A sensory-based adaptive walking control algorithm for variable speed biped robot gaits

Andrew L. Kun
University of New Hampshire, Durham
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
A SENSORY-BASED ADAPTIVE WALKING CONTROL ALGORITHM FOR
VARIABLE SPEED BIPED ROBOT GAITS

by

ANDREW L. KUN
BSEE, University of New Hampshire, 1992
MSEE, University of New Hampshire, 1994

DISSERTATION

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Doctor of Philosophy
in
Engineering

May, 1997
This dissertation has been examined and approved.

Dissertation Director. W. Thomas Miller, III,
Professor of Electrical and Computer Engineering

Filson H. Glanz,
Professor of Electrical and Computer Engineering

L. Gordon Kraft, III
Professor of Electrical and Computer Engineering

David E. Limbert,
Professor of Mechanical Engineering

Lee L. Zia
Associate Professor of Mathematics

5/6/97
Date
This dissertation is dedicated to my parents Marija and Ladislav Kun who gave me the gift of intellectual curiosity.
ACKNOWLEDGMENTS

First and foremost I would like to thank Professor Tom Miller for his generous support. I learned a great deal from him and I feel very fortunate that I had him as my advisor.

I would like to thank Professors Filson Glanz, Gordon Kraft, and David Limbert for serving on my committee, and for their help throughout my graduate career. I would also like to thank Professor Lee Zia for serving on my committee.

I would like to thank Steve Kun for his help throughout my college years, from my freshman year to my Ph.D. dissertation.

Thanks to Norbert Valverde for discussing bipedal walking with me - every time he stopped by the Robotics Lab something started working.

I would also like to thank Jon Scalera for his help on various parts of this project.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDICATION</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xviii</td>
</tr>
<tr>
<td>LIST OF ACRONYMS</td>
<td>xix</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>xxi</td>
</tr>
</tbody>
</table>

## CHAPTER

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>1.1 PROBLEM IDENTIFICATION</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>1.2 RESEARCH GOAL</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>1.3 DISSERTATION ORGANIZATION</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>2. HYPOTHESES AND RESEARCH APPROACH</strong></td>
<td>7</td>
</tr>
<tr>
<td><strong>2.1 THE MAIN HYPOTHESIS</strong></td>
<td>7</td>
</tr>
<tr>
<td><strong>2.2 THE SECOND HYPOTHESIS</strong></td>
<td>7</td>
</tr>
<tr>
<td><strong>2.3 THE THIRD HYPOTHESIS</strong></td>
<td>9</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4 The Fourth Hypothesis</td>
<td>11</td>
</tr>
<tr>
<td>2.5 Research Approach</td>
<td>14</td>
</tr>
<tr>
<td>3. Background</td>
<td>18</td>
</tr>
<tr>
<td>3.1 Research Effort Classification Methods</td>
<td>18</td>
</tr>
<tr>
<td>3.1.1 Classification According to Control Goals</td>
<td>18</td>
</tr>
<tr>
<td>3.1.2 Classification According to Control Methods</td>
<td>20</td>
</tr>
<tr>
<td>3.1.3 Classification According to Bipedal Walking Gaits</td>
<td>23</td>
</tr>
<tr>
<td>3.1.4 Classification According to Result Verification Methods</td>
<td>24</td>
</tr>
<tr>
<td>3.2 Research on Bipedal Walking</td>
<td>24</td>
</tr>
<tr>
<td>3.2.1 Projects Illustrating Classification According to Control Goals</td>
<td>24</td>
</tr>
<tr>
<td>3.2.2 Projects Illustrating Classification According to Control Methods</td>
<td>30</td>
</tr>
<tr>
<td>3.2.3 Projects Illustrating Classification According to Bipedal Walking Gaits</td>
<td>34</td>
</tr>
<tr>
<td>3.2.4 Projects Illustrating Classification According to Result Verification Methods</td>
<td>35</td>
</tr>
<tr>
<td>3.3 Building an Experimental Biped</td>
<td>36</td>
</tr>
<tr>
<td>3.4 Biped Research at the UNH Robotics Lab</td>
<td>39</td>
</tr>
<tr>
<td>4. System Hardware</td>
<td>41</td>
</tr>
<tr>
<td>4.1 Low Level Hardware</td>
<td>42</td>
</tr>
<tr>
<td>4.1.1 Biped Supporting Structure and Motors</td>
<td>42</td>
</tr>
<tr>
<td>4.1.2 Biped Sensors</td>
<td>44</td>
</tr>
<tr>
<td>4.1.3 Virtual Body Angle Sensors</td>
<td>45</td>
</tr>
<tr>
<td>4.1.4 Low Level Control Hardware</td>
<td>51</td>
</tr>
<tr>
<td>4.1.4.1 Heat Dissipation Issues</td>
<td>60</td>
</tr>
</tbody>
</table>
4.1.5 Battery Power 60
4.1.6 Tethered Operation 61
4.2 High Level Hardware 62
4.2.1 Serial Communication Link 63

5. Adaptive Dynamic Balance 64

5.1 Posture Space 64
5.2 Hypotheses 66
5.3 Control Architecture 68
5.4 Neural Network Training and Qualitative Results 73
5.5 Discussion 74

6. Adaptive Static Balance 75

6.1 Static Walking Gait 75
6.2 Adaptive Control of Static Balance Walking 77
6.3 High Level Control 78
6.3.1 High Level Software Implementation 79
6.3.2 High Level Control Architecture 81
6.3.3 The Gait Generator 85
6.3.4 Balance Control Using CMAC Neural Networks 91
6.3.5 Posture Command Correction Using CMAC Neural Networks and PID Control 93
6.3.6 Overview of CMAC Neural Networks Used in the High Level Controller 95
6.3.7 Treating the Three-Dimensional Walking Problem as Two Two-Dimensional Problems 98

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
7.2.1 SLOW MOTION ALLOWED DYNAMICS TO BE NEGLECTED 129
7.2.2 REACTIVE CONTROL USED ALONG WITH ADAPTIVE CONTROL. 129
7.2.3 RIGHT-LEFT LEAN ANGLE COMMANDED TO BE CONSTANT IN SINGLE SUPPORT PHASE 129
7.2.4 RIGHT-LEFT LEAN POSTURE COMMAND CORRECTIONS PERFORMED BY BOTH A
CMAC AND A PID CONTROLLER 129
7.2.5 EXTENSIVE USE OF MEASURED CENTERS OF FORCE (CFs) 130
7.2.6 THE SUPPORTING FOOT HAS TO BE UNDER THE UPPER BODY 130
7.2.7 INTEGRAL CONTROL USED TO PROVIDE GOOD FOOT CONTACT IN DOUBLE SUPPORT
PHASE 131
7.2.8 DESIRED FRONT-BACK LEAN ANGLE CONSTANT 132
7.2.9 DESIRED FRONT-BACK LEAN ANGLE LESS THAN FOR DYNAMIC WALKING 132
7.2.10 THE CONTROLLER COULD ONLY IMPLEMENT SLOW WALKING RATES 133
7.2.11 SIMPLIFIED KINEMATICS 134
7.3 UNIFIED CONTROLLER DESIGN HYPOTHESES 134
7.3.1 MAJOR HYPOTHESES 134
7.3.2 SECONDARY HYPOTHESES 135
7.4 WALKING GAIT 138
7.5 ADAPTIVE CONTROL OF WALKING 141
7.6 HIGH LEVEL CONTROL 143
7.6.1 HIGH LEVEL SOFTWARE IMPLEMENTATION 144
7.6.2 HIGH LEVEL CONTROL ARCHITECTURE 144
7.6.3 THE GAIT GENERATOR AND LEARNING TO LIFT THE FEET (CMAC 1) 146
7.6.4 FRONT-BACK BALANCE CONTROL (CMAC 2) 156
7.6.5 POSTURE COMMAND CORRECTION USING CMAC NEURAL NETWORKS (CMAC 3 AND
CMAC 4) 157
7.6.6 SIMPLIFIED KINEMATICS 160
7.6.7 FOOT CONTACT CONTROL THROUGH ANKLE POSITION CORRECTION (CMAC 5) 161
7.6.8 OVERVIEW OF CMACs USED IN THE HIGH LEVEL CONTROLLER 164
7.7 LOW LEVEL CONTROL 167

8. RESULTS 170

8.1 VARIABLE GAIT SPEED EXPERIMENT 171
8.1.1 CHANGE IN PERFORMANCE WITH TRAINING 174
8.1.2 WALKING PERFORMANCE WITH CONSTANT GAIT SPEED 177
8.1.3 PERFORMANCE WITH GAIT SPEEDS CHANGED ON THE FLY 180
8.1.4 PERFORMANCE AT SPEEDS THAT WERE TOO HIGH FOR THE CONTROLLER 187
8.1.5 MOTOR DRIVE SIGNAL OBSERVATIONS DURING WALKING 189

8.2 WALKING VELOCITY AND STRIDE LENGTH MEASUREMENT EXPERIMENT 193
8.3 RELIABILITY EXPERIMENT 194

9. DISCUSSION 196

9.1 DISCUSSION OF VARIABLE GAIT SPEED EXPERIMENT RESULTS 196
9.1.1 DISCUSSION OF CHANGE IN PERFORMANCE WITH TRAINING 197
9.1.2 DISCUSSION OF WALKING PERFORMANCE WITH CONSTANT GAIT SPEED 199
9.1.3 DISCUSSION OF PERFORMANCE WITH GAIT SPEEDS CHANGED ON THE FLY 202
9.1.4 DISCUSSION OF PERFORMANCE AT SPEEDS THAT WERE TOO HIGH FOR THE CONTROLLER 206
9.1.5 DISCUSSION OF MOTOR DRIVE SIGNAL OBSERVATIONS DURING WALKING 209

9.2 DISCUSSION OF THE RESULTS OF THE WALKING VELOCITY AND STRIDE LENGTH MEASUREMENT EXPERIMENT 211

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
## LIST OF FIGURES

**FIGURE 1.1** The NPCM has to be within the stability region in the case of static balance.  

**FIGURE 2.1** Stability region: Position of the NPCM in the case of static balance, and the ZMP in the case of full foot contact dynamic balance  

**FIGURE 2.2** Biped gait (front view)  

**FIGURE 2.3** Idealized biped model  

**FIGURE 2.4** Relationship between the time period the foot spends in the air, the initial lean angle, and the sideways leaning speed  

**FIGURE 2.5** Research approach and hypotheses  

**FIGURE 3.1** Virtual joints representing attitude  

**FIGURE 3.2** Approaches to solving control problems  

**FIGURE 3.3** Support angle before and after disturbance  

**FIGURE 3.4** Different types of terrain  

**FIGURE 3.5** The outline of the biped of Grishin et al. constrained to the sagittal plane  

**FIGURE 3.6** Side view outline of a robot with direct drive for the hip joint and belt drive for the knee joint  

**FIGURE 3.7** Frontal view outline of a robot with DC motor drive for the rotary joints and pneumatic cylinder drive for the prismatic joints  

