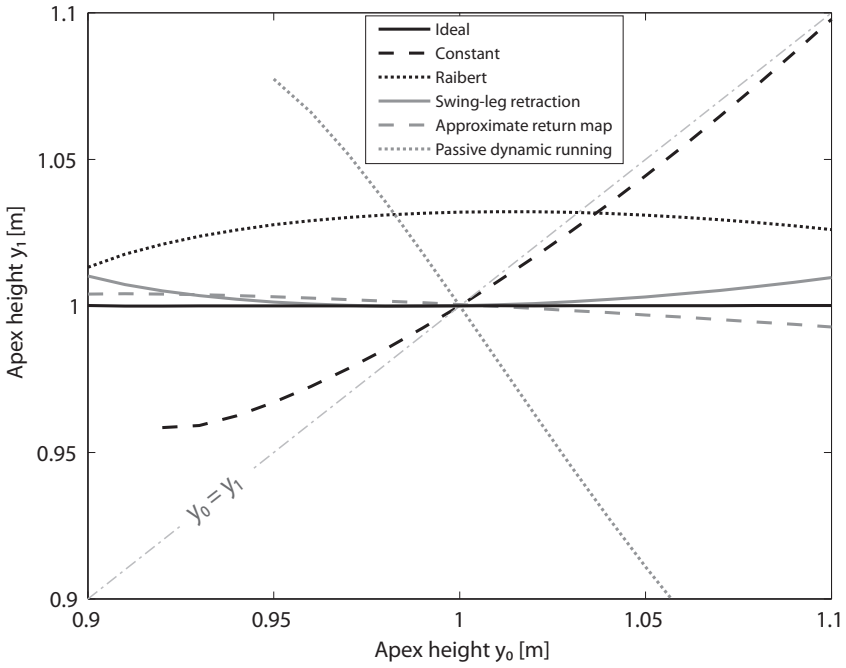


Figure 7: Return map of the controllers found in literature. This return map shows the resulting height at the next hop y_1 as a function of the starting apex height y_0. The starting apex height determines the resulting angle of attack for the controller, which in turn determines the height at the next apex. Controllers that have angles of attack close to the ideal angle of attack, also give resulting apex heights very close to the desired apex height of 1m.

Controller	Average difference with ideal angle of attack [degrees]	Average difference with desired height at next apex [m]	Parameter settings
Constant angle of attack	1.30	0.0385	$\alpha_0 = \text{const} = 66.4^o$
Raibert	0.80	0.0278	$k_{\dot{x}} = 0.11m/(m/s)$
Swing-leg retraction	0.10	0.0035	$\alpha_R = 62^o$, $\omega_R = 36^o/s$
Approximate return map	0.10	0.0032	-
Passive dynamic running	2.86	0.0680	$\omega_h = 931^o/s$

Table I: Average difference between angle of attack given by controllers and ideal angle of attack over range of initial apex heights between 0.9 and 1.1m; Parameter settings

needed. The instance of apex will be quite difficult to measure. Furthermore, a leg angle sensor is needed.

For return map control, three inputs are needed. The speed and apex height are required to calculate the desired leg angle. Also, a leg angle sensor is needed to set the leg angle to the desired value.

Finally, passive dynamic running needs no sensors at all. As the name already indicates, this control is completely passive and requires no inputs.

B. Energy

When looking at controllers for running robots, it is also important to look at energy aspects. As explained before, the SLIP-model is conservative, so energy is not a measure of performance for this model. In real robots (and humans for that matter), energy losses can be substantial and energy use of a controller becomes quite relevant. Furthermore, if a running robot is viewed as a model for human running, it is unlikely that humans use a type of control that uses a lot of energy, when other control is available, provided quantitative results are similar. Two of the controllers discussed in Section III have energy benefits, which will be discussed here.

Swing-leg retraction has energy benefits, because it lowers the ground-speed of the leg. In real robots and humans, legs have mass and therefore impact losses occur at touchdown. By swinging the leg backwards during flight, the horizontal speed of the leg with respect to the ground is lowered and thus impact losses are decreased.

