

# Quadruped Walking

## Modeling and Optimization of Robot Movements

Diploma Thesis

unofficial but complete translation of German original

Uwe Düffert

22.07.2004 (German original)

February 19, 2006 (translation finished)



Artificial Intelligence Group  
Department of Computer Science  
Humboldt University Berlin

Committee: Prof. Dr. sc. Hans-Dieter Burkhard, Dipl.-Phys. Jan Hoffmann



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	RoboCup . . . . .	1
1.2	Robots and Field . . . . .	1
1.3	Motivation . . . . .	2
1.4	Aim . . . . .	3
1.5	Integration . . . . .	4
1.6	Terms . . . . .	4
<b>2</b>	<b>Modeling</b>	<b>7</b>
2.1	Aim . . . . .	7
2.1.1	Features of Walking . . . . .	7
2.1.2	Linearity of Activation . . . . .	8
2.2	Kinematics . . . . .	8
2.2.1	Forward Kinematics . . . . .	8
2.2.2	Inverse Kinematics . . . . .	10
2.2.3	Anatomy-related Corrections . . . . .	12
2.3	A Simple Walk Model . . . . .	13
2.3.1	Standing . . . . .	14
2.3.2	Moving the Center of Gravity . . . . .	14
2.3.3	Step Calculation . . . . .	15
2.3.4	Statically vs. Dynamically Stable Walking . . . . .	15
2.3.5	Parameters . . . . .	16
2.4	Extended Walk Model . . . . .	16
2.4.1	Improvements . . . . .	17
2.4.2	Wheel Model . . . . .	18
2.4.3	Parameters . . . . .	20
2.4.4	Choosing and Interpolating Parameter Sets . . . . .	21
2.4.5	Parameter Determination . . . . .	24
<b>3</b>	<b>Optimization</b>	<b>25</b>
3.1	Optimization Methods . . . . .	25
3.1.1	Calculability of an Optimal Walking Pattern . . . . .	25
3.1.2	Walking Patterns Inspired by Nature . . . . .	25
3.1.3	Approach . . . . .	26
3.2	Localization . . . . .	27
3.2.1	Localization Guide . . . . .	28
3.2.2	Position Calculation . . . . .	29
3.2.3	Smoothing . . . . .	30
3.2.4	Position Accuracy . . . . .	31

3.3	Omnidirectional Optimization . . . . .	32
3.3.1	The Path . . . . .	33
3.3.2	Fitness Function and Quality Measure . . . . .	35
3.3.3	Optimization Prerequisites . . . . .	37
3.3.4	Learning Method . . . . .	39
3.3.5	Crossing, Mutation and Population . . . . .	41
3.3.6	Results of Evolution . . . . .	41
3.4	Measurement of Different Parameter Sets . . . . .	44
3.4.1	Speed Measurement . . . . .	44
3.4.2	Turn Measurement . . . . .	45
3.4.3	Parameter Set Measurement . . . . .	46
3.5	Calibration . . . . .	51
3.5.1	Calibration Result . . . . .	53
3.6	Result of Optimization . . . . .	55
<b>4</b>	<b>Summary and Outlook</b>	<b>57</b>
<b>A</b>	<b>Bibliography</b>	<b>61</b>
<b>B</b>	<b>Note of Thanks</b>	<b>65</b>

# 1 Introduction

In this thesis a customary four-legged robot is taught to walk. Being able to walk with legs is one major prerequisite for the acceptance and usefulness of robots in human surroundings. On one hand such robots have to be suitably constructed for walking, on the other hand a given construction does not determine how a good walking with that robot will look like. Therefore the way to a competitive walking is shown with a concrete example here.

## 1.1 RoboCup

The RoboCup is an international scientific organization with the aim to support the development of robotics and Artificial Intelligence (see [20, 34, 2]). Since 1997 a RoboCup world championship is held once a year to evaluate the current research results in robot soccer matches. In these matches teams of robots and software agents<sup>1</sup> compete against each other. There are different leagues to allow several research topics and guaranty fairness nevertheless. Besides a couple of two-legged demonstrations there is only one league for legged robots at the moment. In that league four-legged robots built by Sony are used. It is the only league where teams are not allowed to modify the hardware.

The Humboldt University Berlin, represented by the chair of Artificial Intelligence of the computer science department takes part as team (AiboTeamHumboldt, former Humboldt Heroes, see [1]) in the Sony league of RoboCup since 1999. In international events the AiboTeamHumboldt is part of the GermanTeam<sup>2</sup> [10, 31] that it initiated in 2001. The GermanTeam became RoboCup world champion in its league in 2004.

## 1.2 Robots and Field

Robots of the model Sony Aibo ERS-110 (Figure 1.1 a) were used in the Sony league up to the end of the year 2000 and were replaced by the model Aibo ERS-210(A) (see Figure 1.1 b) in early 2001 for the following three years. Since 2004 the model Aibo ERS-7 (Figure 1.1 c) is also permitted. All these models have four legs with three motors each and a movable head with a color camera. Numerous additional sensors as well as all components needed for autonomous behavior (processor, main memory, removable memory, accumulator) are already integrated. The hardware differences between robot models result in different values for certain parameters, but the same approach and modeling can be applied to all of them.

---

<sup>1</sup>computer programs that autonomously fulfill an assigned task

<sup>2</sup>German national team of RoboCups Sony league consisting of Humboldt University Berlin, University Bremen, Technical University Darmstadt, and University Dortmund



**Figure 1.1:** The robots used, partially with tricots, all in typical posture: a) ERS-110, b) ERS-210, c) ERS-7

An ERS-210 will be used in this thesis for theory, basics and first results. The resulting methods and parameters will be adapted to ERS-7 and modified to achieve good results for both currently used robot models in the end.

The soccer field used in Sony Four Legged League (see Figure 1.2) is modified from year to year. On one hand, the matches shall remain interesting, but on the other hand unrealistic simplifications will be removed step by step, e. g. by increasing the field size or by removing single landmarks (color coded orientation marks) or the outer border.

One of the few unchanged characteristics of the field that is especially interesting for walking is the material of the underground – a green carpet. Such a carpet was also used in all experiments done for this thesis, because the walking of the robots pretty much depends on the underground. Using a different underground material would give different results for the same approach and will not be investigated further in this thesis.

### 1.3 Motivation

First of all, before thinking about aspects such as perception or Artificial Intelligence, it has to be settled, how such a robot will move, i. e. how locomotion can be realized with four legs containing three motors each. Without that knowledge it is impossible to implement a behavior control, because the possible actions of the robot are not yet



*Figure 1.2: The (still) current soccer field of the RoboCup Four Legged League, here at the RoboCup world championship 2002 in Fukuoka*

known. In addition it is hard to give a robot a robust perception of its environment without knowing anything about expected speeds or vibrations of the robot.

Therefore it has to be examined how a robot can keep its balance and avoid falling down, how it can react on changing walking speed and direction requests as flexible as possible, which is the best walking pattern and which optimization methods can be used successfully.

Besides other researchers all teams participating in RoboCup leagues with multi-legged robots have to face these problems. Therefore a number of papers concerning the topic already exists, e. g. on omnidirectional walking (mixing forward speed, sideways speed and turning speed in any proportion) [14], on fast walking forward [5] or on automated optimization [21]. As many of those partially contrary concepts as possible shall be combined in this thesis.

## 1.4 Aim

The aim here is the optimization of walking for given four-legged robots on largely known outer conditions such as ground qualities or robot dimensions. The main idea is not to understand why a certain walking is better than another, but to show a way to find such walking patterns nevertheless. Therefore the experiences made in the last years as well as the calculation and optimization methods used for it are shown. So the main focus is on practical translation into action and all data and measurements in the following chapters come from real robots accordingly.

Simulations of the robots and its movements were completely omitted because a sufficiently exact simulation for the robots used simply does not exist at the moment.

Although much time was invested in the creation of suitable software for the robots, there will be no source code excerpts in this thesis as that would not help to comprehend the problems dealt with. The complete source code can be found at <http://www.uwe-dueffert.de/code/>.

## 1.5 Integration

The software created for this thesis was integrated into the software of the GermanTeam [4, 7, 33] at all times and uses the GermanTeam architecture. This architecture [31] divides playing soccer (or any other complex task) into separate subtasks, called modules.

For the optimization of walking not only the module for generating walking patterns (*WalkingEngine*) but also the modules *ImageProcessor*, *SelfLocator*, *BehaviorControl*, and *HeadControl* are needed and will be described in more detail when it is appropriate.

The integration into the GermanTeam software guarantees that the methods used are fit for practice, reusable, and keep up with current developments (e. g. new robot models in Sony Legged League). Therefore the considerable additional expense for integration was advisedly accepted.

## 1.6 Terms

### Stability

Aim of this thesis is always the creation of fast, stable walking. Walking is stable if the robot does not overbalance and does not make a false step, but instead it largely and reproducibly happens what was intended by the activation.

Two kinds of stability can be distinguished. Static stability means, that a movement can be stopped at any time without forcing the robot to topple, because the perpendicular of its center of gravity is always above the polygon formed by its ground touching feet (after [27]). In contrast to that dynamic stability requires, that the robot does not topple within an uninterrupted movement.

### Activation

The behavior control module generates a series of motion requests resulting in the activation of leg joint motors. This activation fluctuates quiet much in practice (e. g. when playing soccer) to react on the rapidly changing environment. Therefore it is not sufficient to optimize only constant special cases like fast walking forward without taking other movements requested by a behavior control and the transitions to them into account. There is a significant difference between the maximum speed reachable with a constant walk request and the maximum speed with permanent direction and orientation corrections to make the robot reach a certain target or keep a certain path.

### Omnidirectionality

Omnidirectionality is the ability to walk into any direction, i. e. mixing forward speed, sideways speed and turning speed in any proportion. As the used robots are able to do that in principle, it is desirable to make use of this ability in higher behavior control layers.

## Odometry

For a given activation the robot should even blindly have an idea<sup>3</sup> about how it will actually move because of this activation. The ability to infer the covered distance only from own joint movements or foot movements is called odometry. Ideally the relocations requested by activation, actually reached and calculated by odometry are identical. In most cases correction parameters that minimize the occurring differences between calculated (expected) and actually measured velocities are required to come close to that ideal.

## Walk Parameters and Kinematics

Besides such correction parameters a walking motion always has numerous additional parameters such as step length or shoulder height that influence the series of foot positions. Foot positions and leg joint angles leading to these foot positions can be converted into one another with kinematics, the science of the geometric description of motions without taking forces into account.

Forward kinematics calculates foot positions from joint angles whereas inverse kinematics calculates joint angles from foot positions. Good values for all walk parameters used have to be found before the final joint angles can be calculated with inverse kinematics. That can be done by calculating optimal parameters or by purposeful trials in combination with proper learning methods.

## PID-Controller

Before the activation signal for a requested leg joint angle reaches the corresponding motor, it runs through an PID controller. This controller is responsible for deriving an appropriate motor activation from current (actual) joint angle and requested (target) joint angle. For these purposes it has three parameters:  $P$  (proportional) with high activation in case of high difference between actual and target value,  $I$  (integral) with high activation in case of long enduring difference, and  $D$  (differential) with high activation in case of an increasing difference.

## Walking Pattern, Duty Factor and Full Step

Walking patterns or gaits for robots with multiple legs like trot or gallop can be differentiated according to the times of lifting and lowering single legs. Therefore walking patterns can be characterized by the average percentage of time a single foot has contact to the ground, or in other words: by the average percentage of feet being on the ground at the same time. This share is also known as duty factor. Trot for example alternately keeps two legs on the ground and the others in the air, so the duty factor of trot is 0.5.

A full step is one step with each leg. A constant walking motion repeats after each full step accordingly, so a walking pattern is completely characterized by specifying one full step.

---

<sup>3</sup>the robot needs a model of its position in space that can be updated even without visual information

## **Calibration**

The equalization of requested speed and really measured speed by modifying the activation is called calibration in this thesis. Requesting diagonal walking for example might result in a robot walking to the side slower than expected and slower than straight on. In this case the activated sideways speed will be increased for the given walk request. After that the activated forward and sideways speed do not match anymore, but the resulting movement matches the originally requested one.

## **Unities and Coordinates**

All distances in this thesis will be given in millimeters, velocities in millimeters per second, angles in radians, and angular velocities in radians per second, if no other unit is explicitly mentioned.

In three-dimensional coordinate systems the x-axis points forward, the y-axis to the left, and the z-axis upward.

## 2 Modeling

### 2.1 Aim

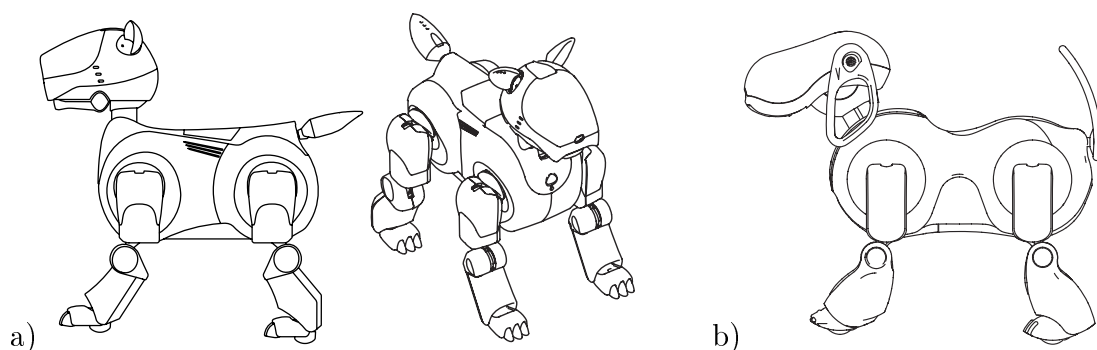
A walk model that can generate a series of joint angles for all involved joints from a given motion request is required to make robots like those in Figure 2.1 walk with their legs. Such a model as well as its development will be presented in this chapter.

#### 2.1.1 Features of Walking

Walking speed is a crucial criterion in the domain of RoboCup. Teams whose robots walk significantly slower than those of the fastest teams have a disadvantage that can hardly be compensated. Among the teams with sufficient walking speed other characteristics of walking become more interesting and even match-deciding.

An omnidirectional walking that is invariant to the speed of activation change is e. g. desirable to allow a behavior control to react on a rapidly changing environment as fast as possible without impairing the stability of walking. Previous attempts of automatic walk parameter optimization only concentrated on maximum walking speed [5, 21] or on additional features such as tolerance to different underground materials that are not yet relevant in RoboCup [15].

Therefore the aim of this thesis is the creation of a competitive walk model that meets the requirements of RoboCup. Furthermore this model shall allow for the selection and usage of different walk parameters as well as for an objective valuation of the resulting walking.



*Figure 2.1: Schemata of Sony a) ERS-210 and b) ERS-7*

### 2.1.2 Linearity of Activation

Ideally the actually reached walking speed is proportional to the activation causing it. As far as I know all walk models used in the Sony legged league so far are based on this assumption. Unfortunately that is only a rough approximation in practice.

There are deviations, especially at the limits. Instead of walking very slow the robot will move too few or not at all because of friction and inertia. It will not reach the theoretically expected speed at full activation either because of the inertia of its parts. Therefore the assumption of a walking speed proportional to its activation results in weighing two disadvantages against each other. Either only the part with a linear correlation is usable or deviations between expected and real speed impairing the odometry while vision information is missing have to be accepted.

So a proportionality assumption simplifies modeling and calibration but always in combination with a loss to performance or quality of walking. Therefore a possibility to by-pass this assumption will be shown in section 2.4 allowing for local calibration and optimization of walking.

## 2.2 Kinematics

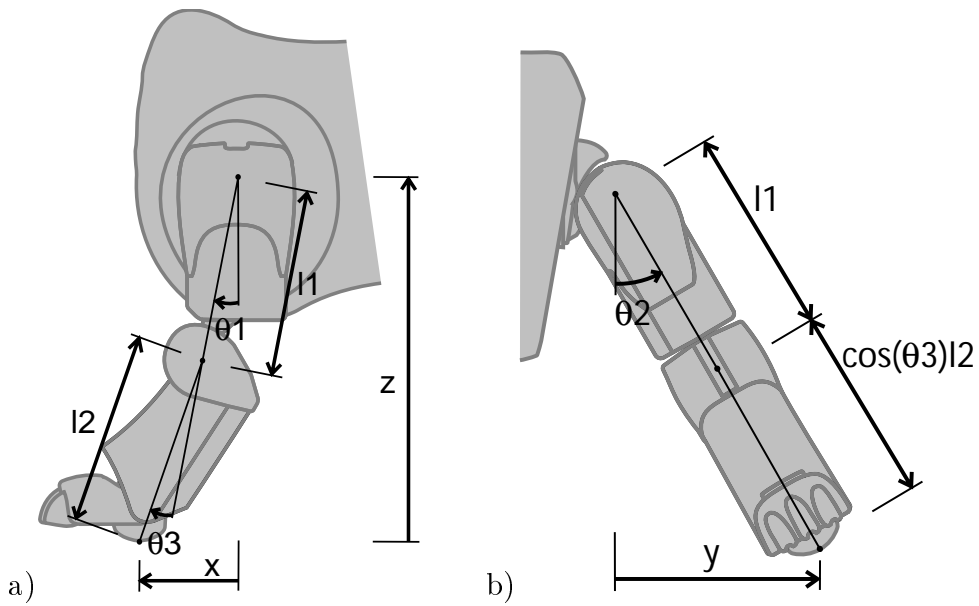
All walk models used in RoboCup and presented in this thesis later on describe the position of the feet relative to the robots body. After calculating the desired foot positions the joint angles leading to these foot positions have to be determined. This problem is handled by inverse kinematics.

The joints at or in the legs of a Sony Aibo allow for reaching all foot positions that are possibly interesting for walking motions. Therefore at least one solution exists for the leg joints to be calculated. In general, a non-linear equation system has to be solved for that, which is often only numerically possible and can result in several solutions. For the physique given by a Sony Aibo an analytic solution exists, and with adequate assumptions (explained when appropriate) this solution is even unique.

In section 2.2 only the left front leg and the coordinate system of its shoulder will be used, if not stated otherwise. Other legs require using different signs at certain points (because of symmetries) and another thigh length  $l_1$  for the rear legs, but the methods stay the same. In its own coordinate system the shoulder has the coordinates  $(0, 0, 0)$ , the leg has the desired coordinates  $(x, y, z)$ , the lower leg length is  $l_2$  and the joint angles are called  $\theta_1$ ,  $\theta_2$ , and  $\theta_3$  (as in Figure 2.2).