**FIGURE 4.1** Biped hardware  

**FIGURE 4.2** Frontal and side views of the UNH biped  

**FIGURE 4.3** Biped metal supporting structure and motors, with joint names  

**FIGURE 4.4** Foot pressure sensors  

**FIGURE 4.5** Accelerometer beam  

xii
FIGURE 6.14  LOW LEVEL CONTROL OUTLINE 103
FIGURE 6.15  LOW LEVEL CONTROL SOFTWARE STRUCTURE 104
FIGURE 6.16  LOW LEVEL CONTROL ARCHITECTURE 107
FIGURE 6.17  FOOT WITH NUMBERED PRESSURE SENSORS - TOP VIEW 108
FIGURE 6.18  FOUR CASES WHEN THE FEET ARE NOT PARALLEL TO THE GROUND 111
FIGURE 6.19  REACTIVE LEAN ANGLE CONTROL ALGORITHM 115
FIGURE 6.20  DESIRED RIGHT-LEFT LIFT AND MEASURED ZMP_{RL} EARLY IN TRAINING WITH 12 CM STRIDE LENGTH (TOP GRAPH) AND AFTER 5 MINUTES OF TRAINING WITH 12 CM STRIDE LENGTH (BOTTOM GRAPH) 119
FIGURE 6.21  MEASURED ZMP_{FB} AND THE DESIRED RIGHT-LEFT LIFT EARLY IN TRAINING WITH 12 CM STRIDE LENGTH (TOP GRAPH) AND AFTER 5 MINUTES OF TRAINING WITH 12 CM STRIDE LENGTH (BOTTOM GRAPH) 120
FIGURE 7.1  RELATIONSHIP BETWEEN THE TIME PERIOD THE FOOT SPENDS IN THE AIR, AND THE LEAN ANGLE AND THE SIDEWAYS SPEED AT THE MOMENT THE FOOT IS LIFTED 125
FIGURE 7.2  IT IS IMPORTANT TO KEEP THE SUPPORTING FOOT UNDER THE UPPER BODY 131
FIGURE 7.3  TORQUE (T) ON "HIP" JOINT DUE TO BODY LEAN 133
FIGURE 7.4  OUTLINE OF THE PREPROGRAMMED SIDEWAYS SPEED PROFILE 139
FIGURE 7.5  BIPED GAIT USED BY THE UNIFIED WALKING CONTROLLER 139
FIGURE 7.6  OVERALL BIPED SYSTEM BLOCK DIAGRAM 142
FIGURE 7.7  SIMPLIFIED MODEL OF THE HIGH LEVEL CONTROL 143
FIGURE 7.8  HIGH LEVEL CONTROL ARCHITECTURE 145
FIGURE 7.9  CMAC 1 OUTPUTS A NEW CORRECTION AT THE BEGINNING OF EVERY EXTEND LEG PHASE. LOCAL MINIMA AND MAXIMA IN THE RIGHT-LEFT LEAN ANGLE CAUSE THE REEVALUATION OF THE CMAC 1 CORRECTION. 149
FIGURE 7.10  THE SIDEWAYS MOTION CONTINUES DURING SINGLE SUPPORT 152
FIGURE 7.11  THE EXTENSION OF THE LIFTED FOOT SLOWS DOWN AFTER IT CONTACTS THE FOOT. THIS REDUCES THE SHOCK CAUSED BY LANDING. 154
FIGURE 7.12  FOUR POSTURE{1} PARAMETERS 155
FIGURE 7.13  CONTROL DELAY 159
FIGURE 7.14  LOW LEVEL CONTROL OUTLINE 168

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 7.15 Low level control architecture

Figure 8.1 Right-left angles at the beginning of training

Figure 8.2 Right-left angles after 5 minutes of training

Figure 8.3 Right-left and front-back ZMP at the beginning of training

Figure 8.4 Right-left and front-back ZMP after 5 minutes of training

Figure 8.5 Right and left foot lift and the measured right-left and front-back angles for the sideways leaning speed of 7.1 °/sec

Figure 8.6 Right-left and front-back ZMP for the sideways leaning speed of 7.1 °/sec

Figure 8.7 Commanded and measured right-left angles, and the absolute value of the error for the sideways leaning speed of 7.1 °/sec

Figure 8.8 Commanded and measured right-left angles, and the absolute value of the error for the sideways leaning speed of 12.5 °/sec

Figure 8.9 Right and left foot lift and the RLCF for the sideways leaning speed of 10.7 °/sec

Figure 8.10 Right and left foot lift and the measured right-left and front-back angles for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec

Figure 8.11 Right and left foot lift and the measured right-left ZMP (ZMP_{RL}) for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec

Figure 8.12 Right and left foot lift and the measured front-back ZMP (ZMP_{FB}) for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec

Figure 8.13 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is decreasing.

Figure 8.14 Absolute value of the commanded sideways leaning speed and the measured value of the right-left ZMP (ZMP_{RL}). The sideways leaning speed is decreasing.

Figure 8.15 Absolute value of the commanded sideways leaning speed and the measured value of the front-back ZMP (ZMP_{FB}). The sideways leaning speed is decreasing.
Figure 8.16 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is increasing.

Figure 8.17 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is changing "randomly".

Figure 8.18 Right and left foot lift and the measured right-left ZMP (ZMP_{RL}) for the sideways leaning speed changing randomly.

Figure 8.19 Right and left foot lift and the measured front-back ZMP (ZMP_{FB}) for the sideways leaning speed changing randomly.

Figure 8.20 Right and left foot lift and the measured right-left and front-back angles for sideways leaning speed of 14.3 °/sec.

Figure 8.21 Right and left foot lift and the measured right-left ZMP (ZMP_{RL}) for sideways leaning speed of 14.3 °/sec.

Figure 8.22 Right and left foot lift and the measured front-back ZMP (ZMP_{FB}) for sideways leaning speed of 14.3 °/sec.

Figure 8.23 Left ankle drive signal. The absolute value of the sideways leaning speed is gradually increasing.

Figure 8.24 Left ankle-Y drive signal. The absolute value of the sideways leaning speed is gradually increasing.

Figure 8.25 Left knee drive signal. The absolute value of the sideways leaning speed is gradually increasing.

Figure 8.26 Left hip and hip-Y drive signal. The absolute value of the sideways leaning speed is gradually increasing.

Figure 9.1 Front-back angle dip when the foot is lifted.

Figure 9.2 Torque around the ZMP created by the lifted foot.

Figure 9.3 Left foot lift and RLCF_R showing the biped rocked up on the right edge of the right foot.

Figure 9.4 Left knee drive signal and the right foot lift. The gait speed is gradually increasing.

Figure 9.5 The relationship between the maximum sideways leaning speed and the walking velocity is nearly linear.

Figure 9.6 Empirical probability p(v_t) that the biped will fall during a step for a given sideways leaning speed, v_t.
| Figure A.1.1 Virtual sensor       | 229 |
| Figure A.2.1 Motion of the link connecting the ankle and the knee joints during leaning | 235 |
| Figure A.2.2 Foot drift due to the rotation around the Z-axis | 237 |
| Figure A.3.1 The UNH biped walking   | 239 |
LIST OF TABLES

Table 5.1 List of posture parameters 65
Table 6.1 CMAC input vector elements 96
Table 7.1 Training errors for the two CMAC 5 outputs during the six gait phases 163
Table 7.2 CMAC input vector elements 166
Table 8.1 Walking parameter settings 172
Table 8.2 Allowed pairs of sideways leaning speed and foot lifting speed 172
Table 8.3 Settings before change 174
Table 8.4 Parameter settings for Figures 8.5 through 8.9 177
Table 8.5 Average stride length ($\bar{L}$) and average walking velocity ($\bar{v}$) measurement results for three trials ($T_1$, $T_2$, and $T_3$) 194
Table 8.6 Number of steps taken and corresponding number of falls 195
Table 9.1 Empirical probability $P(v_i)$, that the biped will fall during a step for a given sideways leaning speed, $v_i$ 215
Table A.4.1 Record of how the different settings were changed with time for one trial of the variable gait speed experiment 241
LIST OF ACRONYMS

CF - CENTER OF FORCE
CMAC - CEREBELLAR MODEL ARITHMETIC COMPUTER
FBCF - FRONT-BACK CENTER OF FORCE
FBCF_L - FRONT-BACK CENTER OF FORCE ON THE LEFT FOOT
FBCF_R - FRONT-BACK CENTER OF FORCE ON THE RIGHT FOOT
FL_i - READINGS FROM PRESSURE SENSORS ON LEFT FOOT (i = 1, ..., 4)
FR_i - READINGS FROM PRESSURE SENSORS ON RIGHT FOOT (i = 1, ..., 4)
H_HP - HIGH-PASS FILTER TRANSFER FUNCTION
H_LP - LOW-PASS FILTER TRANSFER FUNCTION
I/O - INPUT AND OUTPUT
NPCM - NORMAL PROJECTION OF THE CENTER OF MASS
PC - PERSONAL COMPUTER
PCB - PRINTED CIRCUIT BOARD
PD - PROPORTIONAL AND DERIVATIVE (CONTROL)
PID - PROPORTIONAL, INTEGRAL AND DERIVATIVE (CONTROL)
PWM - PULSE WIDTH MODULATION (OR MODULATED)
RLCF - RIGHT-LEFT CENTER OF FORCE
RLCF_L - RIGHT-LEFT CENTER OF FORCE ON THE LEFT FOOT
RLCF_R - RIGHT-LEFT CENTER OF FORCE ON THE RIGHT FOOT
SH - SECONDARY HYPOTHESIS

TCF - TARGET CENTER OF FORCE

ZMP - ZERO MOMENT POINT

ZMP_{FB} - FRONT-BACK COMPONENT OF ZERO MOMENT POINT

ZMP_{RL} - RIGHT-LEFT COMPONENT OF ZERO MOMENT POINT
ABSTRACT

A SENSORY-BASED ADAPTIVE WALKING CONTROL ALGORITHM FOR VARIABLE SPEED BIPED ROBOT GAITS

by

Andrew L. Kun
University of New Hampshire, May, 1997

A balance scheme for handling variable speed gaits was implemented on an experimental biped. The control scheme used pre-planned but adaptive motion sequences in combination with closed loop reactive control. CMAC neural networks were responsible for the adaptive control of side-to-side and front-to-back balance. The biped performance improved with neural network training. The biped was able to walk with variable speed gaits, and to change gait speeds on the fly. The slower gait speeds required statically balanced walking, while the faster speeds required dynamically balanced walking. It was not necessary to distinguish between the two balance modes within the controller. Following training, the biped was able to walk with continuous motion on flat, non-slippery surfaces at forward progression velocities in the range of 21 cm/min to 72 cm/min, with average stride lengths of 6.5 cm.
1. INTRODUCTION

1.1 Problem Identification

The research presented in this dissertation deals with biped robot walking. The first question on most people's mind when thinking about bipedal walking robots is what the purpose of such robots may be. The answer is that walkers are much better suited for many types of terrain than wheeled vehicles, therefore if we want robots to traverse these types of terrain it makes sense to build biped robots or robots with more than two legs. One example is rocky terrain, where wheeled vehicles have difficulty operating. Humans can walk over rocky terrain with relatively little difficulty. Certain environments are created with human bipedal walking in mind. Therefore, bipedal walking robots would be well suited for work in such environments. For example, service robots that would do work in buildings could be bipedal walkers.

A biped control scheme of practical relevance would most likely have to be able to control bipedal gaits for different gait speeds. The term "gait speed" describes the stepping rate of the biped and the time the lifted foot stays in the air, that is the "single support period". Faster gaits have higher stepping rates with shorter single support periods, while slower gaits have lower stepping rates and longer single support periods. A practical control scheme would have to make it possible to change gait speeds on the fly, while the biped is walking. For example, service robots walking in buildings would
have to make stops, avoid obstacles, and perform various tasks that require different gait speeds and the ability to switch between these gait speeds in real-time.

Different bipedal gait speeds correspond to two different walking gaits: the \textit{statically balanced}, and the \textit{dynamically balanced} gait. Very slow walking requires a statically balanced gait, while faster walking requires a dynamically balanced gait. A biped walking with a statically balanced gait is said to have \textit{static balance}. The gait has the property that if the walking motion is frozen at any instant of time, the biped is stable. This is achieved by keeping the normal projection of the robot's center of mass (NPCM) within the limits defined by the biped feet, while moving slowly enough that the biped dynamics can be ignored. When only one of the two feet is in contact with the ground, the NPCM has to be within the area of that foot. When both feet are on the ground the NPCM has to be within the polygon determined by the outer corners of the biped feet. Both cases are illustrated in Figure 1.1, where the above described regions are hashmarked.

The second walking gait is the dynamically balanced gait. A biped walking with a dynamically balanced gait is said to have \textit{dynamic balance}. In the case of dynamic balance the NPCM is permitted outside the boundaries described above. If the NPCM is outside these boundaries gravity will tend to make the biped fall over. Note that this effect of gravity can be counteracted by joint torques. However, the biped may be falling during parts of the gait, and unless the feet are controlled correctly, it could fall on the ground.
The UNH Robotics Laboratory has been involved in biped robot walking research for a number of years. Over these years the Laboratory has produced increasingly sophisticated versions of its two legged walker. Every version of the control algorithm of the UNH biped was designed to control walking for a small range of walking gait speeds, and it implemented dynamic balance. Trying to slow down the gaits created insurmountable problems for the implemented control algorithms, because they were not built to handle statically balanced walking. The controllers of biped robots built at other research institutions had the same characteristic - they all provided stable control for a small range of gait speeds, either by providing static balance or dynamic balance control.