Passive dynamic running also has energy benefits, for two distinct reasons. As found by Alexander in 1990 [16], running animals and robots can save energy if they use a spring to swing their legs forward during flight phase. It was found experimentally that for a certain type of running robot without hip spring, the hip actuator consumes 40% of total energy at top speed, most of which is used for swinging the leg forward [41]. In passive dynamic running, this motion is powered by the hip spring. Furthermore, it can be expected that in normal operation of passive dynamic running, the leg will be swinging backwards during the final part of the flight phase, thereby reducing ground speed of the leg. Just like in the case of swing-leg retraction, this will reduce impact losses.

The results of the qualitative comparison of the controllers are summarized in table II.

VI. Discussion

In the previous sections, five angle of attack controllers from literature were discussed and compared in a qualitative and quantitative manner. This section will first give a discussion of the results of these comparisons and then discuss some restrictions of this this literature study and the way the controllers were compared.

A. Discussion of results

In the comparison of the five angle of attack controllers, two controllers clearly outperform the other three, namely swing-leg retraction and approximate return map control. Especially in the quantitative comparison, they outperform the next best controller by eight times. In the qualitative comparison swing-leg retraction control also performs well, needing only two sensors (one to measure the instance of apex and one to measure the leg angle). Moreover, swing-leg retraction also has energy benefits, since retraction of the swing-leg lowers impact losses at touchdown. Approximate return map control needs three sensors. Because of very good quantitative results, it is still considered a promising controller. Both controllers should be investigated further to determine which would perform best on a real robot.

The constant angle of attack controller, Raibert controller and passive dynamic running give far less promising results. All three controllers give angles of attack that are quite far from the ideal angle of attack, especially compared to the two promising controllers. Constant angle of attack control is the most basic control available and needs only one sensor, yet quantitative results are disappointing. The assumptions that have to be made for the Raibert controller are too crude to give an angle of attack close to the ideal angle of attack. Passive dynamic running has the benefit of being completely passive and requiring no sensors at all, but it also has no control parameters, resulting in very high deviations from the ideal angle of attack.

B. Limitations

Some limitations exist to the generality of the research discussed in this paper. First of all, the quantitative comparison of the model was performed on the SLIP-model. This is the simplest model possible of a human or robot runner, consisting of a point mass as a body and massless spring as a leg. It has only one leg. As was seen in Section II, literature suggests that gaits found with this model are very similar to gaits seen in human and robot runners, and controllers that perform well on this model usually also perform well when implemented on a real robot. To corroborate our belief that the current results found with the SLIP-model are valid, future tests on the more promising controllers will be performed on a more complete model of a running robot.

Controller	Number of sensors	Energy benefits
Constant angle of attack	α_0	
Raibert	$\dot{x}$, T_s, α_0	
Swing-leg retraction	t_{flight}, α_0	+
Approximate return map	$\dot{x}$, y, α_0	
Passive dynamic running	-	++

Table II: Summary of the qualitative comparison of angle of attack controllers

In this paper, the controllers are all compared to the ideal angle of attack, but the ideal angle of attack is never considered as a controller in its own. There are several reasons for this. First and foremost, this is a literature study, and only controllers found in literature are considered here. In the future, the ideal angle of attack may be considered as a controller. A point of attention will be that the ideal angle of attack for a certain situation needs to be computed numerically. It either needs to be computed in real-time, which might be too computationally intensive to implement on a real robot, or extensive lookup tables need to be constructed. Also, to know the ideal angle of attack it is still necessary to know the state of the robot, thus requiring at the very least a height and leg angle sensor.

VII. Conclusions

In this paper, the five types of angle of attack controllers that are found in literature are compared in a quantitative and a qualitative manner. From these comparisons it can be concluded that swing-leg retraction and approximate return map control have the best overall results and thus provide the most promising control. Results for the other three controllers are not nearly as good; thus they are not considered promising controllers.