### 2.2.1 Forward Kinematics

But lets look at the underlying forward kinematics first. It calculates resulting foot positions from given joint angles. The foot position  $(x, y, z)$  relative to its shoulder results from an appropriate coordinate transformation of the local coordinate system of the foot into that of the shoulder. Based on the (simplified) anatomy of the robot this transformation consists of five single transformations:



**Figure 2.2:** Left front leg with extents and angles: **a)** side elevation for the calculation of knee joint angle  $\theta_3$ , **b)** front elevation for the calculation of shoulder joint angle  $\theta_2$

1. clockwise rotation about the  $y$ -axis by shoulder joint angle  $\theta_1$
2. counterclockwise rotation about the  $x$ -axis by shoulder joint angle  $\theta_2$
3. translation along the  $z$ -axis by an amount of (negative) thigh length  $-l_1$
4. clockwise rotation about the  $y$ -axis by knee joint angle  $\theta_3$
5. translation along the  $z$ -axis by an amount of (negative) lower leg length  $-l_2$

Let  $\text{Rot}_n(\alpha)$  denote a counterclockwise rotation by  $\alpha$  along the  $n$ -axis and  $\text{Trans}(v)$  a translation by vector  $v$  in the terms below. Then the foot position relative

to the shoulder can be calculated from the joint angles as follows (in homogeneous coordinates useful to simplify the concatenation of those five single transformations):

$$\begin{aligned}
 \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} &= \text{Rot}_y(-\theta_1) \cdot \text{Rot}_x(\theta_2) \cdot \text{Trans} \begin{pmatrix} 0 \\ 0 \\ -l_1 \end{pmatrix} \cdot \text{Rot}_y(-\theta_3) \cdot \text{Trans} \begin{pmatrix} 0 \\ 0 \\ -l_2 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\
 &= \begin{pmatrix} \cos(\theta_1) & 0 & -\sin(\theta_1) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta_1) & 0 & \cos(\theta_1) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_2) & -\sin(\theta_2) & 0 \\ 0 & \sin(\theta_2) & \cos(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \\
 &\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\theta_3) & 0 & -\sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta_3) & 0 & \cos(\theta_3) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -l_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\
 &= \begin{pmatrix} l_2 \cos(\theta_1) \sin(\theta_3) + l_2 \sin(\theta_1) \cos(\theta_2) \cos(\theta_3) + l_1 \sin(\theta_1) \cos(\theta_2) \\ l_1 \sin(\theta_2) + l_2 \sin(\theta_2) \cos(\theta_3) \\ l_2 \sin(\theta_1) \sin(\theta_3) - l_2 \cos(\theta_1) \cos(\theta_2) \cos(\theta_3) - l_1 \cos(\theta_1) \cos(\theta_2) \\ 1 \end{pmatrix} \cdot
 \end{aligned} \tag{2.1}$$

## 2.2.2 Inverse Kinematics

The task of inverse kinematics in this case is to solve the equation 2.1 for  $(\theta_1, \theta_2, \theta_3)$ , because we want to calculate those angles from a given  $(x, y, z)$ . The knowledge about the anatomy of the robot leads to an appropriate sequence of calculating the desired angles.

### 2.2.2.1 Calculation of Knee Joint Angle $\theta_3$

Fixing knee joint angle  $\theta_3$  restricts the possible foot positions to a sphere around the shoulder, so  $\theta_3$  only depends on the distance between shoulder joint and foot position. Therefore the knee joint angle can be calculated using the law of cosine as long as the distance between shoulder and foot is not longer than the leg length.

The angle included by thigh  $l_1$  and lower leg  $l_2$  (see Figure 2.2 a) is the difference between  $\theta_3$  and  $\pi$ , therefore it is

$$\begin{aligned}
 \cos(\pi - \theta_3) &= \frac{l_1^2 + l_2^2 - (x^2 + y^2 + z^2)}{2 l_1 l_2}, \text{ so that} \\
 \theta_3 &= \pi \pm \arccos \left( \frac{l_1^2 + l_2^2 - (x^2 + y^2 + z^2)}{2 l_1 l_2} \right) \\
 &= \mp \arccos \left( \frac{(x^2 + y^2 + z^2) - l_1^2 - l_2^2}{2 l_1 l_2} \right) \cdot
 \end{aligned}$$

Here inverse kinematics provides two solutions, because (at least in theory) two different knee positions can result in reaching the desired foot position. The knee joints

of the robots used can actually be bent into both directions, but considerably further into one of them. Assuming that knee joints are only bent into their preference direction, the remaining solution is:

$$\theta_3 = \arccos\left(\frac{x^2 + y^2 + z^2 - l_1^2 - l_2^2}{2l_1l_2}\right) . \quad (2.2)$$

### 2.2.2.2 Calculation of Shoulder Joint Angle $\theta_2$

Knowing knee joint angle  $\theta_3$  the shoulder joint angle  $\theta_2$  can be calculated, because now it only depends on the given desired sideways distance  $y$  of the foot (see Figure 2.2 b). Fixing  $\theta_2$  restricts the possible foot positions to a circle around the shoulder in a distance of  $y$  beside the robot. It is

$$y = \sin(\theta_2) \cdot (l_1 + l_2 \cdot \cos(\theta_3)) \quad , \text{ thus}$$

$$\theta_2 = \arcsin\left(\frac{y}{l_1 + l_2 \cos(\theta_3)}\right) \quad , \text{ because anatomy requires } |\theta_2| \leq \frac{\pi}{2}. \quad (2.3)$$

### 2.2.2.3 Calculation of Shoulder Joint Angle $\theta_1$

If the shoulder joint angle  $\theta_2$  is close to  $\pi/2$  and the knee joint angle  $\theta_3$  close to 0, the leg is stretched sideways and  $\theta_1$  is irrelevant and should be 0, because rotating a leg stretched sideways around its own axis does not change the foot position.

Otherwise the equations from 2.1 have to be solved for  $\theta_1$ . As  $\theta_2$  and  $\theta_3$  are already calculated, those equations can be simplified drastically by introducing the auxiliary variables  $a$ ,  $b$ ,  $d$ ,  $\beta$  (see Figure 2.3). Thus it is:<sup>1</sup>

$$a = l_2 \sin(\theta_3) \quad (2.4)$$

$$b = (l_1 + l_2 \cos(\theta_3)) \cos(\theta_2) \quad (2.5)$$

$$d = \sqrt{a^2 + b^2} \quad \text{and}$$

$$\beta = \arctan(b, a) \quad .$$

$a$  and  $b$  can also be described differently using  $d$  and  $\beta$  now:

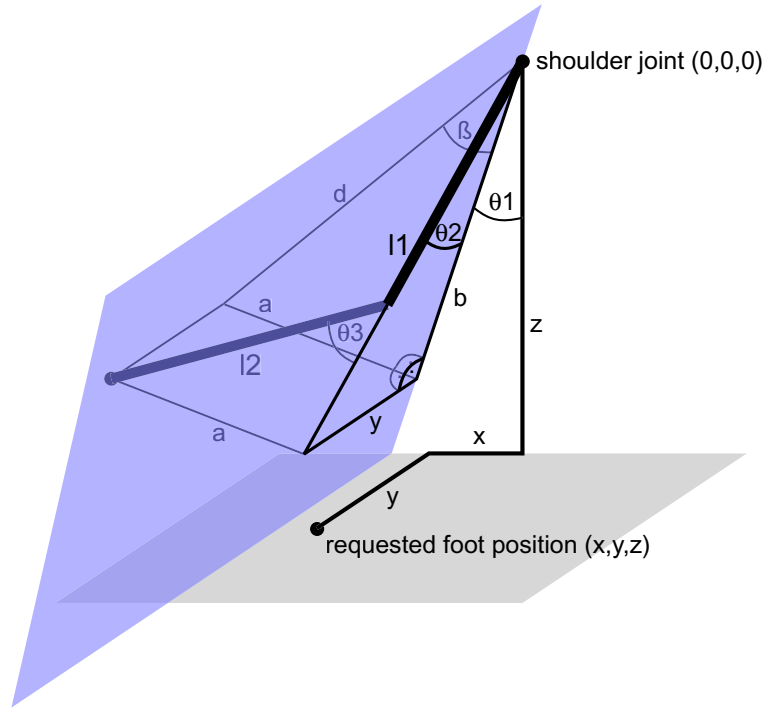
$$a = d \cos(\beta) \quad \text{and} \quad (2.6)$$

$$b = d \sin(\beta) \quad . \quad (2.7)$$

With these auxiliary variables it is:

$$\begin{aligned} x &= l_2 \cos(\theta_1) \sin(\theta_3) + l_2 \sin(\theta_1) \cos(\theta_2) \cos(\theta_3) + l_1 \sin(\theta_1) \cos(\theta_2) && \text{using 2.1} \\ &= a \cos(\theta_1) + b \sin(\theta_1) && \text{using 2.4 and 2.5} \\ &= d \cos(\theta_1) \cos(\beta) + d \sin(\theta_1) \sin(\beta) && \text{using 2.6 and 2.7} \\ &= d \cos(\theta_1 + \beta) && \text{addition theorem} \\ z &= d \sin(\theta_1 + \beta) \quad . && \text{analog} \end{aligned}$$

<sup>1</sup>In this thesis  $\arctan$  will always be used with the parameter order  $\arctan(x, y)$  (Mathematica syntax) and not with the parameter order  $\arctan(y, x)$  (C syntax).



**Figure 2.3:** Model of a robots leg for the calculation of shoulder joint angle  $\theta_1$  with all auxiliary variables

So we can finally calculate:

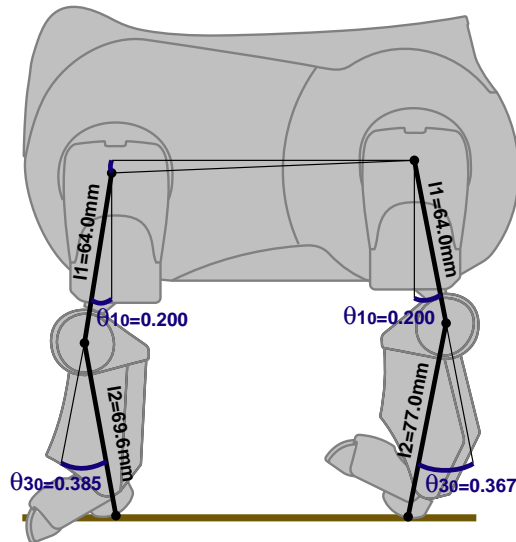
$$\begin{aligned} \theta_1 + \beta &= \arctan(z, x) \quad , \text{ and thus} \\ \theta_1 &= \arctan(z, x) - \beta \quad . \end{aligned} \quad (2.8)$$

Condensed, we get the following joint angles (except for the special cases *leg stretched sideways* and *foot position unreachable*):

$$\begin{aligned} \theta_3 &= \arccos\left(\frac{x^2 + y^2 + z^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \\ \theta_2 &= \arcsin\left(\frac{y}{l_1 + l_2 \cos(\theta_3)}\right) \\ \theta_1 &= \arctan(z, x) - \arctan\left(\left(l_1 + l_2 \cos(\theta_3)\right) \cos(\theta_2), l_2 \sin(\theta_3)\right) \quad . \end{aligned} \quad (2.9)$$

### 2.2.3 Anatomy-related Corrections

Previous calculations assume, that all three joint angles of a leg are 0 if the leg is completely stretched downward. Obviously that is the case for a simple wire frame, but the anatomy of a Sony Aibo somewhat diverges from this assumption. Here



**Figure 2.4:** ERS-210: Requesting an angle of 0 degrees at all leg joint motors results in vertically stretched outer shapes of the legs, but not in angles of 0 degrees between the joints.

requesting an angle of  $0^\circ$  at all joints results in vertically stretched legs, but the actual angles between the joints are not at all 0 (see Figure 2.4).

Pretty good walk parameters can be found even without correcting that difference, but the resulting walk is different from what the erroneous model predicted. Until short after the RoboCup world championship 2003 in Padova the GermanTeam was not even aware of that error, although the correction is easy.

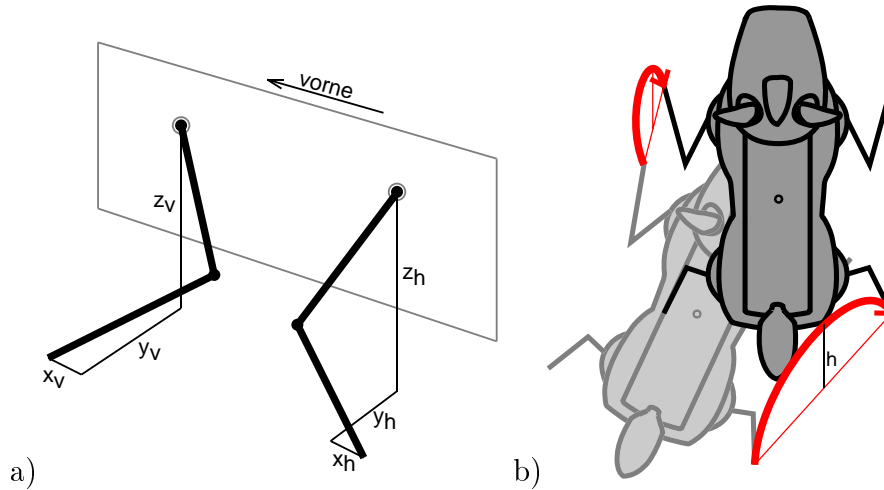
After correct calculation of the angles  $\theta_i$  between the joints the angles  $\psi_i$  between the leg parts can be deduced by adding constants (see Figure 2.4). As the robot expects those angles between leg parts as input, the following corrected angles have to be sent to the legs:

$$\begin{aligned}\psi_3 &= \theta_3 + \theta_{30} \\ \psi_2 &= \theta_2 \\ \psi_1 &= \theta_1 - \theta_{10} \quad .\end{aligned}$$

## 2.3 A Simple Walk Model

Each of the robots used (see Figure 2.1) has four legs with three active joints respectively, thus 12 stepper motors and therefore 12 degrees of freedom. Every 8 milliseconds (=1 frame<sup>2</sup>) a target angle has to be preset for each joint. The number of degrees of freedom has to be reduced to be able to generate a usable walking

<sup>2</sup>smallest unit of time in the communication with the robots hardware: receiving sensor data and sending joint angles is done in multiples of a frame



**Figure 2.5:** A first walk model and its parameters: **a)** The side view contains all spatial parameter. **b)** The top view shows, that moving the center of gravity of the robot implies the target foot position of a step that is carried out in a half ellipse.

motion under this conditions. Appropriate assumption are necessary for that, and possible suchlike assumption will be presented below.

### 2.3.1 Standing

First the simplest case will be examined: The robot stands, so its feet reside in constant position relative to its body. The following assumptions will be made: All shoulder (or hip) joints should have the same height above ground, so the feet can be seen as hanging under the shoulder layer. Furthermore right and left legs shall be symmetric to each other. Therefore the (still free) parameters shown in Figure 2.5 a are sufficient to specify standing. The joint angles necessary to reach the foot positions resulting from choosing values for those parameters will be calculated using inverse kinematics from section 2.2.

### 2.3.2 Moving the Center of Gravity

Before moving single legs, we have to keep in mind, that the main aim is to move the whole robot. It shall shift its center of gravity and turn its direction a certain amount per unit of time, and that preferably uniformly and without too much vibration. So at first a requested walk and turning speed will only change the position of the center of gravity. The feet remaining on the ground have a known position relative to the constantly moving body at any point in time.

All joints of the respective legs can be calculated with inverse kinematics from that. The position relative to the robot body of the feet remaining on the ground does not only depend on the current motion request, but also on the movement of the center of gravity since last dropping those feet, and therefore on motion requests of the past.

### 2.3.3 Step Calculation

Lets examine a constant motion request. After a full step (a step with each leg) the robot shall stand at a new position but with the same leg posture as before. Therefore making a single step means setting a foot to the position were it should be after a full step with the same leg posture.

On the way to its new position the foot shall describe a half ellipse (see Figure 2.5b) above the ground. According to this method a foot will always be set to a new position that seem to be optimal for the current motion request.

During walking single steps start when the feet are far away from each other in walking direction: The feet just lowered are in front of their rest position, whereas the feet to be lifted are behind their rest position. An additional step of half length (and half duration) is needed to get from standing (all feet in their rest position) into a walking motion or vice versa.

To simplify matters, diagonally arranged legs will always be lifted and lowered together. At first go that proved to be sufficiently stable and significantly faster than setting all legs separately. This walking gait is know as *trot*. A four-legged robot having two feet in the air is not statically stable any more, it may keel over. In reality that does not happen during walking, because the lifted legs will be back on the ground early enough. The robot is dynamically stable.

This consideration determines the point in time when to lift certain legs: two legs will always be on the ground (*stance phase*) and two will be in the air (*swing phase*). One leg pair will instantly be lifted, the other pair half a full step later. Thus each leg is on the ground half of the time (*duty factor* 0.5) and in the air the other half.

### 2.3.4 Statically vs. Dynamically Stable Walking

The motion of a robot is called statically stable, if its center of gravity is always above the polygon formed by at least three contact points with the ground (see section 1.6), preventing the robot to keel over. There are applications where this is a useful requirement, it even may have to be a verifiable feature, e. g. for a robot transporting dangerous goods.

Within the soccer leagues of RoboCup this requirement turned out to be needless. That was already supposed during the first RoboCup world championship with four-legged Sony robots 1999 in Stockholm. The statically stable walking integrated ex works was slow and particularly seemed to be slower than necessary.

Thereupon all participating teams decided to implement an own walk model optimized for the needs of robot soccer. Just as most other teams AiboTeamHumboldt successfully set aside static stability. Instead the robots had to lift and lower diagonally arranged legs together, so that only two legs had ground contact at a time (trot, duty factor 0.5).

Only few teams further tried to make the most of the statically stable walking approach, especially the team LRP<sup>3</sup> [16] and in the beginning the team CMU<sup>4</sup> [23]

---

<sup>3</sup>team of the Laboratoire de Robotique de Paris, France

<sup>4</sup>team of the Carnegie Mellon University, Pittsburgh, USA

too. Both made good progress, but in the end their walking motions were not competitive. The high duty factor (0.75, always three out of four legs on the ground) limits the walking speed too much. The fastest known walking modelled as statically stable with a Sony Aibo ERS-210 is approximately half as fast as the fastest known (dynamically stable) walking [21].

Demanding static stability may even lead to unstable walking, because complying with the simple criterion does not ensure stability. A statically just stable walking may as well be dynamically unstable, e.g. if radial forces affect a robot walking a curve. Given erroneous sensors it is difficult to calculate the limits to comply with for dynamic stability exactly enough. So the expected advantage of static stability can completely vanish at a sufficiently high walking speed. And high walking speed is important in RoboCup.

### 2.3.5 Parameters

The following parameters (see Figure 2.5 too) proved to be sufficient to describe walking in first experiments. Their values have been chosen manually to get a walking motion that is as fast and stable as possible:

- $x_v=24$  mm: distance of the front feet to their shoulder in body direction
- $x_h=20$  mm: distance of the rear feet to their shoulder contrary to the body direction
- $y_v=17$  mm: sideways distance of the front feet to their shoulder
- $y_h=17$  mm: sideways distance of the rear feet to their shoulder
- $z_v=103$  mm: height of the shoulder above the front feet
- $z_h=104$  mm: height of the shoulder above the rear feet
- $h=25$  mm: height of lifting a leg during a step
- $T=128 \cdot 8$  ms=1024 ms: duration of a full step (one step with all four legs)

This first walk model was used by us, the AiboTeamHumboldt (former Humboldt Heroes), at the RoboCup world championship 2000 in Melbourne [36]. It enables an ERS-110 to walk at a speed of 140 mm/s in opposite to barely 100 mm/s of the walking integrated ex works.