The problem inspiring this dissertation was that there was no scientific knowledge available that would enable researchers to create a biped walker that could walk with variable speed gaits. More specifically, existing knowledge was not sufficient to devise a controller that would:

Figure 1.1 *The NPCM has to be within the stability region in the case of static balance.*
• enable a biped to walk at a range of different gait speeds, with the slowest possible speed implementing statically balanced walking, and the faster speeds implementing dynamically balanced walking;

• enable the biped to change gait speed on the fly such that, within the range of attainable speeds, the biped would be able to switch from any gait speed to any other gait speed, at any given instant of time.

In order to simplify the problem of this research the author decided to consider only the case of the UNH biped walking on a hard, flat, non-slippery surface. It seemed reasonable to expect that a new controller, that would enable the biped to walk at variable speeds, will not perform perfectly - the existing controllers for the UNH biped were fairly reliable, but if the biped walked for a long enough time it would eventually fall. The reason for this imperfection was that the biped did not have a mechanism for quickly correcting controller errors, or a mechanism for disturbance rejection. Every so often the adaptive controller makes mistakes, and these mistakes are not corrected by some extraordinary action that would correspond to a human kicking out sideways to prevent falling to the opposite side. Disturbances are caused mostly by the uneven floor the biped is walking on. The biped’s mechanical structure also introduces non-repetitive disturbances in the form of slop in the joints, gear stiction, shifting of the foot plates, flexing of the links, etc. Some combinations of these disturbances cause falls. However,
most of the time the biped can walk considerable distances without falling, for example 5-6 meters. The new controller was required to reach a similar level of reliability.

1.2 Research Goal

The main research goal was to create a new controller that would allow the UNH biped to walk with variable speed gaits. The goal was to build the controller without detailed knowledge of the biped dynamics. Such a controller could then be used as a basis for similar controllers of more complex biped robots in future research efforts. We can call this control algorithm the unified biped walking control algorithm, referring to the fact that it unifies the control of high and low speed gaits.

The main goal of the research can be broken down into the following specific aims:

- analyze the static and dynamic walking gaits and identify their important features;
- compare the features of the static and the dynamic gaits to find similarities between them that could be used to create a unified control scheme;
- formulate hypotheses in order to tackle the main problem;
- theoretically develop the new controller;
- implement a unified biped walking control algorithm on the UNH biped that has the following characteristics:
  - walking is implemented for speeds in the range of 25 cm/min to 100 cm/min;
  - the gait speed can be changed on the fly;
  - the biped can walk 5-6 m without falling;
• test the performance of the new controller in order to verify that it meets the above requirements;
• evaluate the testing procedure to confirm that the tests are relevant and that the test results are valid.

1.3 Dissertation Organization

The second chapter of this dissertation introduces the major research hypotheses, and the research approach. The third chapter gives a background on biped walking research. The fourth chapter introduces the hardware of the UNH biped that was used in the walking experiments. The fifth chapter describes the dynamic gait controller that was implemented during previous research at the UNH Robotics Laboratory. Even though the dynamic gait controller was not implemented as part of this research it was considered important that the reader be familiar with its main points, because the design of the unified walking controller was partly based on experiences gained from working on the dynamic gait controller. The static gait controller is introduced in the sixth chapter. The unified biped walking gait controller, which was designed based on experiences with the dynamic and the static gait controllers, is presented in the seventh chapter. The eighth chapter presents the results obtained through several trials of different walking experiments. The ninth chapter discusses the results and addresses their relevance and validity. The tenth chapter gives the conclusions of the research, while the eleventh chapter lists several ideas for future research.
2. HYPOTHESES AND RESEARCH APPROACH

In this section the major hypotheses of the dissertation are presented along with the research approach that was used to develop the unified biped walking control algorithm and to verify the hypotheses.

2.1 The Main Hypothesis

In Section 1.2 we stated that the main goal of this research was to create a controller that would allow the UNH biped to walk with variable speed gaits. The main hypothesis of the research proposed the following. If the static and dynamic walking gaits were analyzed and their important features were identified, one could find similarities between the two gaits, and also identify features that would cause problems if the biped was required to walk with variable speed gaits. This knowledge could then be used to create a new unified biped walking control algorithm that would allow the biped to walk at a range of gait speeds, and switch from any gait speed to any other gait speed within that range, at any given instant of time. The slow speeds would be implementing static walking, and the faster speeds dynamic walking.

2.2 The Second Hypothesis

Analysis of the similarities between the features of the static and dynamic walking gaits led to the formulation of the second hypothesis. As we said in Section 1.1, the statically
balanced gait has the property that if the walking motion is frozen at any instant of time, the biped is stable. This is achieved by keeping the normal projection of the robot's center of mass (NPCM) within the limits defined by the biped feet, while moving slowly enough that the biped dynamics can be ignored. When only one of the two feet is in contact with the ground, the NPCM has to be within the area of that foot. When both feet are on the ground the NPCM has to be within the polygon determined by the outer corners of the biped feet. We can call the above regions the "stability regions". Both cases are illustrated in Figure 2.1, where the regions described above are hashmarked.

The second walking gait is the dynamically balanced gait. In the case of dynamic balance the NPCM is permitted outside the boundaries described above. If the NPCM is outside these boundaries gravity will tend to make the biped fall over. The biped may be falling during parts of the gait, and unless the feet are controlled correctly, it could fall on the ground. The "Zero Moment Point" (ZMP) is the point where the sum of all moments is equal to zero [1]. If the dynamically balanced gait is constrained such that the supporting foot has to be flat on the ground, then the ZMP has to stay within the boundaries hashmarked in Figure 2.1, that is within the "stability region". If the ZMP left the stability region, the contact foot would rotate around one of its edges. We can call dynamic balance with the above constraint full foot contact dynamic balance. Since, in the case of static balance, the NPCM and the ZMP are located in the same point, static balance is a special case of full foot contact dynamic balance.
The second hypothesis of this research was that the property of both static walking and full foot contact dynamic walking that the ZMP has to be within the stability region, as illustrated in Figure 2.1, can be used to create a unified biped walking control algorithm. The hypothesis proposed that the position of the ZMP can be measured, and that the control algorithm should position the ZMP such that the biped is stable. Since the ZMP has to be in the same area for all gait speeds, the controller should not need to distinguish between static and dynamic walking.

2.3 The Third Hypothesis

Let us now look at the third hypothesis of the dissertation. In previous research at the UNH Robotics Laboratory, as well as in research efforts at other institutions, it was found that the control of the balance of two-legged walking machines is difficult for several reasons:

- If the feet are small the system is unstable.

Figure 2.1 Stability region: Position of the NPCM in the case of static balance, and the ZMP in the case of full foot contact dynamic balance.
• Time delays in the control loop amplify stability problems.

• The system nonlinear dynamics and kinematics are difficult to model accurately, and simplified models are generally not adequate.

• Other significant properties like gear stiction, gear play, and foot flex, are difficult to model accurately.

• Since the robot has no direct connection to an inertial frame of reference, the controller must rely on often noisy sensors (foot-force sensors and accelerometers, for example) to represent the relationship between the robot and the external environment.

In other words, biped robots are usually unstable systems with poorly defined dynamics and the information available about the state of the biped comes from noisy sensors. As will be explained in more detail in Section 3.1.2, the problem of controlling such a system can be approached using two major methods. One is to try to create an accurate mathematical model of the robot, and to use classical or modern control techniques to achieve stable walking. The second approach is to use adaptive techniques in the control strategies, in which case the mathematical model of the system can be greatly simplified. The UNH biped has many degrees of freedom and many sensors. It seemed impractical to try to formulate an accurate mathematical model describing the system. The third hypothesis of this research proposed that the new control algorithm could be made adaptive, and that it could improve its performance over time based on feedback from the biped sensors.
2.4 The Fourth Hypothesis

The analysis of experiences gained during the implementation of the dynamic walking controller for the UNH biped led to the formulation of the fourth hypothesis of this dissertation. As a result of the distribution of mass within the biped structure, in order to lift a foot using a dynamic balance gait, it is necessary to first generate lateral momentum toward the opposite side. The foot can then be lifted and moved to a new location. This results in a gait where the biped leans from one side to the other and lifts up the feet when it leans over a sufficient amount, as shown in Figure 2.2. During walking the biped goes through a succession of alternating single support phases (when only one foot is on the ground) and double support phases (when both feet are on the ground). In single support phase the biped gait can roughly be modeled as the motion of an ideal point-mass inverted pendulum, as outlined in Figure 2.2 by the dotted lines and the gray circles.

![Figure 2.2 Biped gait (front view)](image)
Figure 2.3 shows an idealized model of the biped in the frontal plane. The biped is modeled as an inverted pendulum of constant height of 0.5 m, with no mass in the legs and zero width feet. The biped has to lean over to one side in order to lift up the other foot. Let us define the variable "lean at foot lift" as the lean at the moment the foot is lifted up, and the variable "v" as the sideways leaning speed at that moment.

![Idealized biped model](image)

When we program this idealized two-dimensional biped for walking, we can preset the desired amount of time the foot should spend in the air. Let us define the variable "foot lift period" to represent this time. Figure 2.4 shows the relationship between the "foot lift period" and the "lean at foot lift", depending on the value of the "v". In order to keep the foot in the air for a certain amount of time we have to pick corresponding values of the "lean at foot lift", and of "v". The error in the achieved "foot lift period" due to an incorrectly implemented "lean at foot lift" is proportional to the slope of the curves.
The error due to an incorrectly implemented "\(v\)" is proportional to the distance between the curves. Figure 2.4 shows that for larger values of the "foot lift period" the curves have a larger slope and they are farther apart, than for smaller values of the "foot lift period". Therefore, errors in implementing the "lean at foot lift" and/or "\(v\)" will cause a larger error in the "foot lift period" in the case of static walking, which corresponds to a longer "foot lift period", than in the case of dynamic walking. Consequently, static walking requires a more accurate control of the biped’s instantaneous position and speed than dynamic walking.

Figure 2.4 Relationship between the time period the foot spends in the air, the initial lean angle, and the sideways leaning speed

The dynamically stable walking control of the previous versions of the UNH biped worked for relatively small values of the "foot lift period" (less than 0.5 seconds). Trying to slow down the gait, that is to increase the "foot lift period" created serious

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
problems for the controller. The controller was implemented by depending primarily on pre-planned, but adaptive motion sequences, rather than on reactive closed-loop control (as explained in detail in Chapter 5). The controller tried to control the biped’s position and speed perfectly by only using adaptive elements which learned over time, but it did not provide immediate correction when the errors occurred. The fourth hypothesis of this research is that, in order to be able to achieve static as well as dynamic walking, the unified bipedal walking controller should be based on adaptive closed-loop control. The control output would have a pre-planned, but adaptive, component, and a reactive component. The algorithm would be able to improve the walking performance over consecutive steps, using adaptation. It could also react to small errors of the controller and some disturbances, using reactive control.

2.5 Research Approach

With the major hypotheses formulated the author could turn his attention to defining the research approach and the main research stages that would lead to a unified biped walking control algorithm.

Previous versions of the UNH biped implemented dynamic walking. In the process of implementing a unified controller it seemed reasonable that, after gaining a good understanding of dynamic walking, one would need to gain experience with static walking. Static walking would always in fact be quasi static, since even when the biped is moving very slowly the dynamic effects cannot be completely ignored. Adaptation would have an important role in this gait. In order to implement an adaptive controller,
as proposed by the third hypothesis, the new controller could use CMAC neural networks [2, 3, 4]. The following two reasons gave the author confidence that CMAC neural networks can successfully be used in the implementation of the new unified controller:

- In previous successful versions of the UNH biped [5, 6, 7, 8, 9] the control of dynamically balanced walking was based on adaptive CMAC neural networks;
- The experiences from the previous versions of the controller that used adaptive CMAC neural networks were available.