VIII. Future work

Now that two promising angle of attack controllers have come out of this literature study, future work will have to determine which controller will be implemented on the real running robot that is being built in the DBL laboratory. To determine the appropriate controller for the DBL robot, a more complete simulation model of the robot will be built. This model will likely include a torso with inertia (where the mass of the torso may or may not be positioned at the hip of the robot), two legs that have mass and possibly knees.

The model will be used to perform more elaborate analyses of the promising controllers. A stability analysis will show whether certain fixed points are stable with the given controllers. A disturbance rejection analysis will show how far the model can be disturbed without falling over and how quickly it returns to a desired limit cycle. Possible disturbances include steps up/down and pushes. A robustness analysis will show how sensitive the model is to modeling and measurement errors.

The more extensive model of the robot will allow us to examine energy aspects of the controllers, since this model will have energy losses at touchdown and need energy input to swing the leg forward, due to the mass in the legs. These energy aspects will also be considered in the final choice of a controller for the DBL robot.

Finally, angle of attack controllers other than the five found in literature will also be considered as controllers for the DBL robot. As already mentioned in Section VI, an ideal angle of attack controller will be investigated further. Also, combinations of angle of attack controllers will be considered. Swing-leg retraction for example can be combined with e.g. approximate return map control or ideal angle of attack control. The performance of such a combination of controllers will be investigated using the more extensive model and analyses described above.

References

[1] M. Wisse and J. Frankenhuyzen. Design and construction of MIKE; a 2-D autonomous biped based on passive dynamic walking. *Adaptive Motion of Animals and Machines*, pages 143–154, 2006.

[2] M. Wisse, D.G.E. Hobbelen, and A.L. Schwab. Adding an upper body to passive dynamic walking robots by means of a bisecting hip mechanism. *IEEE Transactions on Robotics*, 23(1):112–123, 2007.

[3] M. Wisse. Three additions to passive dynamic walking; actuation, an upper body, and 3D stability. *International Journal of Humanoid Robotics*, 2(4):459–478, 2005.

[4] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient Bipedal Robots Based on Passive-Dynamic Walkers. *Science(Washington)*, 307(5712):1082–1085, 2005.

[5] D. Hobbelen, T. de Boer, and M. Wisse. System overview of bipedal robots flame and tulip: Tailor-made for limit cycle walking. In *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[6] M.H. Kaplan and H. Seifert. Hopping transporters for lunar exploration. *J. Spacecraft and Rockets*, 6(8):917–922, 1969.

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

[8] J.E. Clark, J.G. Cham, S.A. Bailey, E. M. Froehlich, P.K. Nahata, R.J. Full, and M.R. Cutkosky. Biomimetic design and fabrication of a hexapedal running robot. In *IEEE International Conference on Robotics and Automation*, page p. 36433649, 2001.

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

[10] C. Semini, N.G. Tsagarakis, B. Vanderborght, Y. Yang, and D.G. Caldwell. HyQ-Hydraulically actuated quadruped robot: Hopping leg prototype. In *IEEE/RAS Int. Conf. on Biomedical Robotics and Biomechatronics (Biorob)*, pages pp.593–59, 2008.

[11] R. Tajima, D. Honda, and K. Suga. Fast running experiments involving a humanoid robot. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1418–1423. Institute of Electrical and Electronics Engineers Inc., The, 2009.

[12] M. Hirose and K. Ogawa. Honda humanoid robots development. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1850):11, 2007.

[13] B. Brown and G. Zeglin. The bow leg hopping robot. In *1998 IEEE International Conference on Robotics and Automation, 1998. Proceedings*, volume 1, 1998.

[14] M. Ahmadi and M. Buehler. The ARL monopod II running robot: Control and energetics. In *1999 IEEE International Conference on Robotics and Automation, 1999. Proceedings*, volume 3, 1999.

[15] M.H. Raibert, H.B. Brown Jr, and M. Chepponis. Experiments in balance with a 3D one-legged hopping machine. *The International Journal of Robotics Research*, 3(2):75, 1984.

[16] R. Alexander. Three uses for springs in legged locomotion. *The International Journal of Robotics Research*, 9(2):53, 1990.