## 2.4 Extended Walk Model

The walk model introduced in section 2.3 has a number of weaknesses. Some of them are caused by (over)simplifications, that were quite useful for a first approach. Others are based upon false estimations, that partially exist down to the present day within the league.

Therefore an extended walk model will be presented below, that allows for omnidirectional walking and avoids as much of the weaknesses revealed in the meantime as possible. Furthermore the results of other teams as well as own experiences of the last years were taken into account, to be at least competitive to all existing approaches.

### 2.4.1 Improvements

#### Dependency of Ideal Foot Positions on Desired Motion

Calculating new ideal foot (rest) positions for the legs used in the next half step<sup>5</sup> does not always yield good results. During the last step (with the other two legs) a different motion may have been requested. Therefore the feet remaining on the ground may stand good for the previous step but bad for the current one.

This is the main disadvantage of the first model from section 2.3, where a changing motion request can result in an unpredictable position of the diagonal leg pairs to each other leading to unstable walking. The “wheel model” [13] introduced by UNSW<sup>6</sup> in 2000 by-passes this disadvantage and therefore became widely accepted within the league. Consequently the wheel model will be used in this thesis too. Furthermore it will be explained why this model is as superior to the allegedly more precise one.

#### Upright Walking vs. Tilted Robot Body

The best inclination of the body for walking depends on the anatomy of the robot used. A Sony Aibo ERS-210 walks significantly faster (and even more stable), if it virtually walks on the elbows of its front legs. That was first demonstrated by the team UNSW [13] in 2000 in Melbourne (200 mm/s).

This walking style results in a comparatively strong inclination of the robot body to the front and was not taken into account in the model in section 2.3 yet. In that model the feet were moved relatively to the plane through the shoulders, i.e. not parallel to the ground when the robot is inclined. With additional parameters the inclinations of shoulder plane and foot trajectories can be separated.

In the meantime all teams with competitive walking (stable, >200 mm/s) let their Sony Aibos ERS-210 walk with a significantly lowered upper body, and that with up to 270 mm/s [5], 291 mm/s [21], 295 mm/s [30] or even 311 mm/s [32]. The fastest known upright walking is approximately half as fast. The anatomy of the successor ERS-7 additionally advantages walking on the front elbows by rounded lower legs and leads to further increment of the walking speed.

#### Local Optimization and Calibration

The simple walk model acted on the assumption, that the robot can walk into any direction with constant parameters (such as height of the shoulders or duration of

---

<sup>5</sup>half of a full step: complete step with two of the four legs, but not step with half length

<sup>6</sup>team of the University of New South Wales, Australia

a step). It turned out, that special tasks like fast walking straight on can better be performed with modified parameters. Therefore the extended walk model in this chapter will account for that to be able to improve e. g. forward walking without impairing backward walking.

Moreover no concept existed up to now to handle the deviations between theory and practice within the walk model. Global linear correction parameters introduced with hindsight were at least able to let the real maximum speed match the one calculated by the model for walking forward, sideways, backward and turning (separately!).

But walking diagonally, walking at medium speed or mixing walking and turning still resulted in significant deviations between theory and reality. To avoid that, the walk model in this chapter will be locally calibratable for the first time.

### 2.4.2 Wheel Model

Using a four-legged robot would normally imply trying to explicitly exploit the existence of legs, e. g. move the feet independent of each other to optimal positions calculated on certain criteria. Such a separate activation can be quite useful for special tasks like exploring unknown terrain.

But for normal walking it can even be obstructive, because it makes it much more difficult to ensure a proper position of the legs to each other in all cases. Therefore setting single legs can impair the stability of walking, as it could be seen using the walk model from section 2.3. Good walking often follows a certain scheme, that can be a Central Pattern Generator<sup>7</sup> [8] in the nature or the “wheel model” for an Sony Aibo, which was introduced in 2000 by UNSW [13] and repeatedly improved since that time.

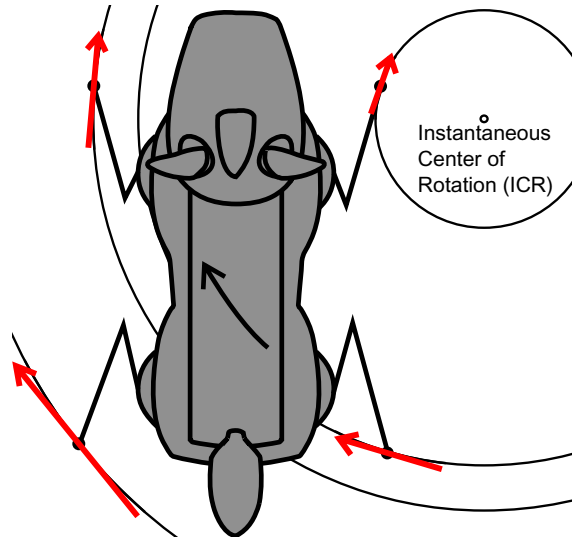
Each constant motion request consisting of a forward speed, a sideways speed and a turning speed results in the movement of the robot on a circle around a fix rotation center (*Instantaneous Center of Rotation*, ICR). Thus the rest positions of the feet of a robot also move on circles (with different diameters) around that rotation center. If the turning speed is 0, that rotation center is at infinity, so the rest positions of all feet move with the same speed into the same direction.

Imagining the feet as tiny wheels mounted at their rest positions, the speed and direction of every wheel is determined: the wheels have to follow their circles around the common center of rotation (see Figure 2.6). Instead of moving tiny wheels each foot has to make a step of proportional size into a suitable direction, i. e. tangential to the circle around the rotation center.

Using a wheel model, foot positions relatively to the robot body exclusively depend on the current motion request (and the temporal position within a full step). They are independent of old motion requests and old foot positions, therefore a consistent position of all legs to each other is guaranteed at any time.

---

<sup>7</sup>mechanism known from nature, where the movement of extremities is stimulated by a centrally generated pattern



**Figure 2.6:** Simultaneous walking and turning with the wheel model: The rest positions of the feet move on circles around a common rotation center. This is realized by steps (red) that are deduced from imagined wheel movements and tangential to the optimal circles.

This characteristic is an important difference to the first walk model from section 2.3. There all feet not involved in the current step remain on the ground at a constant position. So the position of those feet to their respective shoulder depends on the changing position of the center of gravity of the robot, i. e. not at all only on the current motion request, but also on the development of the motion request since last dropping those feet. Changing the current motion request may result in feet remaining at the ground at positions inappropriate for the current motion. Therefore walking can become unstable when the motion request changes rapidly. The robot may even make a false step, although the desired foot position was calculated precisely when dropping that feet.

The wheel model tolerates the incorrect calculation of single foot positions in favor of a consistent position of the legs at all times. The feet wont be set exactly onto the ideal circles, but instead onto a straight line tangential to that circles (see Figure 2.6). The movement of feet remaining on the ground relatively to its shoulder is always modelled as straight line, although it should be a circular arc in reality. Changing the current motion request will “correct” the position of the feet remaining on the ground relative to its shoulder as well.

All those effects can force the feet on the ground to leave their old positions, i. e. to slide, if this is required to get the legs into a consistent position to each other according to the current motion request.

What looks like a calculation mistake of the wheel model at first view, improves the stability of walking in the end and thus even increases the walking speed, because unwanted rolling does neither occur nor has to be prevented by artificially

limiting speed or acceleration. Therefore the wheel model significantly outmatches the allegedly more accurate calculation from section 2.3.

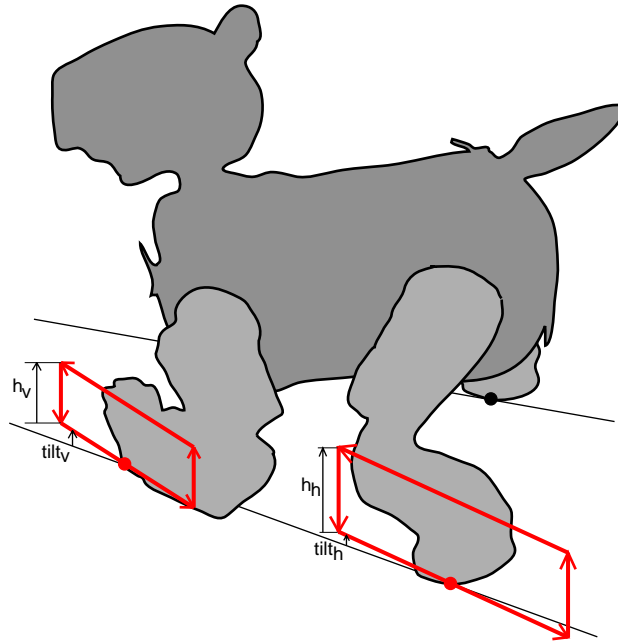
### 2.4.3 Parameters

In this thesis only those parameters will be used, that proved to have significant influence on the speed and stability of walking according to the experiences of the last years. Those parameters can be organized into four groups:

1. parameters specifying the rest positions of the feet relative to the body, as known from the first model used in the previous chapter:
  - $x_v, y_v, z_v$ : position of a front foot relative to the center between the front shoulders
  - $x_h, y_h, z_h$ : position of a rear foot relative to the center between the rear shoulders
2. parameters specifying the foot trajectories (see Figure 2.7):
  - $h_v, h_h$ : maximum height of a foot above ground during a step
  - $tilt_v, tilt_h$ : tangent of the angle between the (theoretical) foot trajectory and the ground; this inclination of the foot trajectory can be used to pinpoint how energetic the robot will put its feet down or keep it there to be well grounded and avoid sliding
3. parameters specifying the gait:
  - $gp_v, gp_h$ : fraction of a full step a front respectively rear foot is modelled to have ground contact
  - $l = (l_{vl}, l_{vr}, l_{hl}, l_{hr})$ : relative point in time of lifting each of the four legs, e. g.  $(0, 0.5, 0.5, 0)$  specifies the mostly used trot, where the left front and right rear leg is lifted at the beginning of a full step (0) and the other two legs half a full step later (0.5)
4.  $T$ : duration of a full step in frames à 8 milliseconds

In the past the introduction of additional parameters repeatedly proved to improve walking motions, e. g. by deliberately varying the body height during walking (*Canter Action*, see [14]) or allowing more complex foot trajectories than ellipses or parallelograms (*Free Form Quad*, see [5]). But on the one hand it was shown, that speeds near the maximum known walking speed can be reached without such additional parameters [21]. On the other hand, it is totally unclear in most cases, how those parameters can be used in omnidirectional walking and/or how they have to be adapted for that.

For example deliberately shifting weight perpendicular to the walking direction can be useful for walking forward (shifting from right to left) whereas it handicaps walking sideways (shifting from front to rear) because of different symmetries or weight distribution.



**Figure 2.7:** Parameters of the foot movements: feet are moved in parallelograms, the direction and size of which is determined by the wheel model.

Because of their unknown influence on omnidirectional walking a couple of additional parameters was deliberately omitted here, although it is known or assumable that they improve walking at least in isolated cases. This avoids a dramatic increasing of the complexity of optimization as well as the examination of erroneous assumptions on the influence of those additional parameters on omnidirectional walking.

#### 2.4.4 Choosing and Interpolating Parameter Sets

Up to now only a single parameter set and the parts of it were mentioned. But it was shown, that different parameter sets are optimal for different motion requests. For example the GermanTeam used one parameter set for forward walking and one for backward walking in 2003 and interpolated between those two when switching.

Therefore a model was in demand that reflects this circumstance as terminatory as possible. Moreover using several parameter sets has the advantage to be able to do without global correction parameters that are e.g. compensating unwanted turning when walking sideways. Instead additional parameter sets can be used for those motions that were only badly modelled or corrected before.

Based on these considerations a model was created, that uses 127 different parameter sets. One of those parameter sets contains (in addition to all previous parameters) information on the motion request it is optimized for and on the motion that actually has to be activated to realize the requested one. 127 parameter sets seem to be very many, especially considering that omnidirectional walking is possible quite well with only one parameter set. Nearly half of those parameter sets can be derived from the remaining ones using right-left-symmetries. Nevertheless using

127 parameter sets can be useful to improve clearness. The parameters are allowed to be all the same initially, except for the motion request they are “optimized” for.

Specific parameter sets can be optimized and calibrated afterwards. Symmetries will be taken into account thereby to avoid needless work. If it was already known in advance how the optimized parameter sets will differ, most of the parameter sets were superfluous and could be replaced by few simple correction parameters.

Unfortunately all previous assumptions into that direction proved to be either imprecise or limiting. Apparently the existing non-linearities enforce it (when using only a single parameter set) to find a compromise between average exactness and reachable maximum speed. Using several parameter sets instead allows independent calibration and optimization.

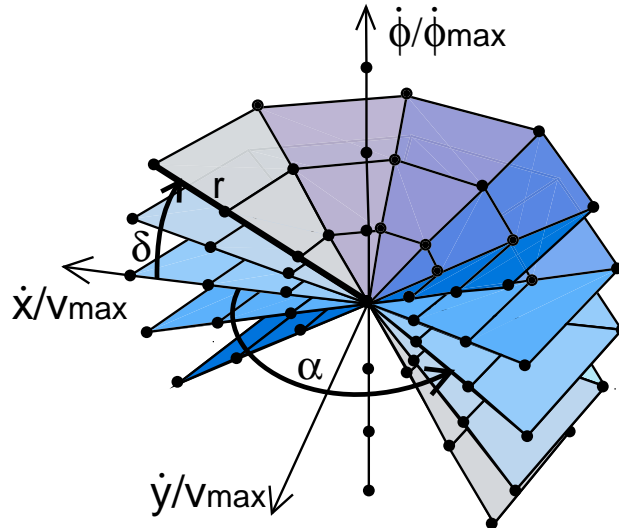
Motion requests usually consist of three components, namely forward, sideways and turning speed  $(\dot{x}, \dot{y}, \dot{\varphi})$ , that can be used to calculate the walking speed  $v = |(\dot{x}, \dot{y})|$ . For choosing a proper parameter set those three components are unfavorable, because a equidistant raster over them will divide interesting areas too coarsely, but uninteresting or unreachable areas (like walking and turning at maximum speed at the same time) too finely.

Instead of  $(\dot{x}, \dot{y}, \dot{\varphi})$  three other components will be used: walking direction, turn walk proportion and normalized overall speed  $(\alpha, \delta, r)$ , so to speak a transformation to polar coordinates. Using eight walking directions, three speed steps and seven turn walk proportions results in 127 reasonable combinations, because turning without walking does not need a walking direction (see Figure 2.8).

Choosing  $v_{max} = 300$  and  $\dot{\varphi}_{max} = 2,7$  as maximum speeds for normalization is geared to the speeds reached with an ERS-210 up to now, but that is not at all a fixed limit. Thus future optimizations are allowed to have normalized speeds higher than 1. The components used are:

$$\begin{aligned}
 \text{walking direction } \alpha &= \arctan(\dot{x}, \dot{y}) \\
 &\rightarrow \left[ -\pi, -\frac{3\pi}{4}, -\frac{\pi}{2}, -\frac{\pi}{4}, 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4} \right] \\
 \text{turn walk proportion } \delta &= \frac{2}{\pi} \arctan \left( \frac{v}{v_{max}}, \frac{\dot{\varphi}}{\dot{\varphi}_{max}} \right) \\
 &\rightarrow \left[ \underbrace{-1}_{\text{turn right}}, -\frac{3}{10}, -\frac{1}{10}, 0, \frac{1}{10}, \frac{3}{10}, \underbrace{1}_{\text{turn left}} \right] \\
 \text{overall speed } r &= \sqrt{\left( \frac{v}{v_{max}} \right)^2 + \left( \frac{\dot{\varphi}}{\dot{\varphi}_{max}} \right)^2} \\
 &\rightarrow [\text{slow, medium, fast}] \quad .
 \end{aligned}$$

Walking directions and turn walk proportions are fixed whereas the three speed steps have to be seen as relative, i. e. as *slow*, *medium* and *fastest*. That has a couple of advantages. It is possible to specify a separate maximum speed for each walking



**Figure 2.8:** The position of the 127 parameter sets used (black dots: standing,  $6\times$  turning only,  $8\times 5\times 3\times$  with walking): The azimuth  $\alpha$  denotes the walking direction, the declination  $\delta$  denotes the normalized turn walk proportion and the radius  $r$  the normalized overall speed.

direction and turning speed with a parameter set specialized for that. Secondly, all other combinations (those for not walking with maximum speed) can use the same parameter set and only the activation needs to be calibrated. Furthermore the three speed steps for a certain combination of walking direction and turn walk proportion can be chosen in a way that minimizes the average deviation between requested motion and actually executed motion over the complete speed range.

For a specific motion request the proper parameter set will be linearly interpolated from its (up to) eight neighboring out of the 127 parameter sets (right and left neighbor per spacial dimension  $\alpha$ ,  $\delta$  and  $r$ ). First the four neighboring interpolations for exact walking direction are calculated, out of those the two neighboring interpolations for exact turn walk proportion, and out of those finally the linear interpolation for the exact normalized overall speed.

Thus a continuous transition between all parameter sets used is ensured. So smoothly changing the requested motion results in a smooth change of the used parameter values, no matter which of the 127 parameter sets they are composed of.

Linear interpolation between the parameter values is used, because it does not make assumptions on the dependencies between single parameters, is easier to calculate and provides a more realistic (pessimistic) estimation of the performance of a mixture of two parameter sets, than e. g. an interpolation based on polar coordinates.

### **2.4.5 Parameter Determination**

The model deliberately uses only those parameters that proved to have significant influence on the quality and speed of walking according to the experiences of the last years. Nevertheless there are still two crucial open questions: which combinations of values for those parameters yield a competitive walking and how fast will this walking be in reality or what has to be activated to actually reach the requested speed. These questions will be dealt with in the next chapter.

## 3 Optimization

Often good and bad walking motions do not differ in the walk model used, but only in the choice of suitable parameters for the same model. The walk model introduced in the previous chapter has many parameters, although several additional parameters used by others teams were left out. Good values for a high number of parameters have to be found to be able to make use of the abilities of such a walk model. Therefore ways of determining better values for the parameters of a given walk model will be examined in this chapter.

### 3.1 Optimization Methods

Besides the time-consuming and inefficient manual trial and error of different parameter sets there are a couple of approaches that promise to deliver good parameter sets in less time. Among those approaches the ones will be chosen that seem to be appropriate for the specific case of optimizing the walking motions of an Aibo.

#### 3.1.1 Calculability of an Optimal Walking Pattern

There are several efforts to calculate a somehow optimal walking for Aibos or similar robots, e.g. [12, 11], that continuously yield better results. Unfortunately these results can not be transferred to real robots as expected. All walking motions completely generated by calculation significantly failed to reach their calculated maximum speed in reality or were at least inferior to walking motions manually or semi automatically optimized on real robots, i.e. did not reach their speed or stability.

Obviously there are still many unknown or badly modeled parameters for the target platform Sony Aibo. Qualitative statements can be derived from the calculation of a walking pattern, e.g. that a certain walk type such as trot looks promising. But quantitative statements like a good value for a certain parameter can be found much better on real hardware at the moment. For that reason optimization in this thesis will exclusively be performed on real robots in their target environment.

#### 3.1.2 Walking Patterns Inspired by Nature

Adopting walking patterns or walk parameters from the nature does only yield good results if the belonging anatomy is very similar to the archetypes one too, e.g. as in [29, 17, 25, 28]. Optimal walking for an existing robot like Sony Aibo may look totally different from that of a creature with comparable proportions, because numerous anatomical details are quite different.