Next, using the experiences from the dynamic and static walking controllers, the unified controller could be designed. Following the second hypothesis of the dissertation the unified controller could be based on the principle of keeping the biped stable by always keeping the ZMP within the stability area, as shown in Figure 2.1.

The new controller would have to be tested to show that it meets the performance and reliability criteria set forth in Chapter 1. The test procedures would have to be verified themselves in order to prove that they are relevant and that the results they produced are valid. Once all this was successfully done we would be in possession of new scientific information that would confirm the hypotheses and solve the problem that started this research.

Therefore, the major elements of the research approach were:
1. Design the static walking controller, using pre-planned, adaptive motion sequences combined with reactive control. The adaptive control can be based on CMAC neural networks.

2. Analyze the new static gait controller and the previously implemented dynamic gait controller, and identify their important features and parameters.

3. Use the results of the analysis of the static and dynamic walking gait controllers to find how the position of the ZMP can be controlled such that the biped is stable for different gait speeds. Based on this method design the unified walking controller.

4. Confirm that the new controller meets the performance and reliability requirements set in Chapter 1.

5. Verify the relevance and the validity of the testing procedures.

The elements of the research approach are shown in block diagram form in Figure 2.5. The blocks on the left represent the research approach. The blocks on the right show the hypotheses that would be used in the individual steps of the research approach.
Figure 2.5 Research approach and hypotheses
3. BACKGROUND

The first experimental biped was built in 1973 by Kato and his colleagues [10]. Since then many researchers have worked on the problem of bipedal robot walking. Their efforts can be classified according to the:

- control goal of the research;
- control method used to achieve bipedal walking;
- desired biped walking gait;
- verification method used in the research.

In the following sections the above classifications are discussed, and some specific research efforts are introduced. Some issues of building an experimental biped are also presented.

3.1 Research Effort Classification Methods

This section describes four classifications of research efforts in the field of bipedal walking.

3.1.1 Classification According to Control Goals

Researchers have addressed a wide variety of control goals for biped walkers. The majority of projects concentrated on questions of balance during walking, hopping, or
running at a constant speed on flat, hard, non-slippery surfaces. Problems related to starting and stopping, that is problems associated with variations in the speed of progression, have also been addressed.

Research has also been done on disturbance rejection. Disturbances are external factors that cause unwanted changes of the robot joint positions, and/or the angle between the supporting foot (feet) and the ground. The most common approach to disturbance rejection is to come up with schemes that compensate for the above changes. Another approach contends that it will not always be possible to compensate for a given disturbance, therefore it is more useful to design a biped that can recover from disturbances that cause falls.

Another control goal addressed by researchers is real-time adaptation to environmental changes, one example being adaptation to a change in the slope of the walking surface. This control goal is similar to the disturbance rejection goal, since environmental changes can be thought of as disturbances. The difference between disturbances and environmental changes is their duration: disturbances are short impacts, while environmental changes influence walking for long periods of time. Note that adaptation in this case does not necessarily mean the use of adaptive control, or neural networks. It includes actions like switching to a different, precalculated mode of walking to deal with environmental changes.
Since controlling a biped is a complicated task, some researchers have decided to tackle the problem by dividing it into two sub-problems: balance in the sagittal plane, and balance in the frontal plane. Several projects concentrated on balancing in only one of the planes, while assuming that balance in the other plane was achieved.

3.1.2 Classification According to Control Methods

The biped control problem can be approached using two major methods. One is to try to come up with a mathematical model of the robot, and to use classical or modern control techniques to achieve stable walking. Using the Newton-Euler equations the biped dynamics can be expressed as:

\[ D(q) \ddot{q} + C(q, \dot{q}) + g(q) = \tau, \quad (3.1) \]

where \( q = \left[ q_0, q_1, \cdots, q_n \right]^T \) is the \( n \times 1 \) vector of generalized joint displacements, \( \tau \) is the \( n \times 1 \) vector of applied generalized forces, \( D(q) \) is the \( n \times n \) mass matrix, \( C(q, \dot{q}) \) is an \( n \times 1 \) vector of centrifugal and Coriolis terms, and \( g(q) \) is an \( n \times 1 \) vector of gravity terms [11]. Note that in order to express the attitude of the robot, it is necessary to introduce three virtual joints with no mass, and no length, as shown in Figure 3.1, and described in [12]. Therefore a biped with \( m \) joints will have \( n = m + 3 \) generalized joints. The values of the applied generalized forces corresponding to the virtual joints are zero.
The above approach has two problems. The first problem is that the more complicated the biped system is, the more difficult it is to create a sufficiently accurate mathematical model. If the model is not accurate enough the control scheme is not going to be successful. The second problem is that control computations using mathematical models can be very time consuming, making them impractical for the real-time task of balancing a biped.

In order to solve the problems associated with mathematical modeling some researchers decided to use *adaptive techniques* in their control strategies. This approach ideally eliminates the above two problems. In the presence of adaptation the necessary mathematical model can be simple, and therefore relatively easy to design and implement in real-time on a computer. At the same time efficient adaptation techniques are available.
(for example different neural networks) which do not increase the computational load beyond the capabilities of modern computers. Researchers have incorporated adaptive techniques in their control schemes in two major ways. One is to use adaptation to estimate the parameters of the mathematical model of the robot, and then use the mathematical model to control the robot. The other way is to set up target outcomes of control that reflect stable walking, and adjust biped actions to reach the target outcomes. The two approaches can be demonstrated with the following two examples.

**Example 3.1** A biped control scheme is designed using the mathematical model of the biped. When the control scheme derived using Equation 3.1 is implemented on the physical biped the biped cannot be balanced. The researcher observes that the model matrices $D(q), C(q, \dot{q})$, and $g(q)$ were formed using estimates of the real biped parameters. This leads the researcher to the conclusion that the estimates used in the model are not accurate enough. Adaptive techniques can be used to improve the estimates of biped parameters. Once better estimates of these parameters are acquired the control law derived from Equation 3.1 will produce better results.

**Example 3.2** Let us assume that a control law is derived for the biped from Example 3.1 using Equation 3.1. Again, after implementing the control scheme on the physical biped, the biped cannot be balanced. This time a different kind of adaptation can be used. For example the researcher can set up a target value for the position of the normal projection of the center of mass, and adapt the values of
To achieve this target value. The resulting control scheme is going to be a combination of a fixed controller based on an approximate mathematical model of the biped, and an adaptive controller continuously adapting the joint displacements.

The different approaches to the biped control problem are summarized in Figure 3.2.

![Figure 3.2 Approaches to solving control problems](image)

3.1.3 Classification According to Bipedal Walking Gaits

The two biped walking gaits, the *statically balanced*, and the *dynamically balanced* gait, were introduced in Sections 1.1 and 2.2. The static gait has the property that if the walking motion is frozen in any instant of time, the biped is stable. In the case of dynamic balance the biped may be falling during parts of the gait, and unless the feet are controlled correctly, it will fall on the ground.
In terms of achieving walking, static balance can always be achieved if one is willing to give up walking speed, and increase foot size. Dynamic balance enables the biped to walk faster, and more efficiently.

3.1.4 Classification According to Result Verification Methods

Some research results have only been verified through computer simulations. These results are valuable because they can point the researchers in the right direction. However most of them cannot substitute results obtained using experiments with physical bipeds, because the biped models used in the simulations are usually not accurate enough.

3.2 Research on Bipedal Walking

This section presents several research efforts in the field of bipedal walking. The works are presented with the classifications discussed in Section 3.1 in mind: subsections introduce projects that illustrate important elements of the four classifications.

3.2.1 Projects Illustrating Classification According to Control Goals

As mentioned in Section 3.1.1, most researchers concentrate on biped walking on flat surfaces, at constant speeds. A common targeted biped behavior is steady walking without the presence of disturbances. One example is the 1993 work of Yamaguchi, Takanishi, and Kato on the WL-12RV robot [13]. In 1986 the authors developed the WL-12 biped, which had a trunk that could be used to stabilize walking. The trunk was used to compensate for the pitch and roll-axis moments about an arbitrary, planned Zero Moment Point (ZMP) [1]. The WL-12RV was an improved version of the WL-12, with
the capability to compensate for the pitch, roll, and yaw-axis moments about an arbitrary planned ZMP. The dynamic walking robot had nine degrees of freedom, a mass of 103.5 kg, and was 180 cm tall. The implemented control method precalculated ZMP paths, and consisted of two main parts. One was a walking control algorithm using preset walking patterns to control the legs. The other part calculated the motion of the trunk. This calculation was based on a mathematical model that represented the robot as a system of particles. The authors evaluated the results of the research through simulations, and walking experiments. Both supported the effectiveness of the control method for walking on a flat floor.

Dunn and Howe in their 1994 article [14] reported on designing a biped, with the control goal of achieving smooth bipedal walking on flat terrain. They modeled the robot as an inverted pendulum, and concentrated on the phase of the gait when the legs exchange roles, that is the swinging leg becomes the supporting leg and vice versa. They called this phase of the gait the exchange-of-support (EOS) phase. The authors proposed that smooth walking can be achieved by providing for a smooth EOS phase. They consequently derived kinematic conditions that ensure that the biped body does not undergo instantaneous changes in velocity during EOS. The article presented brief experimental results that showed the four degree-of-freedom, 5.6 kg robot performing dynamic walking.

Many research efforts had more complicated control goals. One of them came from Zheng, who in his 1987 work [15] proposed two disturbance rejection methods. Zheng
considered the case when the robot is supported by one foot, and defined the angle between this foot and ground as the support angle, \( p \). Ideally \( p = 0 \), but in the presence of disturbances \( p > 0 \), as shown in Figure 3.3. The author considered the case when the disturbance is small, meaning that the robot would stabilize itself without compensation, however compensation is still desirable. The first method he proposed was the so called \textit{velocity-compensation} method. In this method \( dp/dt \) is compensated using joint angular velocities. The result is that in the horizontal direction the vertical projection of the center of gravity keeps running through the middle of the supporting foot. The second method is the \textit{acceleration-compensation} method. This method eliminates the effects of small disturbances by compensating \( d^2p/dt^2 \) using joint accelerations. Zheng verified his findings through computer simulations of the SD-2 robot previously built by his group.

\[\text{Figure 3.3} \quad \text{Support angle before and after disturbance}\]

Disturbances can be caused by external forces, but they can also be caused by slipping. Boone and Hodgins in their work published in 1995 considered the control problem of creating reflexive responses to slipping in bipedal running robots [16]. One of the
complicating factors was that slipping could only be detected after it had occurred. This meant that it could not be avoided by placing the feet correctly using some advance knowledge of spots with low friction coefficients. The authors evaluated two kinds of responses: one-step, and two-step strategies. In one-step strategies the correction was applied in the slip step, and in two step strategies it was applied in the following step. The one-step strategies proved to be inefficient for low friction coefficients, while the two step strategies worked even for friction coefficients around 0.05. However, the responses that continued the slipping step produced smoother recoveries, in the range that they were effective in, than the responses that abandoned the slipping step. The authors came to the above conclusions by running simulations of a biped with telescopic legs. The simulated biped was based on a planar robot constructed by Raibert et al. [17]. Each leg had three degrees of freedom at the hip and a fourth degree of freedom for the length of the leg.

Ideally a biped would be able to walk on numerous types of terrain, just as humans can. However today's bipeds are far from this goal. Some researchers worked on bipedal walking on slopes, and on so-called horizontally composed planes (these surfaces consist of horizontal surfaces connected with vertical surfaces), as shown in Figure 3.4. Work has also been done on gait adaptation, where bipeds adapt their gait to changing terrain conditions.
Zheng and Shen, in their article published in 1990, reported on devising a static motion strategy to enable the SD-2 robot to climb sloping surfaces with positive gradients [18]. The strategy included detecting and measuring the gradient of the slope, recalculating the flat-surface biped gait to make it suitable for walking on the slope, and a scheme for the robot to walk through the transition area joining the level ground and the slope. The proposed scheme was experimentally verified. Golden and Zheng, also in 1990, reported on a fixed static walking gait for the SD-2 robot to climb stairs [19]. They successfully verified the gait experimentally.