[17] R. Blickhan. The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11-12):1217–1227, 1989.

[18] R. Blickhan and RJ Full. Similarity in multilegged locomotion: Bouncing like a monopode. *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology*, 173(5):509–517, 1993.

[19] RJ Full and DE Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of Experimental Biology*, 202(23):3325–3332, 1999.

[20] A. Seyfarth, H. Geyer, M. Gunther, and R. Blickhan. A movement criterion for running. *Journal of Biomechanics*, 35(5):649–655, 2002.

[21] D.G.E. Hobbelen and M. Wisse. Limit cycle walking. *Humanoid Robots*, pages 451–468, 2007.

[22] A. Seyfarth, H. Geyer, and H. Herr. Swing-leg retraction: a simple control model for stable running. *Journal of Experimental Biology*, 206(15):2547, 2003.

[23] H. Geyer, R. Blickhan, and A. Seyfarth. Natural dynamics of spring-like running: Emergence of selfstability. In *Proceedings of the Fifth International Conference on Climbing and Walking Robots: and their supporting technologies: CLAWAR 2002, 25-27th September 2002*, page 87. Wiley, 2002.

[24] RM Ghigliazza, R. Altendorfer, P. Holmes, and D. Koditschek. A simply stabilized running model. *Siam Review*, 47(3):519, 2005.

[25] M.H. Raibert. *Legged robots that balance*. MIT press Cambridge, MA, 1986.

[26] MH Raibert and HB Brown Jr. Experiments in balance with a 2D one-legged hopping machine. *ASME Transactions Journal of Dynamic Systems and Measurement Control B*, 106:75–81, 1984.

[27] E. Muybridge. *The human figure in motion*. Dover Publications, 1955.

[28] B. De Wit, D. De Clercq, and P. Aerts. Biomechanical analysis of the stance phase during barefoot and shod running. *Journal of Biomechanics*, 33(3):269–278, 2000.

[29] WJ Schwind and DE Koditschek. Approximating the stance map of a 2-DOF monoped runner. *Journal of Nonlinear Science*, 10(5):533–568, 2000.

[30] H. Geyer, A. Seyfarth, and R. Blickhan. Spring-mass running: simple approximate solution and application to gait stability. *Journal of theoretical biology*, 232(3):315–328, 2005.

[31] O. Arslan, U. Saranli, and O. Morgul. An approximate stance map of the spring mass hopper with gravityorrection for nonsymmetric locomotions. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1829–1834. Institute of Electrical and Electronics Engineers Inc., The, 2009.

[32] T. McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9(2):62, 1990.

[33] S.H. Collins, M. Wisse, and A. Ruina. A three-dimensional passive-dynamic walking robot with two legs and knees. *The International Journal of Robotics Research*, 20(7):607, 2001.

[34] T. McGeer. Passive bipedal running. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 240(1297):107–134, 1990.

[35] C. Thompson and M. Raibert. Passive dynamic running. In *Experimental Robotics I*, pages 74–83. Springer, 1990.

[36] M. Ahmadi and M. Buehler. A control strategy for stable passive running. In *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings*, volume 3, 1995.

[37] M. Ahmadi and M. Buehler. Stable control of a simulated one-legged running robot with hip and leg compliance. *IEEE Transactions on Robotics and Automation*, 13(1):96–104, 1997.

[38] M. Ahmadi and M. Buehler. Controlled passive dynamic running experiments with the ARL-monopod II. *IEEE Transactions on Robotics*, 22(5):974–986, 2006.

[39] C. Francois and C. Samson. A new approach to the control of the planar one-legged hopper. *The International Journal of Robotics Research*, 17(11):1150, 1998.

[40] SH Hyon and T. Emura. Energy-preserving control of a passive one-legged running robot. *Advanced Robotics*, 18(4):357–381, 2004.

[41] P. Gregorio, M. Ahmadi, and M. Buehler. Design, control, and energetics of an electrically actuated legged robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 27(4):626–634, 1997.