The most obvious consequence of those differences is certainly walking on the front elbows, that looks curious to the uninitiated visitor, because it does not seem to fit

to the robots proportions. But for an Aibo that is much more stable and efficient than walking with more outstretched legs. There are even archetypes in the nature, namely rat-like animals living in Australia, that are a good part smaller than an Aibo, but have similar proportions and can move pretty fast with a lowered upper part of the body.

There are many research activities with the aim to understand the walking mechanisms of biological role models by recreating reflexes and Central Pattern Generators [3, 24, 18, 19]. In opposite to certain body tilt angles or step lengths such generic functional principles can very well be transferred to existing robots. Unfortunately the the mechanics of a Sony Aibo seems to be too inertial for most reflexes (such as flexor reflex for collisions or vestibular reflexes for inclination changes). Moreover reflexes are especially useful in unknown terrain. Therefore this thesis will not deal with reflexes at all.

#### 3.1.3 Approach

Good values for the high number of parameters of the walk can neither be calculated close to reality nor just be taken over from nature nor simply be determined manually. Therefore the optimization of those parameters should be automated as far as possible using methods like genetic algorithms<sup>1</sup> or Reinforcement Learning<sup>2</sup>. For that purpose some preparations are necessary.

Already the first implementors of a really competitive walking with Sony Aibo ERS-210 [14] recognized, that much effort is needed for the preparation of a suitable test environment. Such efforts will be shown in section 3.2.

This is not the first attempt to teach a Sony Aibo better walking using genetic algorithms. Gregory S. Hornby [15] lacked the experiences of five years of (four-legged) RoboCup, thus his trend-setting results are not competitive in todays RoboCup community any more. Other approaches can not be automated well enough or do not use a practice-oriented quality criterion, e. g. only evaluate walking straight on at maximum speed [30, 21]. Here the emphasis is on omnidirectional walking.

Demanding to walk into any direction (i. e. omnidirectionally) is not that natural. Sony Aibos already proved to be able to walk omnidirectionally, therefore it is desirable to utilize that within higher behavior layers.

As a first step constant parameter values are sought (in section 3.3) that allow for fast and stable omnidirectional walking. A suitable starting point is the walk parameter set used by the GermanTeam up to now. At the end of this step there will be a parameter set that enables a robot to walk quite good into any direction, but not extraordinary good into any special.

It was shown, that special tasks such as exclusively walking forward can be executed much faster [21], but a walk parameter set highly optimized for that will per-

---

<sup>1</sup>heuristic optimization methods that usually optimize the parameters of a given approach using biologic evolution as role model

<sup>2</sup>machine learning method that continues the search for better values where it is most promising, e. g. into the direction of the best average improvement since the last step

form pretty badly walking diagonally or backwards. As a compromise, the German-Team used a dynamic cross-fading between two distinct parameter sets for forward and backward walking at the RoboCup world championship 2003 in Padova [33].

The walk model introduced in section 2.4 can choose from much more specialized parameter sets according to the requested motion. Therefore the aim of section 3.4 is to examine, which parameter set is the best for a certain motion request and shall be used for it accordingly. All parameter sets in question have to be surveyed, compared and (if required) optimized to be capable to decide that.

In section 3.5 the chosen parameter sets will finally be calibrated. That is done by modifying the activated speeds in a way, that the resulting speed actually matches the requested one as exact as possible.

## 3.2 Localization

The robots in the Sony Legged League of RoboCup usually use several color coded landmarks at the border of the soccer field for localization. The localization approach based on that and used by the GermanTeam and many other teams is not well suited for automated search for better walk parameters for a couple of reasons: The localization quality depends on the position of the robot, it is e. g. better close to a landmark than in the middle of the field. Furthermore it is very light dependent, because color coded landmarks have to be detected and distinguished from other colored objects in the environment as exact as possible.

Because of the resulting uncertainties a probabilistic approach (Monte Carlo Localization, see [9]) has proved to be appropriate. Such a method successfully avoids leaps in the calculated position, but is relatively inert on the other hand, so the modelled position will follow the actual one only delayed. Moreover a probabilistic method assumes, that knowing a probably position is better than knowing no position at all or a position averaged from two distinct possibilities. This is true for playing soccer but not very useful for evaluating a walk parameter set.

Finally a known odometry is required to update the current position if none or only inconsistent vision information is available. During the examination of a yet unknown walk parameter set the speed of the robot has to be calculated from its position, and not the other way around.

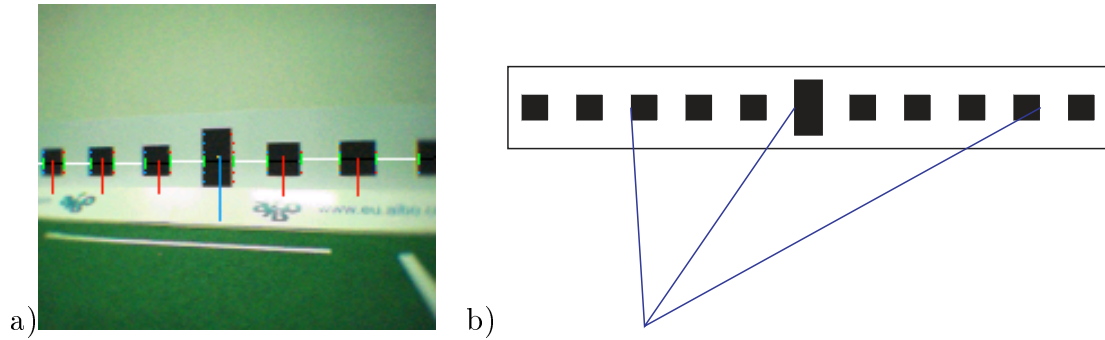
Several teams (UNSW<sup>3</sup>, Austin Villa<sup>4</sup>) just use two landmarks for localization during the optimization of walking straight on. Those landmarks already exist on every Sony league field and the robots can repeatedly walk from one to another and back. Thus only the position of the closer landmark has to be determined exactly.

This approach, first described by Hornby [15], is impressively simple and allows speed measurement fully sufficient for walking straight on. Unfortunately it is not well suited for the optimization of omnidirectional walking as it does not allow to measure the distance to an (imaginary) optimal line nor to distinguish between deviations caused by turning and those caused by walking sideways. Using additional

---

<sup>3</sup>University of New South Wales, Australia

<sup>4</sup>University of Texas at Austin, USA



**Figure 3.1:** The localization guide used: **a)** A picture from the camera of an ERS-210 with parts of the localization guide marked as recognized, **b)** Schematic view of the localization guide with possible angles for position calculation

markings like field lines does not solve this problem satisfactory, because those lines are distributed non-uniformly across the field and can easily be confused. Therefore the position of the robot can not always or not exactly be determined.

Frequently the usage of ceiling cameras is proposed for the determination of exact robot positions. But that is linked with a number of disadvantages like perspective distortions, external image processing, the necessity for a continuous radio contact<sup>5</sup> and additional acquisition costs. A high resolution camera, a wide-angle lens (depending on the available room), and usually additional markings on the robots are required for exact measurements.

Furthermore using a ceiling camera it is regrettably not relevant any more whether the movement to be examined impairs the perception of the robot e.g. by strong vibrations. Other external position determination possibilities like laser scanners were not short-listed for similar reasons.

### 3.2.1 Localization Guide

So all previous localization methods are inadequate for the determination of the yet unknown quality and maximum speeds of a walk parameter set intended for omnidirectional walking. Therefore the localization guide shown in Figure 3.1 was developed, that largely avoids the above-named problems. In opposite to Hornby [15] it does not required walk handicapping cables and neither needs external sensor like ceiling cameras or laser scanners nor a continuous radio contact handicapping walking as well.

The localization guide consists of eleven white DIN-A4 sheets with a black square glued to each of it. It is easy to transport and provides very high contrasts. Therefore it can be used in different locations without any calibration of the vision system, as long as it is sufficiently bright and no other objects with comparably high contrasts

<sup>5</sup>network traffic increases the processor utilization of the robots used and hence can result in interrupted movements

exist nearby. The middle black rectangle is 100 mm wide and 200 mm high, all other have a size of 100 mm×100 mm.

The aim of that construction is being able to know the exact real size of a part of the camera image as large as possible from different distances to the localization guide. Furthermore a one-to-one assignment between image and reality is desired. Out of that, the distance and direction of the camera to the center of the large middle black rectangle can be calculated as exact as possible. The aperture angle of the camera of the robots used allows this exact calculation in a distance of 60 cm to 260 cm, because there at least three quarters of the image width can be used for size determination. At a distance of 60 cm three of the black rectangles fit into the camera image. The whole localization guide covers at least 3/4 of the image width up to a distance of 260 cm.

The head control always tries to keep the bigger middle black rectangle in the middle of the camera image. The camera image will be searched for black/white transitions (small red and blue dots at the borders of black blocks in Figure 3.1 a) on sufficiently dense horizontal scan lines.

Between those black/white transitions the black blocks of the localization guide will be assumed. Therefore the upper and lower end of such a block will be searched on a vertical line in the middle of each neighboring pair of black/white and white/black transitions. Can those ends be found in a distance that fits to the width of the block, then the middle between the upper and the lower end of a black block is a candidate for the middle of that block.

After that a line will be laid through the block middle candidates using linear regression (white line in Figure 3.1 a). On that line the position of all transitions between white and black are determined with subpixel accuracy (such a position is the point with exactly the average of the brightnesses right and left of the transition) and checked for consistency (uniform distances).

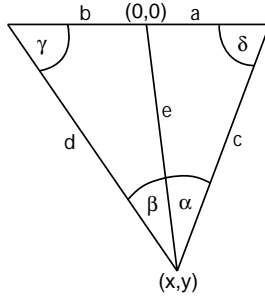
One transition in the middle of the image and the two outmost transitions are translated into angles to the camera and then associated with the exact positions on the real localization guide.

### 3.2.2 Position Calculation

In Figure 3.2 the lengths  $a$  and  $b$  of the seen parts of the localization guide between the black/white transitions are known as well as the belonging observed angles  $\alpha$  and  $\beta$ . Using the law of sines in both small triangles and equating the common line  $e$  results in two possible solutions for angle  $\delta$ :

$$\delta_1 = \arccos \left( \frac{|(2a + b) \sin(\beta) - b \sin(2\alpha + \beta)|}{\sqrt{2((a + b)(a + b - b \cos(2\alpha) - a \cos(2\beta)) + ab(\cos(2\alpha + 2\beta) - 1))}} \right)$$

$$\delta_2 = \arccos \left( \frac{-|(2a + b) \sin(\beta) - b \sin(2\alpha + \beta)|}{\sqrt{2((a + b)(a + b - b \cos(2\alpha) - a \cos(2\beta)) + ab(\cos(2\alpha + 2\beta) - 1))}} \right)$$



$$\gamma = \pi - \alpha - \beta - \delta$$

According to the law of sines it is:

$$\sin(\alpha)/a = \sin(\delta)/e$$

$$\sin(\beta)/b = \sin(\gamma)/e$$

**Figure 3.2:** Correlation between the angles used in position calculation based on the developed localization guide

Only one of those potential solutions yields the correct sums of internal angles in the triangles in Figure 3.2 and therefore this is the angle  $\delta$  in demand. The position  $(x, y)$  of the camera relatively to the medial of the three black/white transitions can now be calculated using the law of sines again:

$$\begin{aligned} c &= \frac{a \sin(\pi - \alpha - \delta)}{\sin(\alpha)} \\ x &= -c \sin(\delta) \\ y &= -a + c \cos(\delta) \end{aligned} .$$

Knowing the camera position  $(x, y)$  implies knowing the line of sight to the medial black/white transition. The robot can determine with what head joint angles and with what pixel in the image it saw that transition. Therefore the position and direction of the robot relatively to the localization guide can be calculated.

### 3.2.3 Smoothing

The robot raw positions obtained from single images of the localization guide are pretty exact in comparison to other localization approaches (see section 3.2.4). There are no noteworthy outliers, because the redundancy in the localization guide would leads to many discrepancies within the taken image for such an outlier. Therefore outliers already fail the consistency check (uniform block sizes and distances) and are discarded. Thus only two interesting types of errors remain.

One type of error are measurement errors of the head joint angles that increase with mechanical stress and yield a less precise position and direction determination. Walking that shakes the head heavily makes positioning more difficult, impairs the position accuracy of walking and will be rated worse thereupon. Thus the reason for these errors are bad parameter sets with a low fitness that will not be used any further. Therefore no additional action is necessary against such errors.

The second type of error is a slight noise that is e.g. caused by blurring (fix focus camera), sensor-related noise in the image and vibrations. Several methods to eliminate this kind of error were tried out.

Laying a quadratic curve through the last measured values to minimize the sum of squares of differences seems to be a convenient estimation of the current position at first glance. But looking at the following measured values reveals that the smoothing effect of this method is negligible.

Even a Kalman filter<sup>6</sup> (see [35]), that is intended to handle noisy data, only yields moderate results. It copes well with uniform situations (constantly good values or constantly high noise), but has troubles with the transitions between those situations as well as with direction changes of the robot, because the self-trained trust in measured values is suddenly wrong.

In the end a simple PID controller produced the best results. If  $m_i = (x_{i_m}, y_{i_m}, \varphi_{i_m})$  denotes the position measures from the image at time  $i$  and  $p_i = (x_{i_p}, y_{i_p}, \varphi_{i_p})$  the position modelled for the same time, then the following controller rule is used for smoothing:

$$p_i = p_{i-1} + \underbrace{0,15}_P \cdot (m_i - p_{i-1}) + \underbrace{0,25}_I \cdot \sum_{j=1}^i (m_j - p_{j-1}) \quad .$$

The effect of that PID controller without differential term (i. e. PI controller) can be seen in Figure 3.3 b.

### 3.2.4 Position Accuracy

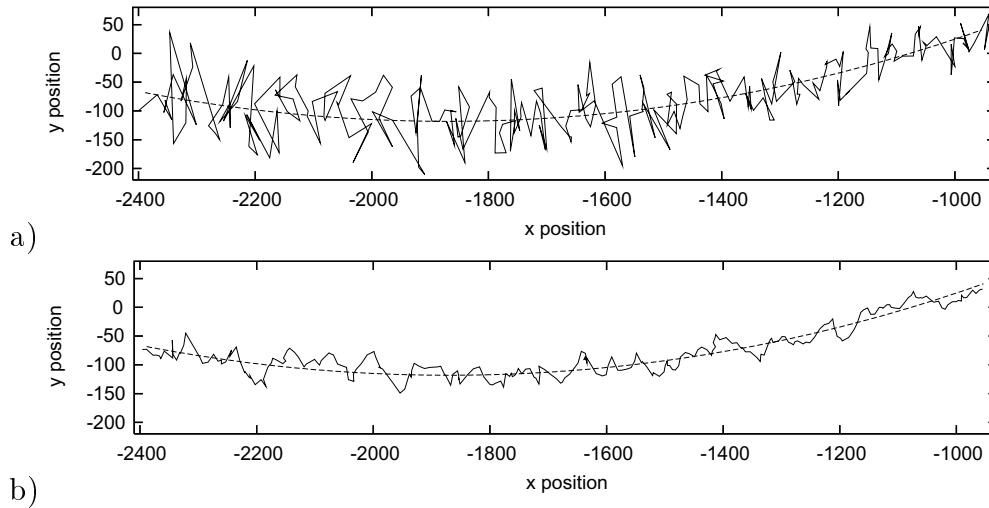
Knowing exact parameters of a robot like camera opening angle and joint angles allows accurate determination of the robot position. The systematic error has the same dimension as the manual measurement with tools, it is below 1 centimeter. Therefore errors caused by mismeasurements are more interesting.

The camera image of a Sony ERS-210 has a horizontal resolution of 176 pixels. An ERS-7 can provide images with more than twice that resolution, but those images do not have a homogeneous brightness distribution any more, because the doubling is reached by interpolation of measured values of three color channels. Thus only the images with lower (=normal) resolution will be used in this thesis.

At a distance of 60 cm to 260 cm to the localization guide the real size of objects seen in at least 3/4 of the image (132 pixels) is known. At least two angles are required for position calculations as in section 3.2.2, each corresponding to about 3/8 of the image width (64 pixels, angle about 0.38). Because of position calculation with subpixel accuracy (taking the point with medium gray tone between white and black area) the measurement error of the distance between two transitions will

---

<sup>6</sup>a filter that uses variances and covariances of previously measured values to derive a noise free prediction



**Figure 3.3:** robot position calculated based on the localization guide during ten seconds of walking forward with soft turning in comparison to the optimal position: **a)** raw data, **b)** the same with PID smoothing

probably not exceed  $1/2$  pixel (angle about  $0.003$ ). Therefore the maximum expected position deviation at a distance of  $2600$  mm will be:

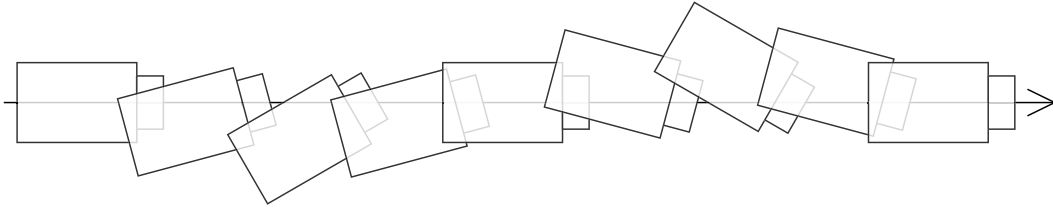
$$\begin{aligned} \alpha = 0,38 + 0,003, \quad \beta = 0,38 - 0,003 &\quad \rightarrow \Delta y = 56,6 \text{ mm} \\ \alpha = 0,38 + 0,0015, \quad \beta = 0,38 + 0,0015 &\quad \rightarrow \Delta x = 11,3 \text{ mm} \end{aligned} .$$

The position of an ERS-210 while walking with slight turning was measured for ten seconds (as shown in Figure 3.3) to verify these theoretical considerations. The square root of the mean square deviation between ideal position and measured position was  $15.9$  mm in x-direction (distance to the localization guide) and  $38.3$  mm in y-direction (parallel to the localization guide).

Using the PID controller introduced above the mean deviation in x-direction slightly increases to  $18.3$  mm, because the PID controlled value can take past measurements into account only. Thus the smoothed position has to follow the real position of the continuously advancing robot with some delay. Contrary to that the mean deviation of the y-position considerably decreases to  $14.6$  mm, because here basically the noise was eliminated. So using the introduced PID controller the significant smoothing is always accompanied with the fact, that the modelled position follows the measured only with some delay at higher speeds, but because of the integral term by far not as much as when using simple calculation of average.

### 3.3 Omnidirectional Optimization

Now, having an automatic method for accurate position determination of the robot, it is possible to search for good values for the parameters of the walk model introduced in section 2.4.



**Figure 3.4:** *The path for the examination of omnidirectional walking specifies the desired development of position and direction of a robot.*

In this section a single parameter set will be sought after, that allows for good omnidirectional walking, minimizing the need to switch to specialized parameter sets for certain motion requests. Such an omnidirectional parameter set will be a useful standard solution, that is expected to be more similar to a specialized parameter set than a parameter set specialized for something different. Therefore the interpolation used by the walk model from section 2.4 between an omnidirectional parameter set and one special for e. g. fast turning will probably look better than the interpolation between two specialists.