Yamaguchi, Takanishi, and Kato [20] in a 1994 work introduced a biped that could adapt to walking on horizontally composed planes. The authors improved on the WL-12RV biped mentioned above by designing and implementing a new shock absorbing foot mechanism. They called the improved version of the biped WL-12RVI. With the new feet the biped was capable of detecting variations of the floor height from the expected.
Using these observations the motion of the lower limbs was modified on a real time basis in order to *adapt* the gait to the change in the surface. The technique was tested in walking experiments, and it was found that the 184.25 cm tall, 110.6 kg mass robot could successfully adapt its gait to scale up to 12 mm steps of the horizontally composed planes.

Inaba, Kanehiro, Kagami, and Inoue in their article published in 1995 [21] took an approach that is completely different from the ones described above. They changed the focus from building a robot that will not fall to building a robot that will be able to stand up after falling. Their ape-like biped had two arms and vision, and it could perform static walking. If the biped fell on its face it could help itself up using its arms. If it fell on its back it could roll over so that it was face down, from which position it could get up on its own. The robot used a *remote brain*, that is high level control computations were performed off-board, and were relayed to on-board components via an RF system. The authors performed successful experiments in which the robot was able to stand up and continue walking after falling on the ground.

In Section 3.1.1 it was mentioned that several researchers have considered balance in only one plane. An example can be found in the article of Grishin, Formal'sky, Lensky, and Zhitomirsky, published in 1994 [22]. They designed a dynamic biped with telescopic legs. The biped motion was constrained to the sagittal plane by passive feet that extended in the frontal plane, as shown in Figure 3.5. The authors performed successful walking experiments with the two degree-of-freedom, 7.5 kg mass, 75 cm tall robot. Similarly the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
bipeds of Dunn and Howe [14], and of Mitobe et al. [23] were constrained to the sagittal plane.

Figure 3.5 The outline of the biped of Grishin et al. constrained to the sagittal plane

3.2.2 Projects Illustrating Classification According to Control Methods

The previous section describes a variety of problems related to biped walking that were taken on by researchers. This section concentrates on the control strategies employed to tackle walking problems.

Mitobe, Mori, Aida, and Nasu in their 1995 work [23] used nonlinear feedback to control the gait of their biped. Motion generation was reduced to the problem of controlling the position and velocity of the robot's center of gravity. The authors devised a mathematical model of the robot, and a control law, based on this model, that produced actuator inputs. After developing the control law, they considered the relationship between the control law and the mechanical design of the biped, and found that the control law was simplified by the proper choice of mechanical parameters. The biped was built to facilitate the
application of the control law, and thus the two problems associated with mathematical
model-based control, discussed in Section 3.1.2, were avoided. Experimental results
show that the four degree-of-freedom robot, which was constrained to the sagittal plane,
successfully performed dynamic walking.

Chaillet, Abba, and Ostertag in 1994 developed what they called the double dynamic
modeling technique [24]. This technique was an extension of the "classical" method of
developing dynamic models, using the Lagrange and Newton-Euler formulations. The
authors took into account the fact that there are losses in the power transition between
robot actuators and limbs due to imperfections in the gears. They also found that the
losses are different depending on the direction of power transition (from actuator to limb
or vice versa). The new model took both these factors into account. If the gears are
perfect the model reduces to the "classical" model. Therefore, the "classical" modeling
technique can be regarded as a particular case of double dynamic modeling. The
effectiveness of the new scheme was tested through simulations of a biped. The biped
was modeled both the "classical" way and using double dynamic modeling, and these
models were used in the control law. It was found that the new modeling method resulted
in superior performance both for oscillatory and non-oscillatory limb motion.

As mentioned in Section 3.1.2 purely mathematical model-based control may not provide
the desired results. In such a case adaptive methods can be used to improve system
performance. This is what Yang and Shahabuddin reported on, in their 1994 article [25].
The authors generated joint trajectories for their five degree-of-freedom robot and derived
a control law based on the dynamics of the biped. They also incorporated an adaptation law which used the tracking errors of the joints to compute parameter estimates for the control law. Simulation results showed that the tracking errors could be reduced using the adaptive scheme.

Kitamura and Kurematsu in their 1990 work [26] started out by modeling the biped as an inverted pendulum. However, they found that this model did not reflect the actual movement of the robot exactly. To cope with this situation they used a multi-layered neural network to augment the results of the calculations based on the inverted pendulum model. Simulations of stationary biped walking showed that the multi-layered neural network could be trained to provide the necessary correction to the output of the mathematical model-based calculations.

In 1993 Li, Takanashi, and Kato [27] reported on implementing a learning control scheme for their WL-12RH biped. The robot's lower limbs followed precalculated trajectories, while the trunk was responsible for balance control. The commanded trunk motion consisted of a preset part and an adaptive part. The adaptive part was calculated using the difference between the measured and desired positions of the Zero Moment Point [1]. The effectiveness of the approach was experimentally supported.

The 1994 work of Stitt and Zheng [28] showed the application of distal learning [29] to biped walking. Distal learning is used to adapt a gait designed for walking on a flat surface to walking on a slope. The method used the forces of interaction between the
biped feet and the floor to determine whether the biped is balanced or not. The authors derived the trajectories for the interaction forces for a stable gait, and used these values as the target values in the distal learning approach. Computer simulations confirmed the validity of the approach.

In 1992 Salatian and Zheng proposed neural network-based gait synthesis methods for a biped climbing sloping surfaces [30]. They used a network consisting of twenty reciprocally inhibited and excited neurons, and implemented unsupervised learning rules using reinforcement signals. The authors have implemented two learning schemes. In the first one, which they called static learning, the neurons were trained only at five points (called primitive points) during a complete step. In the second scheme, called dynamic learning, the neurons were trained continuously during walking. The two schemes were verified through simulations of the SD-2 robot. In 1995 Yi and Zheng reported on improving the above mentioned schemes, and implementing them on the physical SD-2 biped [31]. The improvements included redefining the roles of some of the neurons, increasing the number of primitive points to eight, and changing the number of neurons from twenty to twenty-four. The static learning scheme was successfully implemented and experimentally tested on the physical robot. The dynamic learning scheme from [30] was redesigned into a new scheme called pseudo dynamic learning. In pseudo dynamic learning the neurons were trained only for the first half of the time spent between primitive points. This change was needed because dynamic learning sometimes produced jerky joint motion. The pseudo dynamic learning scheme was successfully implemented and tested on the SD-2.
3.2.3 Projects Illustrating Classification According to Bipedal Walking Gaits

Researchers often have to choose static or dynamic gait for their applications on the basis of the control goals, and environmental constraints. A good example of this is the 1990 work of Golden and Zheng [19], mentioned in Section 3.2.1. The control goal of their research was to synthesize a gait for the SD-2 robot to climb stairs. They found that using a dynamic walking gait would mean that the center of gravity (COG) would move through a series of arcs, with radii determined by the leg length. Such a motion of the COG would not be energy efficient, because the COG would constantly move up and down. The authors argued that in a static gait the COG would still move up and down, but the length of the strides would be shorter and therefore this gait would be more energy efficient. They also reasoned that a static gait is more stable than a dynamic gait, and therefore more suitable for such a complicated application as stair climbing.

One of the first dynamic walking bipeds was built by Miura and Shimoyama. In their 1984 work [32] they reported on two dynamic walkers, BIPER-3 and BIPER-4. The control law was designed by approximating the motion of the robots in the single support phase to the motion of an inverted pendulum. The authors conducted successful walking experiments with the two robots. A more recent research effort that produced a dynamic walking biped is presented in the dissertation of Benbrahim published in 1996 [33]. The author designed and built an experimental biped robot, and developed a reinforcement learning control architecture. The robot could learn how to walk without prior knowledge of its dynamics and with minimum user intervention.
3.2.4 Projects Illustrating Classification According to Result Verification Methods

As mentioned in Section 3.1.4, and as can be seen throughout Sections 3.2.1, 3.2.2, and 3.2.3, researchers use computer simulations and physical experiments to evaluate their work. This section describes two works where great emphasis was put on the evaluation method.

Computer simulations are often approximate, and cannot be trusted to predict the behavior of the physical system with complete accuracy. Fujimoto and Kawamura in their 1995 work [12] implemented a mathematically exact biped simulation model. The model is based on a combination of the methods of general manipulator simulation [34] and contact simulation of rigid body mechanics [35]. The model can be used as a low cost tool for investigating biped control algorithms. The authors used the model to test an autonomous walking control scheme that they designed.

Many researchers prefer to verify their findings using physical bipeds. Kajita and Tani in their article published in 1995 [36] reported on completing an experimental study of a control law for bipedal walking that they proposed in a previous article [37]. The authors identified two problems with applying the theoretical findings to a real biped robot. The first problem was that the theory assumed massless legs. The second problem was that the theory assumed that the body speed at the end of a particular support phase became the body speed of the next support phase after the support leg exchange. Two experiments were conducted to find out the impact of these imperfections. The goal of the support phase experiment was to find the horizontal dynamics of the biped with non-
zero mass legs. The experiment showed that stable walking could be achieved even if the masses of the legs were neglected in the control law calculations. The support exchange experiment was performed to investigate exchanges of leg support. It was found that smooth leg support exchange could be achieved by specifying a non-zero vertical foot velocity at the moment of contact with the ground. It was also found that creating a short two-leg support phase improved the smoothness of the exchange of support. Therefore, the experimental approach confirmed one of the assumptions of the theory (zero mass legs), and improved another aspect of it (body velocity constant over exchange of support).

3.3 Building an Experimental Biped

Several projects introduced in Section 3.2 used experimental bipeds. In the process of building experimental bipeds researchers have to address problems in several areas:

- biped mass distribution;
- actuators;
- sensors.

The biped will be easier to control if the mass is distributed such that the legs are light. There are two reasons for this. The first reason is that the weight of light legs could be neglected and the biped could be modeled as an inverted pendulum. This mathematical model could be used to design a simple control scheme. The second reason is that light legs could be moved quickly, facilitating quick responses to instabilities.
In reality most experimental bipeds have relatively heavy legs. The main reason for this is that bipeds usually use DC motor actuators mounted at the joints. A good example of such a design is Zheng's SD-2 robot [18]. One way to combat this problem is to mount the motors on the trunk, and use belt drives to move the leg joints. The four degree-of-freedom biped of Mitobe, et al. [23] has two DC motors to move the two hip joints, and two DC motors to move the knee joints via belts, as shown in Figure 3.6. This approach can be prohibitively complicated if the legs have many degrees of freedom.

![Figure 3.6 Side view outline of a robot with direct drive for the hip joint and belt drive for the knee joint](image)

Another way to tackle the problem would be to use lightweight DC motors. The problem here becomes the tradeoff between the weight, the available torque, and the price of the motor. The lighter the motor in a given price range, the less torque it can provide. Some researchers have tried to use actuators other than DC motors. Dunn and Howe built a four degree-of-freedom biped [14]. They used DC motors to power the rotary hip joints, and single-acting pneumatic cylinders combined with restoring springs to control the
length the prismatic legs, as shown in Figure 3.7. The authors managed to concentrate approximately 70% of the robot's mass in the upper body.

![Diagram of robot with DC motor drive for rotary joints and pneumatic cylinder drive for prismatic joints.]

*Figure 3.7* Frontal view outline of a robot with *DC motor drive for the rotary joints and pneumatic cylinder drive for the prismatic joints.*

Every experimental biped's control scheme relies on some sensory information to collect data about the biped's state. The control scheme usually needs information about the position of the joints. Joint positions can be determined using for example potentiometers or optical encoders. The angles between the biped and the ground are another piece of information utilized by some controllers. These angles can easily be calculated if the supporting foot is flat on the ground, but have to be measured if it is not. One way to perform the measurements is to use a gyroscope. Often the exact angles between the robot and the ground are not necessary and it is enough to determine whether a foot is in contact with the ground or not. In this case researchers can use a pressure sensor on the sole of the foot. If the control scheme requires information about the center of force on the foot, an array of force sensors can be attached to the sole of the foot.
Researchers have to decide on the type and number of sensors to use. A large number of sensors will provide the controller with a better picture of the biped state, however, the robot has to have enough processing power to utilize all the information.

### 3.4 Biped Research at the UNH Robotics Lab

In initial research at the UNH Robotics Lab concerning two-legged walking, standard supervised learning and temporal difference learning [38] were combined, in order to achieve gait adaptation for a simulated two-dimensional biped with massless legs [39]. This work focused on learning appropriate gait adaptations for achieving sudden body translational accelerations and decelerations, and for recovering from unexpected disturbance forces, starting from a model of steady walking. Similar issues were subsequently studied in the Robotics Lab by Latham [40], using a more realistic two-dimensional biped simulation which accounted for leg masses and foot/floor impact forces. His approach emphasized adapting gaits derived from an inverted pendulum model, in order to accommodate the unmodeled (in the controller) aspects of the biped dynamics. The adaptive walking control strategies developed initially in simulation were then tested and extended in a series of studies using two generations of experimental bipeds. An adaptive dynamic balance scheme was implemented and tested on the first generation of experimental bipeds by Miller [5, 6, 7]. The control scheme used pre-planned but adaptive motion sequences with sensory triggers, rather than closed loop reactive control. A phase-locked central pattern generator was used to conform to and make use of the natural dynamics of the biped. CMAC neural networks were responsible for the adaptive control of side-to-side and front-to-back balance, as well as for

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
maintaining good foot contact. The first generation biped robot was connected to off-board electronics through a heavy, non-elastic tether. This tether acted as a variable external disturbance when the biped was walking. A new hierarchical control hardware was designed and implemented for the UNH biped by Kun [41]. The new hardware eliminated the need for the heavy, non-elastic tether, which was replaced by a lightweight, flexible tether. The dynamic control scheme mentioned above was implemented on the second generation biped, by Miller and Kun [8, 9]. The implementation of the control scheme with the new hardware resulted in improved walking performance.
This chapter describes the UNH biped hardware shown in Figure 4.1. The hardware can be divided into low level and high level hardware. The walker’s metal supporting structure, its motors, the wiring, the battery, the sensors, and the on-board electronics make up the low level hardware. The high level hardware consists of a 200 MHz PentiumPro-based personal computer (PC), and a microcontroller board. The high- and the low level hardware are connected through a thin tether.

![Biped hardware diagram]

**Figure 4.1 Biped hardware**

The frontal and side views of the biped are given in Figure 4.2.
4.1 Low Level Hardware

The low level hardware consists of the robot’s metal structure, the motors, the wiring, the battery, the sensors and the on-board electronics. These will be discussed in more detail in the following sections. The biped sensors include two accelerometers and two gyroscopes. Readings from two gyroscope-accelerometer pairs are used in two virtual body angle sensors. These virtual sensors are implemented in software however, from the point of view of the control algorithm, they appear as hardware sensors. The operation of these virtual sensors will be explained in Section 4.1.3.

4.1.1 Biped Supporting Structure and Motors

The low level hardware is built around a metal supporting structure which consists of two legs, each with five joints, and a sheet metal box. The ten joints are powered by ten DC motors with optical encoders (one for every joint), as shown in Figure 4.3. In order to
make it easier to refer to specific joints, each joint was given a name - these are also shown in Figure 4.3.

![Figure 4.3 Biped metal supporting structure and motors, with joint names](image)

The legs are attached to the bottom of a horizontal metal bar in a symmetrical fashion, while the box is attached to the top of this bar. Joint angles can be changed by activating the DC motor of the given joint. The motors are 12V, 5700 rpm Globe Motors IM-15 gearmotors, with a 180:1 gear ratio, except the two motors in the knees which have a 256:1 gear ratio. Each motor has a Hewlett Packard HEDS-5500 optical encoder on its shaft. These optical encoders produce two pulse waves 180° out of phase, when the motor shaft is turning. The pulses can be counted, to determine the relative angle of rotation. Note that the absolute angle of rotation can only be determined if a reference angle is established at the beginning of the measurement. The resolution of the encoders
is 100 cycles per revolution of the motor shaft. The motors have drive inputs and encoder outputs. The drive inputs of the ten motors are connected to a single DB-25 connector, and the encoder outputs of the ten motors are connected to two DB-25 connectors.

4.1.2 Biped Sensors

The biped is equipped with pressure sensors on the feet, a pair of accelerometers, and a pair of rate gyroscopes.

Four polymer thick film force sensing resistors are mounted on the underside of both feet, near each corner (four 2.54 cm diameter sensors per foot). Each sensor is sandwiched between the upper metal foot plate and a 3 mm thick rubber square, which in turn is bonded to a semi-rigid Plexiglas and rubber bottom plate (Figure 4.4). Each of the sensors is wired up to a simple resistor circuit, with the outputs of these circuits connected to a DB-25 connector.

Figure 4.4 Foot pressure sensors

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Two piezoresistive accelerometers and two solid state rate gyroscopes oriented along orthogonal horizontal axes are mounted near the top of the body in order to provide two-dimensional body acceleration and rotation rate sensing. The accelerometers and the gyroscopes can be used to form two virtual sensors that can detect instantaneous biped body angles in the frontal and in the lateral planes - this is explained in detail in Section 4.1.3. The accelerometers used in the UNH biped are IC Sensors solid state accelerometers, model 3026. These accelerometers feature DC response, high sensitivity, and miniature size. They include a bracket for easy mounting. The acceleration range of the devices is ±2g, where $g = 9.81 \, \text{m/s}^2$. The rate gyroscopes are muRata Gyrostar piezoelectric vibrating gyroscopes. The gyros are miniature in size (they have a mass of 2.7 g), and their range is ±90°/s. The ranges of the accelerometers and gyros are considered satisfactory for the purposes of the task at hand.

4.1.3 Virtual Body Angle Sensors

The biped detects the body angles in the frontal and sagittal planes using two virtual body angle sensors. These virtual sensors process signals from a pair of accelerometers and a pair of gyroscopes.

Let us first look at how the biped’s accelerometers operate. The biped’s accelerometers measure the force acting on a very small beam, which is illustrated in Figure 4.5. The force is measured by sensing how much the beam bends.
The output of the accelerometer will be non-zero in the following cases:

1. if the beam's acceleration has a non-zero component perpendicular to the axis of the beam - this is the output component due to the real acceleration of the sensor;

2. if the axis of the beam and the true vertical form a non-zero angle, $\Theta$ - this is the output component due to gravity, and it depends on the orientation of the sensor;

3. if both of the above conditions are met (unless their contributions are of equal magnitude and of opposite signs).

Let us denote the output of the accelerometer as $<a>$, the acceleration perpendicular to the axis as $\ddot{z}$, and the angle between the axis of the beam and the true vertical as $\Theta$. The output of the accelerometer can then be expressed as:

$$<a> = C_1 \cdot \ddot{z} + C_2 \cdot \sin(\Theta),$$

where $C_1$ and $C_2$ are constants.
This creates a problem when we try to interpret the output of the sensors, because pairs of \((\text{acceleration}, \text{angle-to-vertical})\) values are not mapped uniquely to output voltages. What we would like to get from the accelerometer is the angle of the sensors with respect to the vertical. If the output of the accelerometer is low-pass filtered we can get an estimate of \(\Theta\). Low-pass filtering only yields an estimate because, the frequency contents of the component due to gravity, and the component due to true acceleration may overlap in the frequency domain. If we introduce the operator \(LPF\), which low-pass filters its argument, than the estimate of the angle of the sensor with respect to the vertical \(\hat{\Theta}\), can be expressed as:

\[
\hat{\Theta} = \sin^{-1}\left( \frac{1}{C_1} \cdot LPF(<a>) \right).
\]

If the beam is stationary the sensor will output a constant value, and this value will only depend on the angle between the beam and the vertical.

Next, let us look at how rate gyroscopes are used, without going into the details about the principles of their operation. Rate gyroscopes are solid state devices with an output that is ideally proportional to the rate of the rotational motion around the main axis of the device. When the gyroscope is at rest its output is usually not zero due to noise and inaccuracies of the sensor. This is a problem because, in order to calculate the position (orientation) of the sensor, we have to integrate the gyro's scaled output. To avoid integrating the non-zero offset of the output we can high-pass filter the gyro output. This
will leave us with the high-frequency components of the sensor readings, which correspond to the rotational motion.

The next question is how should the accelerometers and the gyroscopes be mounted if they are to be used in the measurements of the front-back and the right-left lean angle? In order to measure the steady state front-back and right-left lean angles of the biped the accelerometers are mounted in two orthogonal planes: the biped's frontal and sagittal plane. This is illustrated in Figure 4.6. The output of the accelerometer mounted in the frontal plane is used in calculating the front-back lean angle, while the output of the accelerometer mounted in the sagittal plane is used in calculating the right-left lean angle. The changes in the lean angles due to rotational motion are measured using rate gyros. The rate gyros are mounted such that they measure rotational velocities around two orthogonal axes, as shown in Figure 4.6.

![Figure 4.6 Accelerometer and gyroscope positions](image)

Figure 4.6 Accelerometer and gyroscope positions
The biped has two identical virtual sensors - one for measuring the front-back angle, the other for measuring the right-left angle. The block diagram of a virtual sensor is shown in Figure 4.7.

![Figure 4.7 Virtual sensor](image)

The virtual sensors employ the ideas described in the discussion of accelerometer and gyroscope operation. The output of the accelerometer is denoted as $<a>$ in Figure 4.7. An offset value ($offset_a$) is subtracted from $<a>$, the difference is scaled by a gain factor ($gain_a$), and the result is low-pass filtered to acquire the filtered signal, $a_f$. The $a_f$ variable is the argument of the $\sin^{-1}$ function, which yields the estimate of the low-pass filtered angle of the sensor with respect to the ground ($angle_a$). Subtracting $offset_a$ from $<a>$ is necessary because the output of the accelerometer is not always zero when the accelerometer is positioned vertically. The output of the gyro is denoted as $<g>$ in Figure 4.7. An offset value ($offset_g$) is subtracted from $<g>$, and the difference is scaled by a gain factor ($gain_g$). The resulting value is high-pass filtered to acquire the filtered signal, $g_f$. Note that subtracting $offset_g$ is done for numerical reasons - it is intended to prevent overflow in case the gyroscope output is large in steady state. Ideally, the high
pass filter would filter out any DC offset. The filtered signal is integrated to calculate the estimate of the high-pass filtered angle of the sensor with respect to the ground (denoted as \( \text{angle}_g \)). Finally, by summing the estimates of the low-pass and high-pass filtered angle (\( \text{angle}_a \) and \( \text{angle}_g \) respectively) we get the estimate of the lean angle, denoted as \( \text{measured angle} \) in Figure 4.7.

The virtual sensors use matched low- and high-pass filters. The reason for this is the following. The accelerometers give a very good estimate of the biped lean angles in steady state, and the gyros give a very good estimate of the lean angles while the biped is in motion. By using matched filters we can make sure that in steady state, and for low frequency lean angle changes, only the accelerometer readings are used, while for rotational velocities for which the accelerometer readings are significantly tainted by the influence of real acceleration, only the gyro readings are used. Experimentation showed that the system performance is satisfactory when the high pass filter cutoff is set to approximately 0.04 Hz, and the low-pass filter cutoff to approximately 0.03 Hz. The transfer functions of the two matched filters have to be chosen such that the following applies:

\[
H_{LP} + H_{HP} = 1 ,
\]

where \( H_{LP} \) and \( H_{HP} \) are the transfer functions of the low-pass and high-pass filters. Note that this means that the cutoff frequencies of the filters are not independent. The
equations implementing the filters and the integrators of the virtual sensors, and the transfer functions associated with these equations are presented in Appendix 1.

4.1.4 Low Level Control Hardware

The block diagram of the low level control hardware is shown in Figure 4.8.

The low level controller is a digital circuit with two roles. It controls the joint positions, and it provides feedback about the state of the biped to the high level control. If the low level controller malfunctions, the desired joint positions could easily be unreasonable. Trying to move the joints to achieve these angles could result in serious mechanical damage. The protection circuit disables the motor drivers in case of a low level controller malfunction, thus preventing damage to the mechanical hardware. Feedback about the state of the biped comes to the low level controller from the accelerometers, the
gyroscopes, the foot pressure sensors, and from the decoders. There are two accelerometers, two gyroscopes, and eight foot pressure sensors which all produce analog signals, and ten decoders (one for each motor) which provide digital signals.