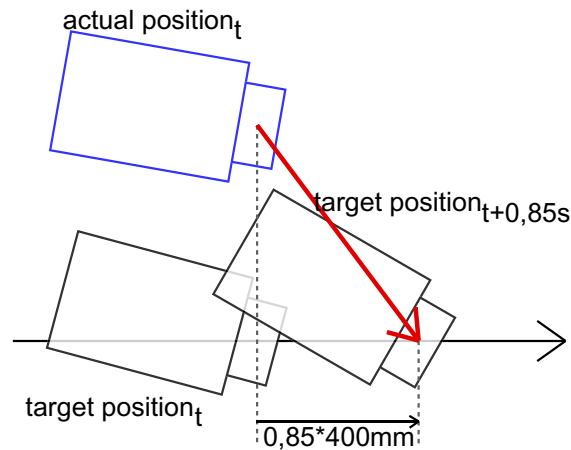
### 3.3.1 The Path

The robot has to follow a given path including different walk and turn directions as fast and as precisely as possible, because the activation in a soccer playing behavior will be comparable. Using such a path distinguishes the approach in this thesis from other methods, where walking is generally optimized for a constant direction of movement at a time. Choosing an adequate path allows to test a parameter set for the quality of omnidirectional walking it produces.

A path defines a target position and direction for every current position and future time. As the walking speed on that path shall not be predetermined but found out instead, the future target position depends on the last known actual position. The target position at the current time always has the same x-coordinate as the actual position and is used as progress indicator in main walking direction on the path. Target x-coordinates in the future result from assuming a speed of 300 mm/s (ERS-210) respectively 400 mm/s (ERS-7), that is slightly above the expected maximum speed of a robot on the path. A path defines the belonging y-coordinate and target turn angle for a given x-coordinate.

The path used here is divided into two parts. The robot has to vary its turn angle when walking forward (see Figure 3.4), namely sinusoidal with an amplitude of  $\pi/4$  and a period of 900 millimeters in main walking direction. Walking back is done straight instead. The forward walking part will be weighted twice as much, because it is the more important part for playing soccer. Walking backward is required to work too, but not necessarily with permanent direction changing.

A robot can only follow such a path if its walking in reality matches the motion intended by the activation to some degree. If the difference is too big, the attempt



**Figure 3.5:** Controlling of the activation of walking used to let the robot follow the path: The activated motion (red) results from the distance to the desired position 0.85 seconds in the future.

to stay on the path or return to it can even increase the deviation from the path. This happens for example when using the walk model without the anatomy-related correction from section 2.2.3. With this correction the robot can follow the predetermined path considerably better.

On the other side, following such a path requires an activation that is able to minimize the deviation between desired and actual position and follows the path as fast as possible at the same time. Such an activation is much closer to one used in practical cases like playing soccer than the usually measured constant motion requests.

Walking should be able to cope with different activations each considered to be useful. Therefore it can be demanded even from an yet not well known walking to be able to follow such a path including the activation control made to follow it, as long as the path does not require impossible movements (e. g. continuous direction jumps).

Thus the following control will be used for the activation of movements: Always the target position 0.85 seconds in the future is headed for, i. e. the activation always tries to correct the deviation between actual position and target within 0.85 seconds (see Figure 3.5). Correcting the turn angle (direction) would often be possible in less time, but trying to correct the position faster results in the robot not making headway any more, because it is completely busy with position correction. So the control used is a good compromise between the contradictory aims fast advance and minimal distance to the optimal path.

That control proved to be capable to let the robot follow the given path speedy even with a not yet measured walking. The control has significant influence on the evaluation of a certain walking. That is acceptable, because the control can be seen as integral part of the task to follow the belonging path. The control used favors

fast and exact walking, works well with the walk parameter sets examined so far, but does not ensure to evaluate totally different parameter sets fairly as well.

The path including its activation control introduced here will not only be used in this section, but additionally once again in section 3.6. So it acts as a benchmark showing that the optimization steps in section 3.3 to 3.5 improve walking.

The robot has to be able to deal with a number of special cases itself to allow fully automated optimization. Normally those cases would be detected and dealt with by a human trainer. Such cases are:

- **Battery low:** The robot has to end the training, save the last results and try to catch attention.
- **Timeout:** The current parameter set is that bad, that the robot was unable to return to the starting point in a certain time. Therefore it has to abort this trial and return to the starting point using known standard walk parameters.
- **Unexpected situations:** The robot has to stand up if it overthrew and has to start a searching behavior if it is lost or does not see the localization guide any more.

All those behaviors are specified in the same way as the soccer playing behavior developed by the GermanTeam, namely in a hierarchical state machine formalized in the XML dialect Xabsl<sup>7</sup> [26]. The state machine of the behavior used is shown in Figure 3.6.

### 3.3.2 Fitness Function and Quality Measure

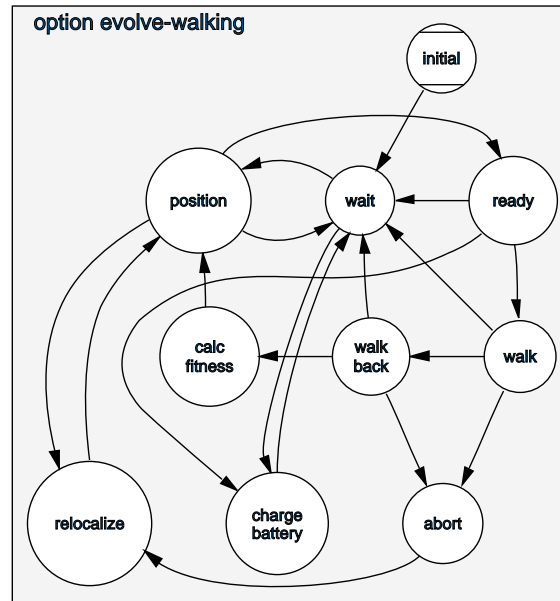
The aim of this thesis is still a method that usage yields walk parameters for a Sony Aibo suitable in RoboCup environment. The quality measure of a walk parameter set determined by a fitness function therefore does not only include maximum walking speed. The intensity of vibrations of the robot (and thus the disturbance of walking stability and image processing) as well as the ability to keep the requested direction are (explicitly and implicitly) taken into account too. The latter is reached by always heading for the target position on the path (see section 3.3.1), and that takes longer if the actual walking differs from the motion intended by that activation.

The resulting quality or fitness of walking can be seen as a valued speed. It is the average speed reached on the path and corrected by unwanted position deviations and vibrations. The following measurements will be used for that:

- $\dot{x}$ : average walking speed in x-direction (along the path, towards the localization guide or away from it respectively) in mm/s
- $\Delta y$ : averaged absolute value of the deviation of the y-position, i.e. average distance to the optimal path

---

<sup>7</sup>eXtensible Agent Behavior Specification Language



**Figure 3.6:** *Xabsl* state machine of the behavior for optimizing omnidirectional walking: The real work such as evolution of parameters or calculation of fitness is done within single states (circles).

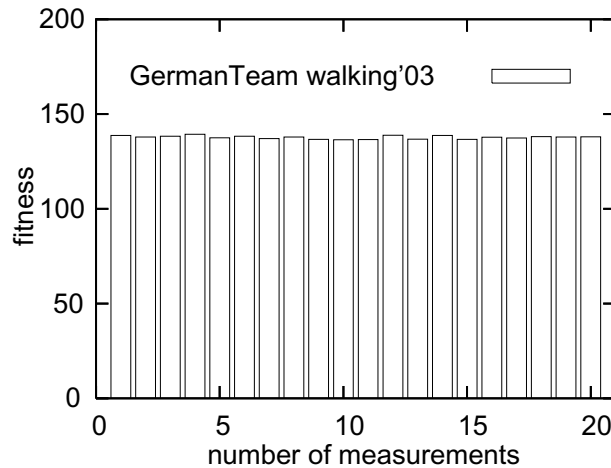
- $\Delta\varphi$ : averaged absolute value of the deviation of the robots direction, i. e. average distance to the optimal direction
- $\ddot{z}$ : averaged value of the acceleration in z-direction; this acceleration is unwanted and recognized as unpleasant stomping
- $p_{\text{blind}}$ : percentage of images the localization guide could not clearly be identified in; if  $p_{\text{blind}}$  is high the walk parameter set used obviously resulted in huge deviations between reality and activation or in unacceptable high vibrations

Other possible values were deliberately omitted, because they do not have any important influence onto the quality of walking, that is not already contained in the values used. Such additional values include accelerations into other directions like shifting of weight from right to left. The difference between activated and actually reached joint angles is used neither, because it is necessary to activate a little more than actually reachable for highest performance.

The five measurements used have to be weighted in a way that ensures appropriate proportions to each other. A walk parameter set unapt to turn could e. g. be used to follow the path very fast but largely ignoring the desired body direction. That shall not be awarded with a high fitness, because it is just not the aim to encourage walking straight on only. Therefore the turn angle deviation has to be penalized that much that turning less in favor of faster walking straight on does not yield a higher fitness.

fitness calculation	$\dot{x}$	$\Delta y$	$\Delta\varphi$	$\ddot{z}$	$p_{\text{blind}}$	fitness
good run	186,6	26,5	0,131	$1,22 \cdot 10^6$	0,049	
rating	186,6	-4,4	-4,3	-7,2	-2,0	= 168,7
bad run	159,0	37,9	0,154	$1,33 \cdot 10^6$	0,128	
rating	159,0	-6,3	-5,1	-8,3	-5,1	= 134,2

**Table 3.1:** Appropriate weighting of the components of the fitness functions allows to differentiate between good and bad parameter sets.



**Figure 3.7:** ERS-210: The results of 20 measurements of the fitness of the walking used by the GermanTeam in 2003 are very close to each other. Thus each parameter set will only be measured once from now on.

Out of those considerations the fitness  $F$  of a parameter set  $P$  was defined as follows (the belonging evaluation of two example runs is shown in table 3.1):

$$F(P) = \dot{x} - \Delta y/6 - 33\Delta\varphi - (10^{-5}\ddot{z} - 5) - 40p_{\text{blind}}$$

Firstly, this fitness function was used to evaluate the walking used by the GermanTeam in RoboCup 2003 repeatedly. The calculated fitness of all attempts was between 136.4 and 139.3 (average 137.7) with a standard deviation of  $\sigma = 0.84$  (see Figure 3.7). Because of that low fluctuations each parameter set will only be measured once from now on.

### 3.3.3 Optimization Prerequisites

The methods introduced above allow an objective evaluation of walking intended for omnidirectional use for the first time. Before starting the real optimization it is worthwhile to examine (and now being able to check), which changes are expected to increase the performance of walking. Subsequently it has to be ensured that the

### 3 Optimization

optimization method to be used is capable of performing and/or exploiting these changes.

The step length leading to maximum walking speed depends on the parameter set used as well as on the walking direction. Therefore the walk model from section 2.4 allows specifying different maximum speeds (and thus different maximum step sizes) for different directions. Unfortunately these maximum (useful) step sizes are not known for a parameter set not measured yet.

Limiting step sizes yields faster and more stable walking in comparison to fully exhausting the leg lengths. Therefore we will start using a walking direction independent step size limitation just as before the introduction of the walk model in section 2.4. This limitation has two parameters: maximum step size in forward (x-) direction and in sideways (y-) direction for all legs. Usually step sizes in between were limited by the ellipse

$$\sqrt{(x/x_{max})^2 + (y/y_{max})^2} \leq 1 \quad .$$

Allowing larger diagonal steps by using the less restrictive

$$\sqrt[4]{(x/x_{max})^4 + (y/y_{max})^4} \leq 1$$

yields a three percent higher fitness (see Figure 3.8) whereas increasing the maximum step size in x or y direction decreases the fitness. Therefore the less restrictive limitation will be used as long as a parameter set and its maximum reachable speeds are not measured. A further reduction of step size limitations does not improve the walking speed any more.

The maximum reachable size is limited by the anatomy of the robot and nearly exhausted, as e. g. touching knees of front and rear legs during walking demonstrate.

A promising candidate for optimization is the step frequency. The duration of a full step used by the GermanTeam for example decreased from 1.02 seconds in 2000 to 0.64 seconds at the world championship in 2003. Further increment of the step frequency without accompanying actions does not increase the walking speed of an ERS-210 any more.

Apparently the robot is unable to lift the feet early or fast enough and starts to slide like on ice instead of moving forward. Besides the anatomy of the robot and the inertia of its parts the foot trajectory used is a reason for that too. Using half ellipses as in the first walk model (see Figure 2.5 b) results in a smoothly looking walking. But the robot seems to lift and lower its feet slower than favorable, especially if the steps are significantly longer than high.

For that reason it is at least easier to find good parameter sets with the feet moving in rectangles or parallelograms. Thus the current walk model (see section 2.4) uses parallelograms too. Nevertheless the question for the optimum foot trajectory is not answered yet.

(P, I, D) values	shoulder joint $\theta_1$	shoulder joint $\theta_2$	knee joint $\theta_3$
ERS-210 standard	(22, 4, 8)	(20, 4, 6)	(35, 4, 5)
ERS-7 standard	(28, 8, 1)	(20, 4, 1)	(28, 8, 1)
P increased	(38, 8, 1)	(30, 4, 1)	(40, 8, 1)

**Table 3.2:** Values used for the PID controller of the joint motors

Another reason for the inertness of the legs at higher speeds is the up to now exclusive usage of standard values for the PID controllers of the joint motors. These standard values are a good compromise between accuracy and speed and successfully avoid overshooting over the target value.

Increasing those values, especially the interesting proportional term, will result in overshooting or trembling in certain situations (unstressed legs, large jumps in the activation). In contrast to that fast walking (fast but continuously changing target values, burdened legs) benefits from increased PID values. The benefit is actually higher than expected: Increasing the P value without changing anything else increases the fitness by about ten percent (see Figure 3.8).

Nevertheless, the PID values will not be changed by evolution, but manually increased instead as far as stability and trembling allow it and an improvement of the fitness is measurable. On the one hand this protects the motors against overstress and on the other hand it avoids the necessity to find a measure for stability and trembling that can be determined by the robot. Table 3.2 contains the different PID values used.

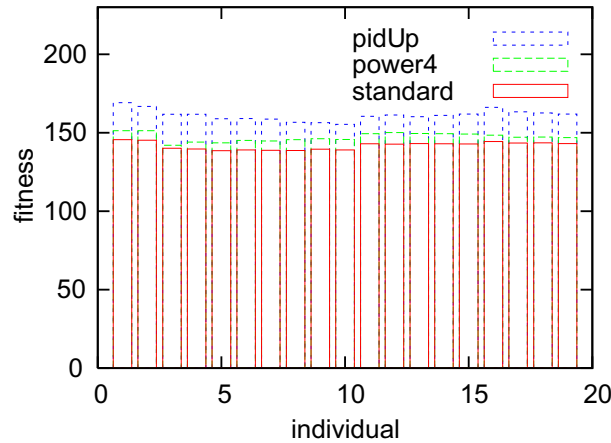
The comparison of walking of an ERS-210 and an ERS-7 with the same walk parameters originally developed for an ERS-210 and the respective standard PID values shows, that an ERS-7 can not execute single extreme walking motions that still worked using an ERS-210. The ERS-7 can not lift its feet fast enough for those motions. For the same reason the standard PID values of an ERS-7 lead to motions that look smoother than on an ERS-210.

A PID controller is intended to correct the difference between desired and actual joint angles. The standard low proportional term causes the motor to do few against small deviations. Therefore trying to walk pretty slow results in stepping on the spot. Large deviations will be corrected stronger, but because of the increasing resistance of inertia not fast enough.

Thus increasing the proportional term of the PID values mainly improves very slow and very fast walking. Negative effects only occur using improper parameter combinations, e.g. very fast movements at very slow speed (tip-toeing), and are recognized as uneasier walking (more trembling).

### 3.3.4 Learning Method

On the basis of the experiences of manually fine-tuning walk parameter sets during the last years it can be assumed, that the exclusive usage of simple (e.g. gradient based) methods is unsuitable because of the often unexpected correlation between



**Figure 3.8:** *ERS-210: The measured fitness of walking is increased by three percent due to less restrictive step size limits (“power4”). Raising the proportional parameter of the PID controller of the joint motors even increases the fitness by ten percent. The fitness values of several walk parameter sets with standard PID values and with increased proportional value are shown.*

changes of parameters and changes of the resulting walking. For that reason one possibility to get better parameter combinations is the largely blind mutation and crossing of appropriate parameter sets. Other authors share the opinion that the usage of evolutionary methods is a convenient way to solve such problems [37].

In [5] and especially in [21] was shown, that Reinforcement Learning can direct the examination of a large search space into a promising direction in spite of using random mutation, at least as long as there is a simple optimization criteria such as maximum speed of a constant motion.

The acceleration of the optimization process that can be expected from Reinforcement Learning here too is achieved by making additional implicit assumptions on the influence of single parameters onto the overall fitness. Just like with other gradient based methods the assumption that an optimum can be found in the direction of the locally best fitness change, makes it more difficult to leave local optima and impossible to reach parameter combinations besides the Reinforcement way. Therefore the additional effort of using Reinforcement Learning was avoided in this thesis.

Another possibility to accelerate the search is found in [6]. There a neural network is used, that proposes only those parameter sets for a test on real robots, whose fitness can not be estimated well enough. Whereas that simplifies the course examination of the whole search space, it obstruct the search for optima near known parameter sets.

All known good walking motions look relatively similar (regarding parameters like shoulder height, body tilt and step frequency) for a fairly long time, but they are continuously accelerated nevertheless. Therefore improvements near existing parameter sets are more likely than at the far end of the search space.

Consequently a simple, largely classic evolution method (see [22] too) will be used in the experiments described in this thesis to get better parameters for omnidirectional walking.

### 3.3.5 Crossing, Mutation and Population

A population of parameter sets is used for the evolution. Each parameter set can be seen as an individual, each single parameter corresponds to a gene. All parameters are modelled as genes on the same, single chromosome, because no correlations between the parameters of a parameter set are known that could be used here.

The usage of real robots required for evolution is time-consuming. Therefore the prior choice of evolution parameters such as population size and mutation rate is done manually with a sense of proportion instead of already evolving the evolution parameters itself. A small population of e.g. ten individuals is sufficient for fast advance while using blind mutation and being able to determine fitness differences only imprecisely. The start population consists of different mutations of a given parameter set.

Those five out of the ten parameter sets of a population with the lowest fitness are selected in every generation and replaced by mutations and crossings of the better half. This is a compromise between fast advance and avoidance of outliers and dead ends. The descendants are created by mutation with a probability of 40 percent and by crossing with a probability of 60 percent.