The decoders are Hewlett Packard HCTL-2016 decoders, made specifically for use with the HEDS-5500 encoders, which are mounted on the motors. These decoders contain a full $4 \times$ quadrature decoder, and a 16-bit up-down counter. Measuring the relative joint angles places a constraint on the minimum number of bits of the counter in the decoder. These relative angles can be measured by decoding the count information contained in the pulses produced by the optical encoders placed on the motor shafts as described in Section 4.1.1. Since the encoders are connected to the motor shaft (not the geared down shaft), the encoders will produce a maximum of $100 \cdot 256 = 25600$ cycles per every full turn of the geared down shaft, where 100 is the resolution of the encoders, and 256 is the maximum gear-down ratio of the motors. The quadrature decoders multiply the resolution of the input signals by a factor of four ($4 \times$ decoding). The geared down shafts are not able to make a full turn without seriously damaging the biped hardware, therefore it is not necessary to be able to count up to $4 \cdot 25600 = 102400$ to know the relative angle change. In fact the shafts can move less than 180° without causing mechanical damage to the biped. This means that the decoders will always produce a number of magnitude less than 50000, that is a number that can be represented by 16 bits. Therefore, a 16-bit counter will be sufficient to track the motions of a shaft. The HCTL-2016 meets this requirement with its 16-bit counter. The HCTL-2016 has an 8-bit bus, which means that
finding the contents of the counter requires two read cycles. A single bit selects the high or the low byte of the counter contents for output on the eight data lines. The counters can be reset using another single logic input, and the eight data lines can be put into high impedance state using yet another single logic input.

At the center of the low level electronics is the Siemens SAB80C166 microcontroller. The SAB80C166 is a 16-bit CMOS single-chip microcontroller for embedded control applications. Among other features it has a ten channel 10-bit A/D converter with 9.75 $\mu$s conversion time, 76 I/O lines with individual bit addressability and 2 serial channels with independent baud rate generators, and parity, framing, and overrun error detection ability. Another very important feature of the SAB80C166 is that it has complete development support including a C-compiler, and an evaluation board equipped with a monitor program. The evaluation board (RMB-166) is made by RIGEL Corporation. This evaluation board along with the monitoring program makes it relatively easy to operate the SAB80C166, because the designer can focus on interfacing the components interacting with the microcontroller and on the control software, instead of designing the electronics needed to operate the SAB80C166, such as memory, bus structure, clock scheme, etc. The protection circuit supervising the microcontroller is an analog circuit with passive components.

The low level electronics uses the UDN2954W full-bridge motor drivers made by Allegro MicroSystems Inc. These drivers are designed for bi-directional control of DC motors,
and can provide continuous output currents of 2 A, and peak start-up currents of up to 3.5 A. The drivers are controlled by two input bits - one turns the current to the motors on or off, while the other determines the direction of rotation of the motors. The full-bridge configuration of the transistors in the driver makes it possible to drive DC motors in both directions using a single power supply. The simplified version of this configuration is shown in Figure 4.9.

![Figure 4.9 Full-bridge configuration](image)

In the full-bridge configuration either all transistors are off, or two diagonally opposite transistors in the bridge are on. If we want to drive the DC motor in one direction we can turn on transistors TR1 and TR4, by setting the ON/OFF, and the DIRECTION control bits. This will make the voltages on the drive inputs of the DC motor $V_{CC}$ and 0 V respectively: $V_{INA} = V_{CC}$ and $V_{INB} = 0 V$. If we now want to reverse the direction of the
motor rotation we have to turn off TR1 and TR4 and turn TR2 and TR3 on, by resetting the DIRECTION control bit. This will make $V_{INA} = 0 \ V$ and $V_{INB} = V_{CC}$, thus reversing the direction of rotation. Notice that, in order to control the speed of rotation of a motor by using the full bridge configuration as shown in Figure 4.9, pulse width modulation (PWM) has to be used. The user has no direct access to the base terminals of the transistors, and the transistors can be either turned completely off, or driven into saturation by the internal logic. This means that the speed of rotation of the motor can only be controlled by turning the transistors on and off for different length periods of time, that is using PWM. Testing showed that $1250 \ Hz$ PWM pulse rates can be used to control the biped motors. Notice that the above configuration minimizes the heat dissipation losses on the transistors. The transistors are either off and there is insignificant current flowing through them, or they are in saturation, and the voltage from collector to emitter is minimal. Either way losses due to heat dissipation in the transistors are the smallest they can be.

The system is powered by $+12 \ V$ voltage. The motor drivers are powered directly by the $+12 \ V$, and the rest of the components are powered from two $+5 \ V$ regulators. Any regulator with approximately $1.5 \ A$ output current can be used, for example the AN7805.

With all the components implementing the blocks of Figure 4.8 described, we can now look at the specific design block diagram, shown in Figure 4.10.
The Siemens SAB80C166 microcontroller plays the central role in the design shown in Figure 4.10. It performs control computations, communicates with the high level logic, and receives feedback input from the foot pressure sensors, the accelerometers, and the gyroscopes. The feedback from the accelerometers and gyroscopes reaches the microcontroller through an analog interface circuit. The interface is necessary because the microcontroller has ten A/D channels, whereas the system requires twelve channels - four for the accelerometers and gyros, and eight for the foot pressure sensors. The interface is controlled by the SAB80C166, and it time-multiplexes the accelerometer and gyroscope readings.
The operation of the microcontroller is monitored by the protection circuit. This circuit is designed to detect software crashes on the RMB-166, and to disable the motor drivers in case of such an event, thus reducing mechanical damage to the biped. In order to achieve this, the protection circuit monitors voltage changes on pin 12 of Port 2 of the SAB80C166. During normal operation the voltage on this pin changes very often, but if the software crashes, the changes stop. The protection circuit can detect that the voltage on the pin is constant, and it then disables the motor drivers.

The SAB80C166 interacts with the motor drivers and the decoders via a digital interface. This interface is necessary because the number of I/O lines required to control the biped is greater than the number of I/O lines available on the SAB80C166 in its RMB-166 evaluation board configuration. In order to see this let us show how many I/O lines are available on the RMB-166, and find how many lines are necessary to control the biped.

The 76 individually programmable I/O lines of the SAB80C166 are organized into four 16-bit I/O ports (Ports 0 through 3), one 2-bit I/O port (Port 4), and one 10-bit input port (Port 5). All bit lines are bit addressable, and all lines of Ports 0 through 4 are individually bit-wise programmable as inputs or outputs via direction registers. Each port line has one programmable alternate input or output function associated with it. Port 0, Port 1 and Port 4 may be used as external memory address and data lines. The pins of Port 2, and some pins of Port 3 can be used as I/O for the capture and compare unit of the microcontroller. The rest of the pins of Port 3 are used with the general purpose timer blocks, and the two serial lines. In addition Port 3 provides the bus interface control
signals WR#, BHE#, READY#, and the system clock CLOCKOUT. Finally Port 5 is used for the analog input channels to the A/D converter.

When an alternate function is used the given port line cannot be utilized for anything else by the user. On the RMB-166 evaluation board the SAB80C166 uses external memory. This means that in our design Port 0 and Port 1, and parts of Port 3 were not available as programmable I/O ports. The design in Figure 4.8 calls for a communication link with the high level logic. There are two serial lines available on the SAB80C166. This suggests that only half of the pins which provide alternate functions for the serial communication lines were not available for our design. However none of these pins were used for general I/O functions, in order to facilitate possible future expansions of the design, which would require an additional serial line. This leaves us with sixteen programmable I/O pins of Port 2, nine programmable I/O pins of Port 3, and ten input pins of Port 5. That is we have twenty-five programmable I/O pins and ten input pins to the A/D converter at our disposal.

In order to control the biped we need, in terms of digital input, an 8-bit wide bus to read the outputs of the decoders, ten bits to enable and disable the decoders connected to the bus, one bit to reset the counters, one bit to select high byte or low byte output on the decoders, and $2 \cdot 10 = 20$ bits to control the motor drivers. This adds up to forty digital I/O lines, which is more than we have at our disposal on the SAB80C166 (twenty-five programmable digital I/O pins). Obviously an interface is necessary to increase the number of digital outputs. The digital interface is implemented using two levels of
transparent latches. The input to the interface consists of three control signals, and ten data signals, a total of thirteen SAB80C166 outputs. The output consists of twenty-eight data signals. Therefore, the number of digital outputs available after the interface is \((25 - 13) + 28 = 40\), which is enough to control the biped using the described scheme.

Approximately 80% of the low level hardware was implemented by manufacturing a printed circuit board (PCB). The board was named the R1.1. Figure 4.11 shows the hardware block diagram of the low level controller with the parts implemented in PCB form placed within a shaded box.

![Figure 4.11](image)

**Figure 4.11** Low level control hardware - parts implemented in PCB form are in a shaded box
The circuit diagram of the low level control hardware and tables describing the connections between the elements of the low level electronics can be found in [41].

4.1.4.1 Heat Dissipation Issues

The motor driver chips and the +5 \( V \) regulators dissipate a lot of heat energy during operation, and they have to be cooled in order to prevent overheating. Three two-fin heat sinks were installed on the R1.1, and the ten motor drivers and two regulators were mounted on them. The size of each fin is 75 \( \times \) 30 mm\(^2\), and the fins are 8 mm apart. Each heat sink cools four components - two of the sinks have four drivers mounted on them each, while the third heat sink cools two motor drivers and the two regulators.

In addition to the heat sinks two 12 V miniature fans helps cool the 5 V regulators and the motor driver chips. The fans are mounted inside the metal box housing the R1.1 board.

4.1.5 Battery Power

When the biped operates off an external power supply, the ripple due to the voltage drop in the power cord is considerable - around 5-6 \( Vpp \). Such large drops in voltage can cause the 5 V regulator outputs to drop below 5 V which in turn can cause the low level control program executing on the SAB80C166 microcontroller to crash. However, if a battery is hooked up in parallel with the external supply it acts as an active capacitor smoothing out the provided 12 V voltage level. In the case of the battery used for the UNH biped, the ripple is reduced to an acceptable level of approximately 1 \( Vpp \).
In order to be able to choose the most appropriate battery for the robot, the system requirements were summarized. The system requires a maintenance free, 12 V supply. The tradeoff between battery capacity and battery size and weight has to be taken into account. In other words, the power density of the battery has to be fairly high. The size is limited by the space available in the biped sheet metal box to approximately $160 \times 75 \times 110 \text{ mm}^3$. The weight is limited by the torque the motors can produce - the battery cannot be too heavy for the biped motors to carry.

Another important consideration is the most suitable method of charging of the selected battery. If the charging method is the constant-voltage method there is no need for taking the battery off line to charge it - it is enough to connect it in parallel with an external power supply.

The above requirements point to using a sealed lead acid battery. These batteries are maintenance free, have a relatively high power density, and they can be charged using the constant voltage method. The battery used in the UNH biped is a Yuasa NP4-12. This is a 12 V, 4 Ah battery. Its size is $90 \times 70 \times 106 \text{ mm}^3$, and it has a mass of 1.7 kg.

4.1.6 Tethered Operation

In order to connect the biped to the external power supply and to the PC, a tether is used. The tether is approximately 8 m long, and has a mass of approximately 0.7 kg. It has five wires: two wires for power and three wires for serial communication. The tether is
secured to a 30 cm long metal support which is attached to the biped body, as shown in Figure 4.12. This metal support has three roles. It acts as strain relief for the tether, it keeps the tether away from the feet of the biped, and it protects the biped motors in case the robot falls during operation.