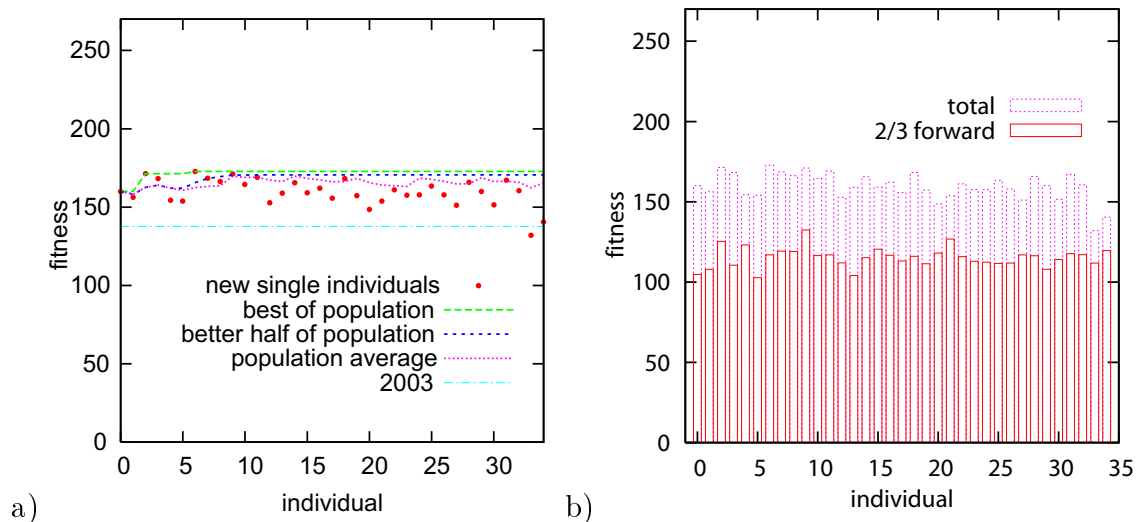
Mutation of a parameter set is done by randomly mutating single genes with a probability of 30 percent, equally distributed up to  $\pm 6$  percent around the former value. Following [32] crossing is done by randomly interpolating between the values of the parents or extrapolating into the direction of the better one respectively.

Using those evolution parameters results in visible and measurable differences between single individuals, but normally no jumps from good to unusable individuals occur in one step. In addition to that a useful part of the search space can be covered in a relatively short time and without prior knowledge about the correlation between single parameters and without many failures.

### 3.3.6 Results of Evolution

First test runs with an ERS-210 were based on the parameter set used by the GermanTeam at the RoboCup world championship in 2003. Already after a few mutations, within the first generation a significantly better parameter set with a fitness of 145.7 instead the previous 137.7 was found (see Figure 3.7).

Because of its higher fitness this first result of the learning method introduced above will be used as starting point for all further evolution attempts, for robots of the model ERS-210 as well as for the ERS-7 later on. The AiboTeamHumboldt even



**Figure 3.9: ERS-210:** **a)** The fitness during the evolution of omnidirectional walk parameter sets increases in the beginning, but then stagnates for a long time. **b)** Splitting the fitness of the parameter sets into their forward component (weighted 2/3) and their backward component reveals, that parameter sets especially good at walking forward are often especially bad for backward walking.

used this parameter set unmodified on its ERS-7 at the German Open<sup>8</sup> in 2004 and won.

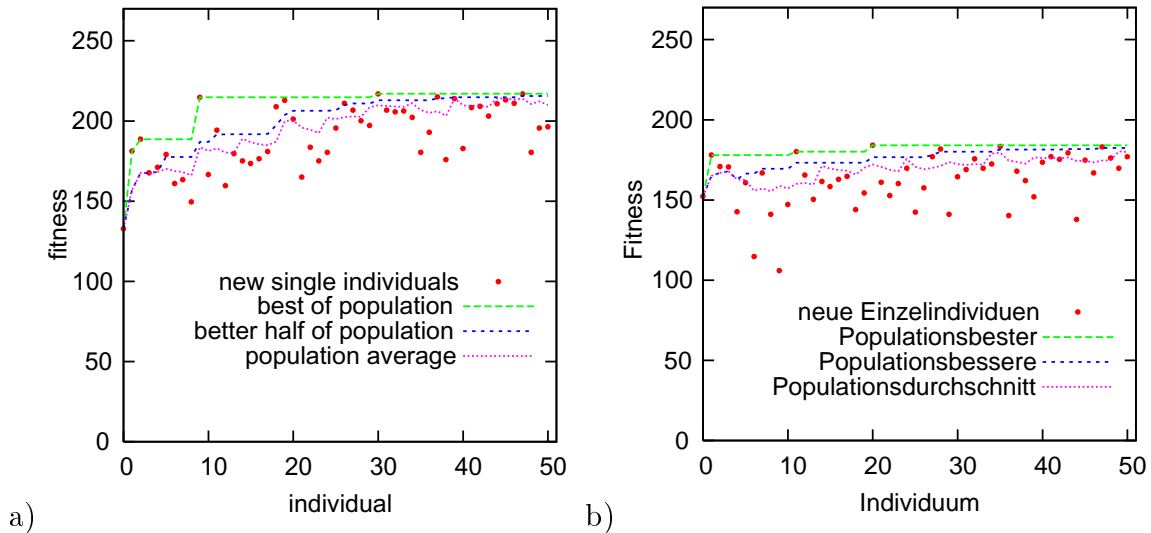
Figure 3.9a shows the results of an evolution of omnidirectional walk parameters. Soon parameter sets with higher fitness were found, but no further improvement happened for a long time. A closer examination revealed (as shown in Figure 3.9 b) that parameter sets with an especially good fitness on the forward part of the path were especially bad on the backward part.

As the walk model used is capable of handling several parameter sets, the idea of optimizing a single parameter set for omnidirectional walking was abandoned at this point. Instead of that one parameter set for forward walking and one for backward walking will be optimized.

The desired aim of finding a single parameter set that covers most motion requests is not dropped with that decision. A parameter set yielding high fitness on the forward part of the path is usually capable of producing good backward walking too, but not with maximum step size respectively speed. So instead of one population of parameter sets two populations will be used from now on, one for walking on the forward part of the path and one for the backward part.

Figure 3.10 shows the development of the fitness of walk parameter sets using two separated populations. This results in parameter sets for forward walking and parameter sets for backward walking each having a significantly higher fitness than the parameter sets in Figure 3.9 that had to be able to cope with forward and backward walking.

<sup>8</sup>open German championship of RoboCup, <http://www.robocup-german-open.de>



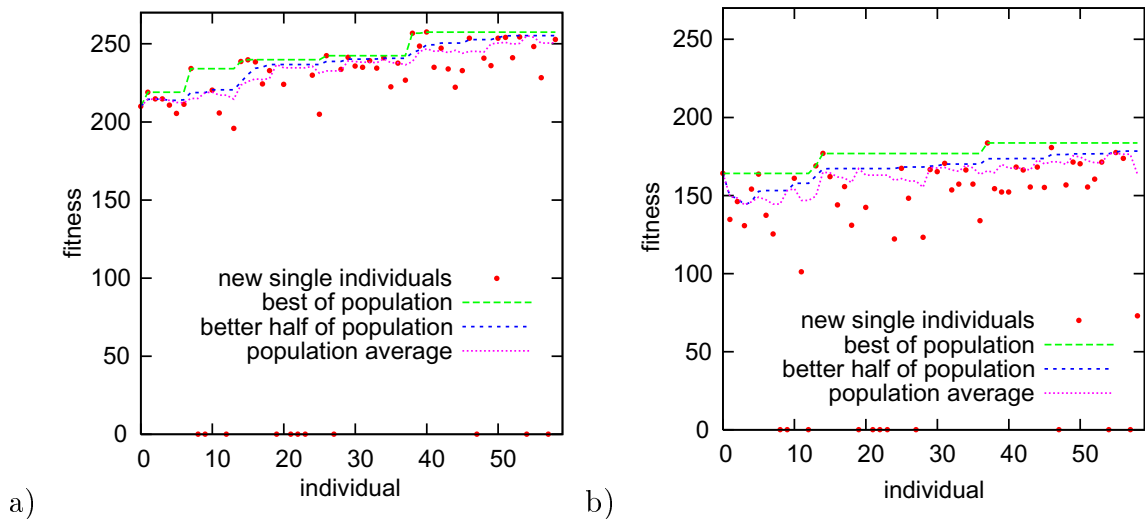
**Figure 3.10:** *ERS-210: Evolution of walk parameter sets with two separated populations for the forward and the backward part of the path, a) Development of the fitness in the population of forward walk parameters, b) Development of the fitness in the population of backward walk parameters*

All experiments up to now were carried out using robots of the model ERS-210. As the successor (ERS-7) is available in the meantime, all further experiments will be applied to the new model ERS-7. Fortunately, good walk parameter sets for an ERS-210 yield good results when used with an ERS-7 too.

But beyond that an ERS-7 has a higher potential: Anatomic modifications like slightly increased leg length and stronger motors allow for considerably higher maximum speeds of about 400 mm/s [32] in comparison to about 300 mm/s reachable with an ERS-210.

The development of the fitness of walk parameter sets for an ERS-7 is shown in Figure 3.11. Starting point was the same parameter set already used in the previous two evolution experiments and originally developed for an ERS-210. Separated populations of parameter sets for forward and backward walking were used for the new robot model ERS-7 as well.

Here the differences between forward and backward walking are even more apparent than on an ERS-210, because the modified anatomic details (e.g. rounded lower front legs) primarily improve forward walking. Therefore the forward walking fitness of a parameter sets for an ERS-7 is significantly higher than the backward walking fitness or the fitness of a walking for an ERS-210.



**Figure 3.11:** ERS-7: Evolution of walk parameter sets with two separated populations for the forward and the backward part of the path, **a)** Development of the fitness in the population of forward walk parameters, **b)** Development of the fitness in the population of backward walk parameters

### 3.4 Measurement of Different Parameter Sets

The walk model introduced in section 2.4 uses 127 parameter sets that can be optimized separately. However that is not necessary in all cases and sometimes not even useful. A good starting point for all 127 parameter sets is duplicating the two parameter sets found in section 3.3: one for walking forward with turning and one for fast backward walking, each copy having its own corrections (if needed).

A couple of compatible parameter sets originating from the past and the work of others ([32]) as well as from manual modifications exist that yield better results in special cases like walking straight on or turning only.

All parameter sets in question have to be measured to be able to decide objectively which parameter set shall be used for which of the 127 motion requests. If it turns out that none of the tested parameter sets can perform a certain requested motion well, than a parameter set can be optimized specifically for that motion request. In some cases it is obvious which parameter modification can solve the problem, then the optimization can be done very quickly by manually correcting the parameter. Otherwise methods from previous studies optimizing walk parameters for constant motion requests can be used [5, 21, 32].

#### 3.4.1 Speed Measurement

Initially the speed of a certain constant walking (i. e. of a certain parameter set with a certain motion request) shall be measured as exactly as possible. Thereby the rotation speed  $\dot{\varphi} = d\varphi/dt$  as well as the translation speeds  $\dot{x} = dx/dt$  forward and  $\dot{y} = dy/dt$  to the left are of interest. The overall distance covered in time  $t$  after

a running start will be determined for that. This is only possible if an appropriate start position according to the motion to be measured is calculated before. Let  $(x_0, y_0, \varphi_0)$  denote such a start position and direction and  $(x_t, y_t, \varphi_t)$  the position of the robot after time  $t$ , then we get:

$$\begin{aligned} \varphi_t &= \varphi_0 + t \cdot \dot{\varphi} \\ x_t &= x_0 + \int_0^t (\dot{x} \cdot \cos(\varphi_t) - \dot{y} \cdot \sin(\varphi_t)) dt \end{aligned} \quad (3.1)$$

$$y_t = y_0 + \int_0^t (\dot{x} \cdot \sin(\varphi_t) + \dot{y} \cdot \cos(\varphi_t)) dt \quad . \quad (3.2)$$

The turning speed used in that equations is:

$$\dot{\varphi} = \frac{\varphi_t - \varphi_0}{t} \quad . \quad (3.3)$$

If the robot does not turn at all during the movement ( $\dot{\varphi} = 0$ ) the equations 3.1 and 3.2 can easily be solved for  $\dot{x}$  and  $\dot{y}$ :

$$\begin{aligned} \dot{x} &= \frac{(x_t - x_0) \cos(\varphi_0) + (y_t - y_0) \sin(\varphi_0)}{t} \\ \dot{y} &= \frac{(y_t - y_0) \cos(\varphi_0) - (x_t - x_0) \sin(\varphi_0)}{t} \quad . \end{aligned}$$

Instead of that we get for  $\dot{\varphi} \neq 0$ :

$$\dot{x} = \frac{\dot{\varphi}}{2 \sin\left(\frac{\dot{\varphi} \cdot t}{2}\right)} \left( (x_t - x_0) \cos\left(r_0 + \frac{\dot{\varphi} \cdot t}{2}\right) + (y_t - y_0) \sin\left(r_0 + \frac{\dot{\varphi} \cdot t}{2}\right) \right) \quad (3.4)$$

$$\dot{y} = \frac{\dot{\varphi} \cdot \sin\left(\frac{\dot{\varphi} \cdot t}{2}\right) \left( (y_0 - y_t) \cos\left(\varphi_0 + \frac{\dot{\varphi} \cdot t}{2}\right) - (x_0 - x_t) \sin\left(\varphi_0 + \frac{\dot{\varphi} \cdot t}{2}\right) \right)}{\cos(\dot{\varphi} \cdot t) - 1} \quad . \quad (3.5)$$

This way the desired speeds  $\dot{x}$ ,  $\dot{y}$  and  $\dot{\varphi}$  can be calculated with equation 3.3, 3.4 and 3.5 from the covered distance between a suitable start position  $(x_0, y_0, \varphi_0)$  and the end position  $(x_t, y_t, \varphi_t)$  after a suitable time  $t$ .

### 3.4.2 Turn Measurement

The calculation above only works exactly enough, if the position and direction of the robot can be determined sufficiently long and continuously during the movement to be measured. Unfortunately this is not the case for measuring fast turning.

The turning speed of really fast turning can best be determined by measuring the time between several zero-crossings of the turn angle, i. e. the time for several complete rotations. However, it is difficult to move the head in a way that keeps the localization guide within the image as long as it in a visible position (for the robot) while turning fast. The robots used are able to track the localization guide reliably

up to a turning speed of about one rotation in 5 seconds ( $\dot{\varphi}=1.2$  rad/s), still being able to determine the walking speeds  $\dot{x}$  and  $\dot{y}$  as well.

At higher turning speeds the head is held straight ahead stiffly to avoid missing the localization guide because of inaccurate head movements as well as making it undetectable because of motion blur. As the localization guide can only be detected for a short time (but more securely) with a stiff head, the walking speeds can not be determined any more. That is tolerable for very fast turning.

#### 3.4.3 Parameter Set Measurement

Measuring the speed with running start requires a adequate start position according to the expected motion. Each start position is chosen in a way the allows a robot to keep track of the localization guide for several seconds during the execution of the requested motion without colliding with the board around the test carpet or with the localization guide. The measurement starts not until two seconds after starting the motion because the robot will not reach its final speed much earlier.

The development of walking speeds while increasing the activation has to be examined in all directions to measure one parameter set completely. Thereby it is sufficient to consider the 8 walking directions and 7 turn walk proportions used in the walk model from section 2.4.

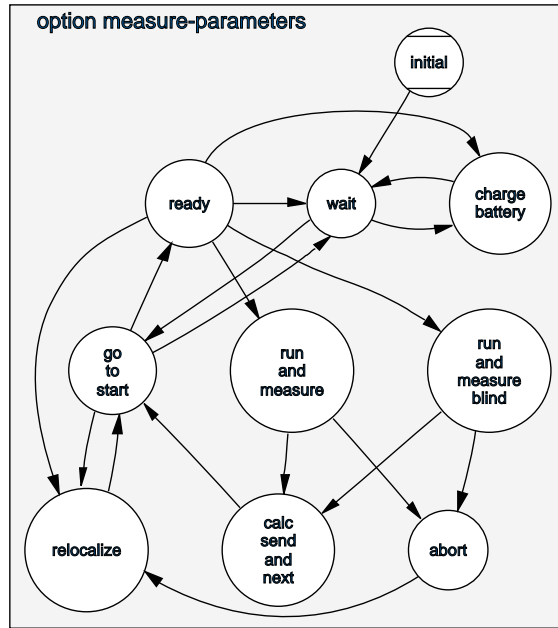
Taking symmetries into account only turning into one direction and walking into eight directions with three turn walk proportions is necessary, so altogether  $1+3\cdot 8 = 25$  directions are to be measured. This will still be somehow expensive, although it is automated quite well. Then again, this step is only necessary once per parameter set, and for specialized parameter sets even just a part of it, e. g. the measurement of turning.

Now the 25 walk and turn directions will be measured separately each with increasing speed by the behavior control developed for that task (Figure 3.12). Therefore the normalized overall speed (without unit, see section 2.4.4) into the current direction will be increased by 0.05 per step.

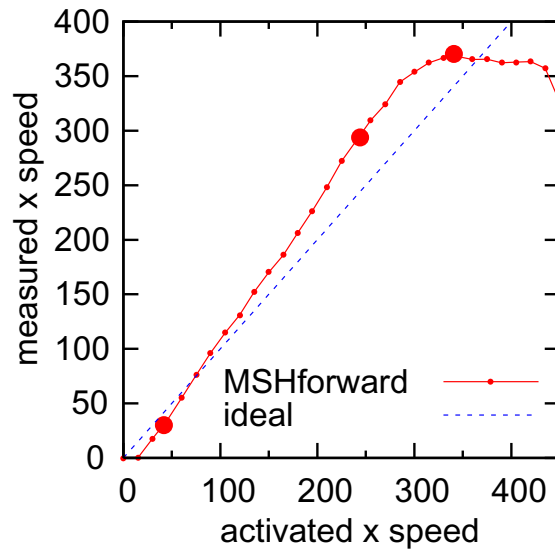
If increasing the activated speed does not increase the measured speed any more or the proportion of activated and measured speed significantly decreased, the behavior control starts to measure the next walking direction. Using this method results in about 15 to 35 measured values per direction (corresponding to a normalized overall speed of 0.75 to 1.75), depending on how fast the robot can actually walk into that direction.

As an example the result of measuring a specific parameter set will be presented and explained here for two walking directions. The parameter set was developed for the German Open 2004 by the GermanTeam member Microsoft Hellhounds.

Lets take a look at forward walking in Figure 3.13 first. The measured speed is lower then the activated one up to a speed of about 80 mm/s, where both speeds match. At higher speeds the the robot is faster than expected and reaches its max-



**Figure 3.12:** Xabsl state machine of the behavior for measuring one parameter set: The real work like choosing the next motion request to be measured or determining real speeds is done within single states (circles).



**Figure 3.13:** Result of measuring the forward walking speed of an ERS-7 using a single fixed parameter set (MSHforward): The values chosen for slow, medium and highest speed on the basis of that measurement are marked.

### 3 Optimization

imum speed at about 370 mm/s. A further increment of the activation does not increase the walking speed any more.

The walk model from section 2.4 uses three parameter sets per walking direction, one for slow, one for medium and one for maximum speed. One of the aims of that construction is the minimization of deviations between requested and measured speed over the complete speed range with as few parameter sets to be calibrated as possible. To reach that aim, three appropriate speeds were chosen manually (and marked in Figure 3.13), between which linear interpolation approximates the gradient well.

Based on the measured values examined so far, the decision for those three speeds step could also have been made automatically using simple criteria. But the measurements additionally revealed deviations in the sideways speed and turning speed that were not yet taken into account.

Now we look at the measurement of the same parameter set for the walking direction  $\pi/4$  (diagonal to the left front) and the turn walk proportion 0.1 (Figure 3.14). Walking direction and turn walk proportion result in robot movements on a circle with fixed diameter at variable speed.

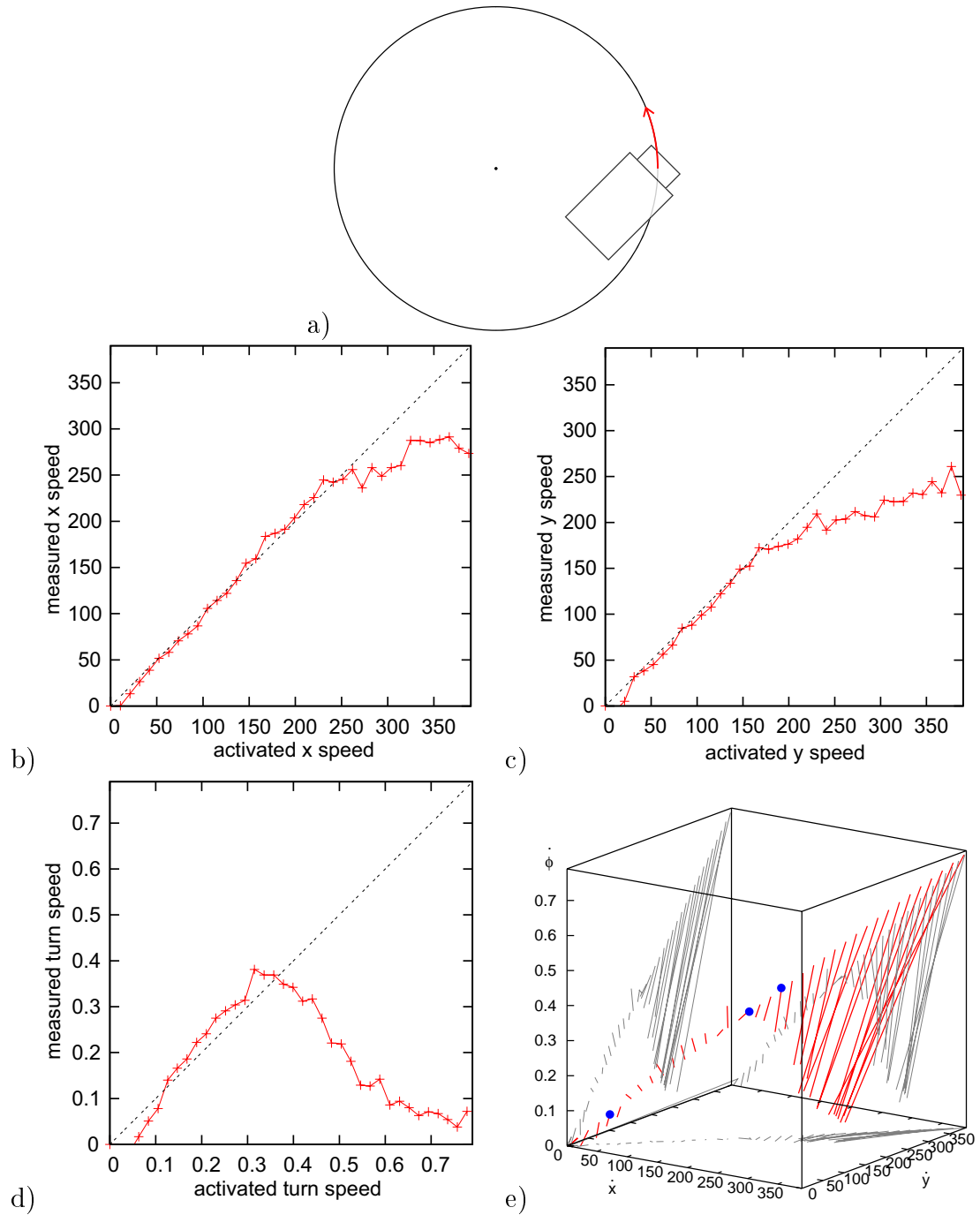
In the beginning, the measured turn and walking speeds follow the activated ones pretty well. Starting at a requested sideways speed of about  $\dot{y} = 180$  mm/s and a belonging turning speed of  $\dot{\varphi} = 0.36$  rad/s the measured sideways speed increases much slower than the activated one, the measured turning speed even decreases.

Appropriate positions for low, medium and maximum speed have to be chosen here too. Thereby the occurring deviations shall not only be detected but also be corrected. Especially for maximum speed that decision is not trivial and thus wont be left to an automatic. Increasing the activation may increase the overall speed, but starting at the position where the measured turning speed decreases the desired turn walk proportion is not (and probably can not) be reached any more. Therefore the maximum speed for the desired turn walk proportions is slightly above that inflection point, because the measured turning speed can be corrected a little bit there using a higher activation (for turning). Out of those considerations a maximum walking speed of 210 mm/s (sideways as well as forward) and a belonging turning speed of 0.42 rad/s was used as maximum (blue markings in Figure 3.14 e).

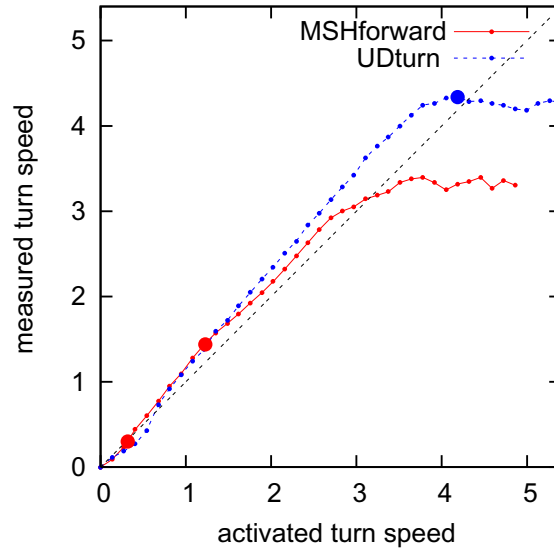
The exact values for the maximum speed into the desired direction as well as the activation necessary to actually reach that will be determined later using the calibration in section 3.5. The first measurement here already gives a much better estimation for what is reachable than before. The medium speed is set to the point where the measured turning speed starts to decrease, because linear interpolation between the speed steps will minimize the average error that way.

Not only one parameter set in question exists, but several. Luckily not all of them have to be measured completely, because often it is sufficient to measure those special cases for which a certain parameter set might be more appropriate than the standard solution.

Specialized parameter sets were found especially for walking backwards and turning. A parameter set optimized for fast turning for example has a lower distance



**Figure 3.14:** Measurement of an ERS-7 parameter set for the walking direction  $\pi/4$  (diagonal) and the turn walk proportion 0.1: **a)** Walking direction and turn walk proportion cause the robot to walk on a circle with fixed diameter at variable speed. Activated and measured values for **b)**  $\dot{x}$ , **c)**  $\dot{y}$  and **d)**  $\dot{\phi}$  are shown as well as **e)** the distance between activated and measured speed (red) with its projections onto the three planes (gray).



**Figure 3.15:** Comparison of the measurement of the turning speed of an ERS-7 between an omnidirectional standard parameter set (MSHforward) and a parameter set specialized for turning (UTurn): The specialized parameter set will be used for fast turning because of its much higher maximum speed. The chosen speeds for slow medium and fastest turning are marked.

between front and rear feet in their rest positions to minimized the robots “turning circle”. With such foot positions it will not be possible to make large forward steps, therefore walking forward with such a parameter set does not have to be measured at all.

The comparison of the measured turning speeds in Figure 3.15 shows, that the parameter set specialized for turning clearly outperforms the standard parameter set while turning fast. Therefore turning at maximum speed will be done with exactly that specialized parameter set. Slow and medium turning on the other side should be performed using the standard parameter set to make direction changes as smoothly as possible.

The medium speed has to be chosen considerably lower than the maximum speed of the standard parameter set, because between medium and maximum speed an interpolation between the standard and the specialized parameter set will be used. Only with such a safety margin it can be expected, that already the interpolation with a small portion of the specialized parameter set increases the actually measured speed further. Therefore the medium turning speed was fixed at 1.44 rad/s in this case.

The walk model allows to use very different parameter sets for relatively similar motion requests, although that is counterproductive, because there will be interpolation between those parameter sets for the motion requests in between. Therefore it is useful to avoid using parameter sets that considerably differ from their neighbors. Otherwise the advantage of a slightly higher maximum speed would probably

parameter	walking forward	walking backward	turning
$x_v$	50.4 mm	43.2 mm	33.6 mm
$y_v$	80.3 mm	79.1 mm	78.1 mm
$z_v$	77.6 mm	88.5 mm	86.5 mm
$x_h$	-52.1 mm	-49.7 mm	-29.6 mm
$y_h$	81.5 mm	77.7 mm	76.9 mm
$z_h$	105.1 mm	101.9 mm	113.7 mm
$h_v$	5 mm	9.8 mm	5 mm
$h_h$	24 mm	28.6 mm	24 mm
$tilt_v$	-0.25	0.072	-0.25
$tilt_h$	0.016	-0.092	0.045
$gp_v$	0.49	0.46	0.5
$gp_h$	0.49	0.51	0.5
$(l_{vl}, l_{vr}, l_{hl}, l_{hr})$	(0, 0.5, 0.5, 0)	(0, 0.5, 0.5, 0)	(0, 0.5, 0.5, 0)
$T$	512 ms	488 ms	352 ms

**Table 3.3:** The final values of the most important parameter sets used: The meaning of those parameters was explained in section 2.4.

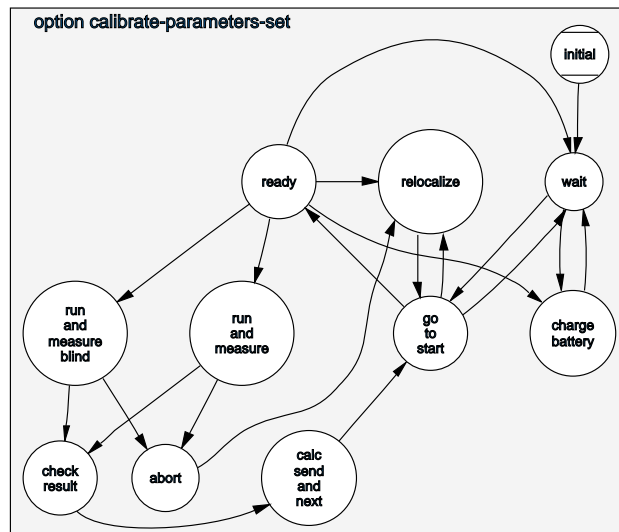
be counterbalanced by worse results of the interpolation between those differing neighbors.

After measuring all parameter sets in question it is known now, witch parameter set is most suitable for which motion request (see table 3.3 too). Furthermore it is known how the normalized overall speed of walking and turning (see section 2.4.4) of a specific parameter set depends on the activation. With that knowledge it was possible to chose suitable values for minimum, medium and maximum speed for all directions. Thus the selection of those 127 parameter sets used by the walk model is finished.

## 3.5 Calibration

The individual parameter sets still have to be calibrated to let the actual movements match the requested ones as much as possible. Such differences are e. g. caused by inertia, friction, inaccuracies within the model or mechanical limitations and result in speed as well as direction deviations, that can for example be seen in Figure 3.14.

The existence of those deviations is already known from the measurement of the parameter sets, but it is not yet corrected. In opposite to the previously used simplifying assumption of few linear correction parameters the calibration of single parameter set allows for minimizing the deviations only occurring at certain motion requests like diagonal walking for the first time.



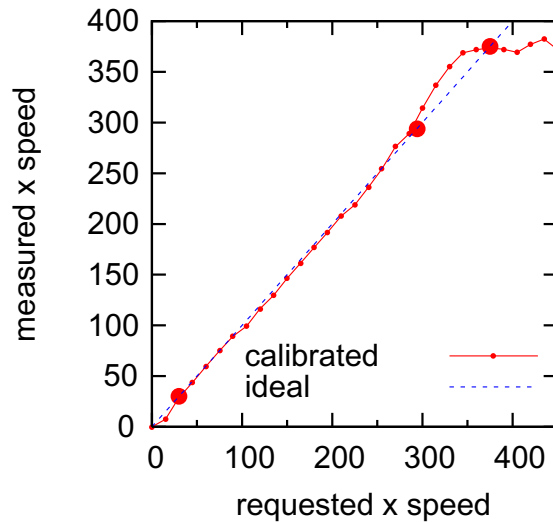
**Figure 3.16:** Xabsl state machine of the behavior for calibrating all parameter sets for the motion request they shall be used for: The real work like choosing the next parameter set or measuring it is done within single states (circles).

The behavior for calibration is specified using Xabsl again (Figure 3.16). Those 66 out of the 127 parameter sets of the walk model from section 2.4 that do not emerge from each other by right left symmetries will be measured using their respective target speed. That takes about 20 minutes altogether and produces a list of requested, correctedly activated and actually measured speeds for all parameter sets.

The activation of those parameter sets not intended for maximum speed can be optimized automatically. That is done by correcting the activation by half of the difference between requested and measured speed. This way the deviation between target and actual speed is reduced (ideally halved) without risking to overshoot the target speed.

The parameter sets intended for maximum speed can not be optimized the same way. On the one hand it is desirable to adjust the maximum speed into a certain walking direction instead of the activation if the measured speed is higher than the requested one. On the other hand it can not be guaranteed, that changing the activation by half of the difference between request and measurement will actually decrease that difference. Instead of that the examined motion may already be exhausted. A further increment of the activated turning speed for example will not be able to increase the actually measured turning speed in that case. Therefore the correction of the activation of parameter sets intended for maximum speed will be done manually and only on the basis of requested, correctedly activated and actually measured speeds.

That way a complete calibration run takes nearly an hour, half of that for automated measurement and correction and the other half for manual checking and calibration of the parameter sets intended for maximum speed.



**Figure 3.17:** Result of the measurement of the forward walking speed of an ERS-7 based on the interpolation between three calibrated parameter sets for slow, medium, and highest speed (marked): The deviations between requested and measured speed are significantly smaller than without calibration (see Figure 3.13).

### 3.5.1 Calibration Result

Aim of the calibration was the minimization of the difference between requested and actually measured speed. The forward walking from Figure 3.13 was measured again, this time with an activation calibrated in multiple steps to see the effect of calibration. Figure 3.17 shows the resulting considerably improved correspondence between requested and measured speed.

The illustration does not show which speeds have to be activated to let the measured speeds of the marked parameter sets match the requested ones. It is only important, that this is possible by an appropriate correction of the activation, not only for the three calibrated (marked) speed steps, but also for the ranges in between using interpolation. The remaining deviation between medium and maximum speed is the price to pay for reaching a higher maximum speed.

Besides this forward walking example the average deviation between requested and actual speed of all parameter sets to be calibrated was determined. Here the absolute values of the deviations instead of its squares were averaged to avoid single mismeasurements and systematically higher deviations e. g. at higher target speeds getting too much weight.

Table 3.4 shows the determined average deviation of all parameter sets to be calibrated in several phases of the optimization process. The first line contains the values from the complete measurement of all parameter sets used from section 3.4.1 without any calibration of the activation. Accordingly, the deviations are the highest there. Then the first manual adaptation of the activation directly derived from the complete measurement follows.

### 3 Optimization

	$ \Delta\dot{x} $ in mm/s	$ \Delta\dot{y} $ in mm/s	$ \Delta\dot{\phi} $ in rad/s
uncalibrated	12,9	12,1	0,086
from measurement	7,7	11,4	0,070
1st calibration	6,4	7,6	0,038
2nd calibration	5,8	8,0	0,030
3rd calibration	4,4	8,7	0,021
2 weeks later	7,9	7,5	0,046
other robot	6,9	7,8	0,039

**Table 3.4:** *The average deviation between requested and actually reached speed of an ERS-7 can be reduced considerably by using calibration.*

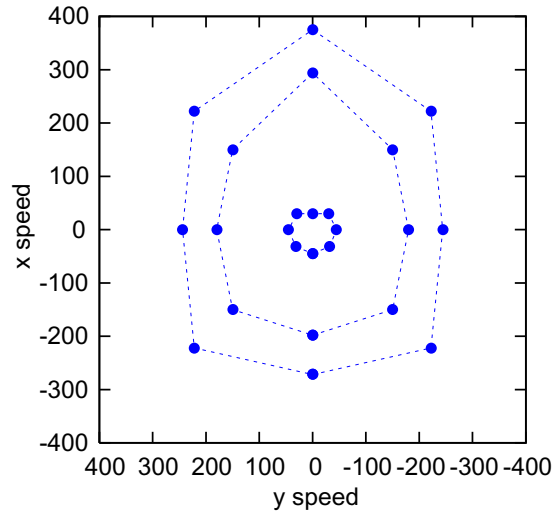
The next three calibration passes result in a further reduction of the deviation between target and measurement (as shown in table 3.4). The deviation of turning speed, that occurred especially using a combination of turning and fast walking, was considerably reduced. Unfortunately this reduction is combined with a slight increment of the sideways speed deviation. Further calibration steps mainly modifying the sideways speed would probably counterbalance that effect.

As the average deviation contains measurement errors as well, it can not be decreased arbitrarily. The measurement was rerun with the result of the third calibration step several times with different robots after a few weeks to find out which accuracy is useful in practice. Thereby it was instantly noticeable that the same robot originally used for calibration lost a good part of its improved accuracy after two weeks of intensive usage (preparations for the world championship). Another robot identical in construction but not used for calibration before also showed higher deviations than the first robot directly after the third calibration step.

Therefore more than three calibration steps would only be useful if the calibration depended on factors such as attrition state of the robots as well and was repeated regularly and for single robots. The amount of time and additional attrition necessary for that can not be legitimated. Thus a calibration in three steps in series is a reasonable compromise, because the deviations occurring when using different robots for a longer time with walking calibrated that way are still significantly lower than without any calibration.

At the end of the calibration we have maximum speeds and calibrated activations for all walking directions and turn walk proportions the walk model can use a separate parameter set for. Figure 3.18 shows the speeds of all those parameter sets used that do not contain any turning.

The speed range covered this way is obviously larger than that covered by a single parameter set and the speed limitation to e.g. an ellipse necessary in that case. Moreover those higher speeds are even calibrated, so the actually measured speeds match the ones intended by the activation much better.



**Figure 3.18:** The chosen minimum, medium and maximum speed of an ERS-7 for all eight walking directions without turning: Using several parameter sets each calibrated for such a speed allows to cover a significantly larger speed range than using a single parameter set with an elliptic speed limitation.

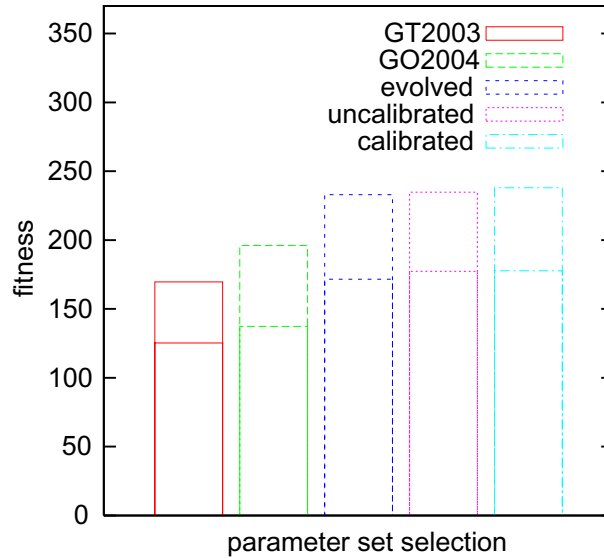
## 3.6 Result of Optimization

After several optimizations the path from section 3.3.1 will be used once again to verify in how far the robots walking has improved.

The optimization of walking was carried out in three steps. The first one was the evolutionary improvement of an omnidirectional parameter set. After that the resulting as well as other available parameter sets were exactly measured to be able to choose the best parameter sets for certain motion requests. Finally all parameter sets used were calibrated for their respective purpose (speed and direction). Using optimized, specialized and calibrated parameter sets is expected to increase the accuracy of walking, the direction dependent maximum speeds, and therefore the fitness of walking on the path.

In Figure 3.19 the progress of the fitness of walk parameter sets for an ERS-7 during the optimization process is shown. The parameter set (“GT2003”), that was manually optimized for an ERS-210 and used in the RoboCup world championship 2003, is ill-suited for an ERS-7. But already the first parameter set (“GO2004”) optimized using the evolutionary approach from this thesis and still intended for an ERS-210 significantly increased the fitness for an ERS-7 as well. Therefore it was used by the AiboTeamHumboldt at the German Open 2004.

The subsequent ERS-7 specific evolution separated for forward and backward walking (“evolved”) once more clearly increased the fitness. Only minor improvements to the fitness were achieved by selecting those 127 parameter sets in section 3.4 (“uncalibrated”) and by the calibration (“calibrated”) in section 3.5. Choosing several distinct parameter sets improved walking basically in special cases like very



**Figure 3.19:** ERS-7: The evolutionary method used produced parameter sets with a significantly increased fitness whereas the calibration did not have a noteworthy influence onto the fitness. The fitness values of selected parameter sets or parameter set combinations are shown, subdivided into their forward (weighted 2/3) and backward components.

fast turning that were not covered by walking on the path. Calibration mainly improved the accuracy of walking, particularly at slower speeds, and therefore had little influence on the fitness function either.

All three optimization steps (evolution, choosing specialized parameter sets, and calibration) made a relevant contribution to the improvement of walking. This has been confirmed by the victory of the GermanTeam (using the walking optimizations described in this thesis) in the RoboCup world championship 2004.

## 4 Summary and Outlook

Walking motions for four-legged robots of the type Sony ERS-210 and ERS-7 were modeled, optimized and calibrated in this thesis. The resulting walk model allows for omnidirectional locomotion. For the first time it offers the possibility to use differently optimized parameter sets for different walking directions or speeds without the need to switch between them explicitly. Therefore improvements for a certain walking direction do not imply the worsening of walking into other directions any more.

Moreover the occurring speed differences between theory and reality can be corrected locally for the first time. The correction of unwanted effects only occurring at maximum speed, for example, does not have any influence on the exactness of walking at medium speed. Therefore the attainable maximum speeds and the deviation between requested and real motion are independent of each other and the maximum speeds are allowed to be different for different walking directions.

The aim of the optimization method used was in particular the improvement of omnidirectional walking with changing walking directions instead of the more common optimization of constant motions. This way walk parameters were actually found that result in a considerably better walking.

However, it also turned out that it is not profitable to optimize omnidirectional walking (with only one constant parameter set) for speed, because walking forward and backward requires different parameters to reach the respective maximum speed. Instead it is more important to have a walk model like the one in section 2.4 that is capable of using different parameter sets for certain special tasks and simplifies the transitions between those parameter sets as much as possible.

This requires parameter sets which can not only cope with their special task but are suitable for similar motion requests as well. Including a walking direction correction into the optimization process proved to be appropriate to find such parameter sets. The evolution method used was able to produce significantly better parameter sets in a short time.

Future optimization methods should aim at integrating several optimization steps. In particular it seems to be useful to optimize multiple parameter sets including the transitions in between in parallel, because using several different but coordinated parameter sets is the most successful approach at the moment.

After producing good parameter sets the walking speed resulting from all parameter sets in question was measured. Thereby it was remarkable how helpful an accurate method of speed measurement can be. The determination of direction dependent maximum speeds, the objective comparison between two parameter sets

or the detection of the non-linear dependency between requested and actual speed would not have been possible with manual measurements only.

The calibration of walking was based upon that exact measurements as well. For the first time this calibration was done separately for different directions and speeds resulting in a noticeable decrement of the deviations between requested and actually reached speed. That is especially useful in situations with imprecise localization, e. g. while playing soccer in RoboCup.

At the end of the optimization process a walk model with appropriate parameters was available that allows omnidirectional walking and turning up to 30 percent faster than before. The occurring deviations between requested and actually performed motion were reduced to one-third without having to limit the maximum walking speed artificially. Using the presented walking motions contributed to the victory of the AiboTeamHumboldt in the German Open 2004 as well as to the victory of the GermanTeam in the RoboCup world championship 2004 in Lisbon.

Value was set on the possibility to automate as much of the optimization and calibration as possible. The calibration introduced as well as using several parameter sets dramatically increased the complexity, but the additional expenses for a human trainer are kept within a limit due to the automation used.

Furthermore automated methods improve the comparability and reproducibility of the result, because several conditions such as the length of the measured walk distance do not depend on the operator any more. Using divers benchmarks like the introduced fitness function or an exactly measured speed was inevitable at this point, because automated methods need such objective criteria to decide on the quality of a walk parameter set.

An accurate localization was necessary to determine the exact speed of walking. Therefore a lot of work was invested in the creation of such a localization. The availability of another sufficiently exact position and direction determination method would have saved much effort.

But after all the solution found, a simply producible, passive, portable localization guide being analyzable by the robot itself, is an appropriate means. Instead of costly external hardware (ceiling camera, laser scanner) a simple assistance is sufficient here.

If a robot in general misses a dedicated sensor for a certain task and the required information can not or not reliably be obtained from an external source, it is often useful to provide simple markings of important objects or comparable quantities. Humans are able to find such control point in the environment on their own, but presently a robot is still better provided with this information.

The 127 parameter sets used at the end of this thesis are not optimal. Numerous individual cases could have been improved further using appropriate optimization methods. The sideways walking speed of an ERS-7 for example significantly underachieved. However, the optimization of parameters for single constant motion request was not the aim of this thesis, because there are already several publications about methods appropriate for that [5, 32].

Instead more emphasis was put on omnidirectional usability of the walk model, e. g. by having continuous transitions between all parameter sets. The result is a well working cooperation between several relatively similar and individually calibrated parameter sets yielding low deviations between request and reality as well as high speeds.

From my point of view optimizing the foot trajectory has the highest potential at the moment. There were several successful attempts to improve the foot trajectories, e. g. by using unrestricted four points in space instead of a parallelogram in a vertical plane (*Free Form Quad*, see [5], still using an ERS-210 only). Nevertheless many of the occurring effects are not sufficiently understood or modelled yet, e. g. intentionally taking advantage of rolling on the lower legs.

Another example is the result of [21], where very high walking speeds can be reached with a modelled duty factor of 0.43. That means having an average of less than two of the four legs on the ground in theory, but that can not be observed in practice. Instead of that the feet stay on the ground for a part of the time they are modelled to be in the air resulting in an up and down movement of the robot like the Canter Action from [14].

Moreover the analysis of the results of [32] reveals, that the highest walking speed reached by an ERS-7 so far is founded on the “misuse” of parameters of the foot trajectory by an evolutionary algorithm. The aim was to use (as usual) rectangles or half ellipses with constant speeds within the few sections of the foot trajectories without breaks in between. Instead of that the evolutionary choice of parameters resulted in stopping single feet at the corners of their trajectories while other legs continued their steps. Apparently that increased the grip of the feet to the ground resulting in faster walking.

The controllability of rough or variable underground as well as the usage of different, especially two-legged robots was not examined at all in this thesis. But both are requisites for the initially mentioned aim of using robots in human surroundings. Undoubtedly a lot of new problems will be connected with both, but numerous fundamental ideas presented in this thesis can still be applied. A parametric model for omnidirectional walking is required and single motions need to be optimized and calibrated for specific purposes. That should be based on objective data and automated as far as possible. Thus a well proven approach is available that can be adapted to future modelling and optimization of walking motions.

Hope remains that the development of legged robots and their walking motions stays exciting. In this spirit the researches all over the world can be wished, that some day the vision of RoboCup - defeating the current “real” (human) soccer world champion in a fair play with autonomous robots - will come true.



# A Bibliography

- [1] AiboTeamHumboldt. website. <http://www.aiboteamhumboldt.com>, 2004.
- [2] Minoru Asada, Hiroaki Kitano, Itsuki Noda, Manuela Veloso. RoboCup: Today and tomorrow - what we have learned. *Artificial Intelligence*, 110(2):193–214, 1999.
- [3] Aude Billard, Auke Jan Ijspeert. Biologically inspired neural controllers for motor control in a quadruped robot. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000*, volume 6. IEEE, 2000.
- [4] Ronnie Brunn, Uwe Düffert, Matthias Jüngel, Tim Laue, Martin Löttsch, Sebastian Petters, Max Risler, Thomas Röfer, Andreas Sztybryc. GermanTeam 2001. In Andreas Birk, Silvia Coradeschi, Satoshi Tadokoro (editors), *RoboCup 2001 Robot Soccer World Cup V*, number 2377 in Lecture Notes in Computer Science, pages 705–708, Heidelberg, Germany, 2002. Springer. More detailed in <http://www.tzi.de/kogrob/papers/GermanTeam2001report.pdf>.
- [5] Jin Chen, Eric Chung, Ross Edwards, Nathan Wong, Eileen Mak, Raymond Sheh, Min Sub Kim, Ales Tang, Nicodemus Sutanto, Bernhard Hengst, Claude Sammut, Will Uther. rUNSWift 2003. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Heidelberg, Germany, 2004. Springer. To appear.
- [6] Ingo Dahm, Jens Ziegler. Using artificial neural networks to construct a meta-model for the evolution of gait patterns. In P. Bidaud, F. Ben Amar (editors), *Proceedings of the 5th International Conference on Climbing and Walking Robots (CLAWAR 2002)*, pages 825–832, Bury St. Edmunds, GB, 2002. Professional Engineering Publishing.
- [7] Uwe Düffert, Matthias Jüngel, Tim Laue, Martin Löttsch, Max Risler, Thomas Röfer. GermanTeam 2002. In Gal A. Kaminka, Pedro U. Lima, Raúl Rojas (editors), *RoboCup 2002 Robot Soccer World Cup VI*, number 2752 in Lecture Notes in Computer Science, Heidelberg, Germany, 2003. Springer. More detailed in <http://www.tzi.de/kogrob/papers/GermanTeam2002.pdf>.
- [8] J.E.J. Duysens, Henry W.A.A. van de Crommert, Bouwien C.M. Smits-Engelsman, Frans C.T. van der Helm. A walking robot called human: lessons to be learned from neural control of locomotion. *Journal of Biomechanics*, 35(4):447–454, 2000.

- [9] Dieter Fox, Wolfram Burgard, Frank Dellaert, Sebastian Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI'99)*, pages 343–349, Orlando, FL, 1999.
- [10] GermanTeam. website. <http://www.robocup.de/germanteam>, 2004.
- [11] Michael Hardt, Maximilian Stelzer, Oskar von Stryk. Efficient dynamic modeling, numerical optimal control and experimental results for various gaits of a quadruped robot. In *CLAWAR 2003 - 6th International Conference on Climbing and Walking Robots*, pages 601–608, Catania, Italy, 2003.
- [12] Michael Hardt, Oskar von Stryk. The role of motion dynamics in the design, control and stability of bipedal and quadrupedal robots. In Gal A. Kaminka, Pedro U. Lima, Raúl Rojas (editors), *RoboCup 2002: Robot Soccer World Cup VI*, volume 2752 of *Lecture Notes in Computer Science*, pages 206–223, Heidelberg, Germany, 2003. Springer.
- [13] Bernhard Hengst, Darren Ibbotson, Son Bao Pham, John Dalglish, Mike Lawther, Phil Preston, Claude Sammut. The UNSW RoboCup 2000 Sony Legged League team. In Tucker Balch, Gerhard Kraetzschmar (editors), *RoboCup 2000: Robot Soccer World Cup IV*, number 2019 in *Lecture Notes in Artificial Intelligence*, pages 70–72 (64–75), Heidelberg, Germany, 2001. Springer.
- [14] Bernhard Hengst, Darren Ibbotson, Son Bao Pham, Claude Sammut. Omnidirectional locomotion for quadruped robots. In Andreas Birk, Silvia Coradeschi, Satoshi Tadokoro (editors), *RoboCup 2001 Robot Soccer World Cup V*, number 2377 in *Lecture Notes in Computer Science*, pages 368–373, Heidelberg, Germany, 2002. Springer.
- [15] Gregory S. Hornby, Seiichi Takamura, Jun Yokono, Osamu Hanagata, Takashi Yamamoto, Masahiro Fujita. Evolving robust gaits with Aibo. *IEEE International Conference on Robotics and Automation*, pages 3040–3045, 2000.
- [16] Vincent Hugel, Pierre Blazevic. Towards efficient implementation of quadruped gaits with duty factor of 0.75. In *Proceedings of the IEEE International Conference On Robotics and Automation*, pages 2360–2365, 1999.
- [17] Nick Jakobi. The minimal simulation approach to evolutionary robotics. In Takashi Gomi (editor), *Evolutionary Robotics - From Intelligent Robots to Artificial Life (ER'98)*, pages 133–190. AAAI Books, 1998.
- [18] Hiroshi Kimura, Yasuhiro Fukuoka, Yoshiro Hada, Kunikatsu Takase. Adaptive dynamic walking of a quadruped robot on irregular terrain by using a neural system model. *Advanced Robotics*, 15(8):859–876, 2001.
- [19] Hiroshi Kimura, Yasuhiro Fukuoka, Yoshiro Hada, Kunikatsu Takase. Three-dimensional adaptive dynamic walking of a quadruped - rolling motion feedback

- to CPGs controlling pitching motion. In *Proceedings of IEEE Robotics and Automation (ICRA2002)*, pages 2228–2233, 2002.
- [20] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, Eiichi Osawa. RoboCup: The robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 340–347. ACM Press, 1997.
- [21] Nate Kohl, Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2004. To appear.
- [22] John R. Koza. *Genetic Programming*. MIT Press, Cambridge, MA, 1992.
- [23] Scott Lenser, James Bruce, Manuela Veloso. CMPack: A complete software system for autonomous legged soccer robots. In M. Brian Blake (editor), *Proceedings of the 5th International Conference on Autonomous Agents*, Montreal, Canada, 2001.
- [24] M. Anthony Lewis. Gait adaptation in a quadruped robot. *Autonomous Robots*, 12(3):301–312, 2002.
- [25] Hod Lipson, Jordan B. Pollack. GOLEM@home website. Design and Manufacture of Robotic Lifeforms, <http://demo.cs.brandeis.edu/golem/>, 2000.
- [26] Martin Löttsch, Joscha Bach, Hans-Dieter Burkhard, Matthias Jüngel. Designing agent behavior with the extensible agent behavior specification language XABSL. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Heidelberg, Germany, 2004. Springer. To appear.
- [27] Robert B. McGhee, Andrew A. Frank. On the stability properties of quadruped creeping gaits. *Mathematical Biosciences*, 3(3/4):331–351, 1968.
- [28] Stefano Nolfi, Dario Floreano. *Evolutionary Robotics*. MIT Press, 2000.
- [29] Stefano Nolfi, Dario Floreano, Orazio Miglino, Francesco Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics. In Rodney A. Brooks, Patti Maes (editors), *Artificial Life IV*, pages 190–197, 1994.
- [30] Michael J. Quinlan, Stephan K. Chalup, Richard H. Middleton. Techniques for improving vision and locomotion on the sony aibo robot. In Jonathan Roberts, Gordon Wyeth (editors), *Proceedings of the 2003 Australasian Conference on Robotics and Automation*, Brisbane, Australia, 2003.
- [31] Thomas Röfer. An architecture for a national RoboCup team. In Gal A. Kaminka, Pedro U. Lima, Raúl Rojas (editors), *RoboCup 2002 Robot Soccer World Cup VI*, number 2752 in Lecture Notes in Artificial Intelligence, pages 417–425, Heidelberg, Germany, 2003. Springer.

- [32] Thomas Röfer. Evolutionary gait-optimization using a fitness function based on proprioception. In *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Heidelberg, Germany, 2005. Springer. To appear.
- [33] Thomas Röfer, Ingo Dahm, Uwe Düffert, Jan Hoffmann, Matthias Jünger, Martin Kallnik, Martin Löttsch, Max Risler, Maximilian Stelzer, Jens Ziegler. GermanTeam 2003. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Heidelberg, Germany, 2004. Springer. More detailed in <http://www.tzi.de/kogrob/papers/GermanTeam2003.pdf>. To appear.
- [34] The RoboCup Federation. RoboCup website. <http://www.robocup.org>, 2004.
- [35] Greg Welch, Gary Bishop. An introduction to the Kalman Filter. Technical report, University of North Carolina at Chapel Hill, 1995. 2003 renewed.
- [36] Matthias Werner, Helmut Myritz, Uwe Düffert, Martin Löttsch, Hans-Dieter Burkhard. HumboldtHeroes. In Tucker Balch, Gerhard Kraetzschmar (editors), *RoboCup 2000: Robot Soccer World Cup IV*, number 2019 in Lecture Notes in Artificial Intelligence, pages 651–654, Heidelberg, Germany, 2001. Springer.
- [37] Krister Wolff, Peter Nordin. Evolution of efficient gait with humanoids using visual feedback. In *Proceedings of the 2nd IEEE-RAS International Conference on Humanoid Robots, Humanoids 2001*, pages 99–106. IEEE, 2001.

## B Note of Thanks

I would like to thank all people who directly or indirectly helped me to create this diploma thesis. Special thanks go to:

- Prof. Hans-Dieter Burkhard for bringing RoboCup to the Humboldt University and for supervising this diploma thesis
- the RoboCup community for many years of inspiration and motivation
- the GermanTeam, especially Matthias Jüngel, Martin Löttsch, Max Risler, and Thomas Röfer for continuous contributions to the common software
- Jan Hoffmann for looking after this thesis
- Andreas Kunert, Mechthild Düffert, Elisabeth Ludewig, and Janine Kuhn for (German) proof-reading



# Declaration

I, Uwe Düffert, herewith declare, that I wrote this diploma thesis on my own and did not use any unnamed sources or aid.

I agree, that the Humboldt University Berlin makes this thesis publicly available, e. g. in its libraries.

# Afterword

This is “only” a translation of the German original. It is neither proof-read nor corrected nor accepted by anyone else but me. It only exists because I truly believe that a diploma thesis in this world and this area of research should be available in English.

Berlin, 22.07.2004 (German Original)

Berlin, February 19, 2006 (translation finished)