![Figure 4.12 UNH biped with tether](image)

4.2 **High Level Hardware**

The high level hardware consists of a 200 MHz PentiumPro-based personal computer and a SAB80C166 microcontroller board. The personal computer is used to calculate the desired actuator angle sequences in order to implement walking. The microcontroller board is used to facilitate fast serial communication between the high- and low level electronics.
4.2.1 Serial Communication Link

The personal computer's serial line can handle only about $50 \text{ kBaud}$ transmission rates, while the microcontroller boards can communicate at up to $625 \text{ kBaud}$. The microcontroller board connected to the personal computer allows us to use a serial communication link at this high baud rate. The personal computer interfaces with this microcontroller board through its parallel port. The parallel port has a bandwidth wider than necessary for serial communication at $625 \text{ kBaud}$. The microcontroller board takes the data stream received from the personal computer, and serially transmits it to the low level electronics at the high rate, as shown in Figure 4.13. The connection works similarly in the opposite direction. The personal computer's microcontroller board is in fact transparent to the high and low level controllers.

![Diagram of serial connection between high- and low level hardware](image)

**Figure 4.13** Serial connection between high- and low level hardware
5. ADAPTIVE DYNAMIC BALANCE

This chapter introduces the latest version of the dynamic gait controller for the UNH biped. The controller was implemented as part of research conducted before the research presented in this dissertation. The experiences gained from working on the controller presented here were used in building the unified gait controller, which is the topic of this dissertation. Before we proceed with examining the dynamic gait controller, the notion of posture space will be introduced in Section 5.1.

5.1 Posture Space

In order for the biped to walk successfully one has to control the joint angles of the biped legs. That is, the controller has to work in the biped's joint space. Yet, when we visualize the biped walking, it is more intuitive to think in terms of the height of the biped (or the biped hips), the angles of the body in the frontal and the lateral planes, and the position of the feet with respect to the upper body and the floor. For this reason the biped control architecture was created in the posture space. Let us first define what we mean by posture. The biped's posture is defined by the posture parameters. In the design presented here there are seven posture parameters and they are summarized in Table 5.1:
<table>
<thead>
<tr>
<th>Posture Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. right-left lean</td>
<td>the angle of the biped upper body with respect to the normal to the floor in the frontal plane</td>
</tr>
<tr>
<td>2. front-back lean</td>
<td>the angle of the biped upper body with respect to the normal to the floor in the sagittal plane</td>
</tr>
<tr>
<td>3. height</td>
<td>the height of the biped hips</td>
</tr>
<tr>
<td>4. right foot forward</td>
<td>the shortest distance between the normal projection of the right foot on the floor and the normal projection of the biped’s hips on the floor</td>
</tr>
<tr>
<td>5. left foot forward</td>
<td>the shortest distance between the normal projection of the left foot on the floor and the normal projection of the biped’s hips on the floor</td>
</tr>
<tr>
<td>6. right foot lift</td>
<td>the distance of the right foot from the floor</td>
</tr>
<tr>
<td>7. left foot lift</td>
<td>the distance of the left foot from the floor</td>
</tr>
</tbody>
</table>

**Table 5.1** *List of posture parameters*

The posture parameters are graphically shown in Figure 5.1. The body angles can be expressed in degrees. The value of either lean can be positive or negative. The height parameter can be expressed in centimeters, and it is always positive. The position of the feet with respect to the upper body can be given in centimeters, and it can be positive or negative. The vertical distance of the feet from the floor can also be given in centimeters, and it is always positive.
Note that, in order to implement a set of commanded posture parameters, they need to be translated to commanded actuator angles. Since this translation is never perfect, a set of commanded posture parameters will always result in a set of implemented posture parameters that are different from the commanded ones. This can create a problem if the implemented posture parameters are measured using the biped sensors, and we want to use the difference between the measurements and the commanded values in the controller.

5.2 Hypotheses

The research into dynamic balance walking at the UNH Robotics Lab rested on three hypotheses.
The first hypothesis proposed that dynamic walking can be implemented by depending on pre-planned, but adaptive, smooth posture sequences, rather than on reactive closed-loop control. This hypothesis suggested that the dynamic gait posture sequences can be pre-planned using heuristics and a simple model of the biped, and that the pre-planned posture sequences can then be modified using some form of adaptation in order to achieve stable walking. The hypothesis therefore proposed that the biped should "learn" how to walk without falling by learning from previous experiences, rather than try to reject disturbances when they occur.

Figure 5.2 shows the basic walking gait of the biped robot. As a result of the distribution of mass within the structure, the biped cannot simply lift a foot without falling. In order to move a foot, it is necessary to first generate a lateral momentum toward the opposite side. The foot can then be lifted and moved to a new location. The resulting gravitational force when the foot is lifted breaks the momentum and allows the biped to fall back onto the lifted foot. In other words the biped sways in the frontal plane while walking. The second hypothesis dealt with the relationship between the control algorithm and the natural dynamics of the biped. It proposed that a phase-locked central pattern generator be used to conform to and make use of the natural dynamics. The idea was that the central pattern generator, which outputs the pre-planned posture sequences, be synchronized with the swaying motion of the biped through sensory triggers, where the sensory triggers could be certain body angles, or a foot making or breaking contact with the ground. This way the control algorithm would not fight the natural dynamics, but instead make use of it.
Finally, the third hypothesis said that it should be possible to use simplified frontal and lateral plane kinematics to translate logical "posture commands" to joint commands.

5.3 Control Architecture

Figure 5.3 gives the outline of the control architecture. Variables in angle brackets are sampled physical measurements or functions of these measurements. Variables depicted in capital letters are parameters set by the user. Variables in lower case letters represent results of control calculations.

The side-to-side and foot movement motions in the walking process are initiated by a gait oscillator based on simple heuristics and an approximate model of the biped kinematics. CMAC neural networks [2, 3, 4] are used to modulate the gait generator output, as a function of desired step parameters (step length and step rate) and immediate sensor feedback. The CMAC neural networks are responsible for the control of side-to-side and front-to-back balance, as well as for maintaining good foot contact. The control system
creates smooth motion sequences by superimposing pre-planned and adaptive elements: the gait oscillator provides the pre-planned elements of the sequences, while the outputs of the CMAC neural networks add the adaptive elements.

![Diagram of Biped Control Architecture for Dynamic Walking](image)

Figure 5.3 *Biped control architecture for dynamic walking*

The gait oscillator divides the biped steps into three phases: *stepping leg extension phase*, *stepping leg lift phase*, and *stepping leg relaxation phase*. The lateral momentum necessary to lift a foot is generated in the first phase by extending the stepping leg, and thus tilting the biped. In the next phase the biped takes advantage of the lateral momentum, lifts the stepping foot and moves it forward. In the third phase the stepping leg relaxes and the biped is brought back into its vertical position. After this phase the biped legs exchange roles, and the three phases are repeated for the opposite leg. Therefore, the complete gait cycle consists of the following six step phases: *left leg extension phase*, *left leg lift phase*, *left leg relaxation phase*, *right leg extension phase*, *right leg lift phase*, and *right leg relaxation phase*.
The gait oscillator is based on responses to sensory triggers, rather than on reactive closed-loop control. It utilizes the concept of phase-locked central pattern generation to conform to, and make use of, the natural dynamics. In the case of the dynamic biped walking described here, the sensory triggers are the instances of each foot contacting or breaking contact with the ground, as detected by the foot force sensors. The closed loop system forms a phase-locked-loop which synchronizes the gait generator and the biped dynamics. The phase error is derived from the sensory triggers, and the period of the natural dynamics is regulated by modifying the magnitude and speed of the commanded side-to-side lean.

In Figure 5.3 CMAC 1 is used to control the instantaneous front-back position of the hips relative to the feet. It outputs an offset value which is used in the calculation of the desired distance between the vertical projection of the hips on the floor, and the feet. Let us define a variable called the front-back zero moment point (ZMP\textsubscript{FB}). Let \( FR_1, FR_2, FL_1, \) and \( FL_2 \), be the readings from the pressure sensors in the front of the two feet (the “toes”), and \( FR_3, FR_4, FL_3, \) and \( FL_4 \) the readings from the pressure sensors in the back of the two feet (the “heels”). The ZMP\textsubscript{FB} is defined as:

\[
ZMP_{FB} = \frac{\sum_{i=1}^{2} FR_i + \sum_{i=3}^{4} FL_i - \sum_{i=1}^{4} FR_i - \sum_{i=3}^{4} FL_i}{\sum_{i=1}^{4} FR_i + \sum_{i=1}^{4} FL_i} \cdot 40
\]
The ZMP<sub>FB</sub> is in the [-40, 40] range. It reaches its maximum when all the force on the biped feet is on the "toes" of the feet, and it reaches its minimum when all the force is on the "heels". CMAC 1 provides adjustments to the relative body position in order to achieve a value of ZMP<sub>FB</sub> close to zero. In other words the output of CMAC 1 is used to achieve an equal overall distribution of force between the "toes" and the "heels" of the feet during the stepping motions. This has the effect of preventing the biped from falling forward or backward. The neural net is trained using the ZMP<sub>FB</sub> as the training error signal. A state of imbalance at a given time during walking generally results from incorrect postures at earlier times, rather than from an incorrect current posture. Thus, the general technique of training using a temporal eligibility trace [38] is used to distribute the information from the delayed supervised learning over sequential time steps.

The inputs to CMAC 1 are the gait phase variable, the right-left and front-back accelerations and their derivatives, the ZMP<sub>FB</sub>, and different step parameters. The gait phase variable indicates the step stage. The value of this variable is zero at the beginning of the left leg extension phase, and it monotonically increases to 36000 at the end of the right leg relaxation phase. The accelerations and step parameters will influence the optimal instantaneous position of the hips relative to the feet, therefore they have to be included in the CMAC 1 inputs. The acceleration derivatives are used because the output of the physical accelerometers depends on body accelerations as well as on body position. The accelerometers can have the same output for different combinations of body accelerations and body positions. The derivatives provide information that can be used to
reduce the ambiguity caused by this property of the accelerometers. Note that the
dynamic gait controller does not use the rate gyroscopes, because they were not part of
the hardware system at the time this dynamic walking algorithm was implemented on the
UNH biped.

CMAC 2 is used to predict the correct amplitude and speed of side-to-side leaning during
each step. It outputs an offset value which is used to modulate the lean amplitude and
speed of the biped, as calculated by the gait generator. An insufficient lean causes the
foot to lift for too short a duration (or not at all), while too much lean causes the foot to
lift for too long a duration (or for the robot to fall over laterally). The proper amplitude
of lean is dependent on the state of the robot at the beginning of the step, and varies
somewhat from step to step. It varies significantly for different desired step lengths and
rates. CMAC 2 is trained after each step, based on the difference between the desired and
observed foot lift durations for that step.

The inputs to CMAC 2 are the gait phase variable, the right-left and front-back
accelerations and their derivatives, and different step parameters. Their use is analogous
to the use of these parameters in CMAC 1.

CMAC 3 is used to learn kinematically consistent robot postures. Whenever it is desired
that both feet be in solid contact with the floor (double-support phases), the closed-chain
kinematics of the structure have to be addressed. Target positions for all ten motors
cannot be specified independently. In the dynamic gait controller, hip and knee angles are produced directly by the gait controller. CMAC 3 then predicts ankle position corrections in order to keep the biped feet parallel to the floor with the force balanced in the middle of each foot.

The inputs to CMAC 3 are the gait phase variable, and the different step parameters. They are used in a similar fashion as in the other two CMACs.

5.4 Neural Network Training and Qualitative Results

Training of the biped typically proceeds as follows. The three CMAC neural networks are first trained during repetitive foot lift motions similar to marching in place (i.e. no attempt is made to translate the lifted foot). This is typically carried out for five minutes, with different settings for desired foot lift height (in the range 0.5 to 2.5 cm). Frequent human support is required to keep the biped from falling during the first half of this training, and occasional support is required during the second half. Then, training of the three CMAC neural networks is carried out during attempts at walking (translating the lifted foot forward), for increasing step lengths, and/or for various step rates. Again, frequent human support is required during early training for each new parameter setting, while less frequent support is required after 2 or 3 minutes of training at a given setting. After about 60 minutes of total training time, the biped is able to shift body weight from side-to-side while maintaining good foot contact, and to lift a foot off of the floor for a desired length of time, during which the foot can be moved to a new location relative to
the body. Using these skills, the biped is able to start and stop on demand, and to walk with continuous motion on flat surfaces at a rate of up to 100 steps per minute, with step lengths of up to 6 cm per step (corresponding to 12 cm stride lengths).

More detailed results about the dynamic walking performance of the UNH biped can be found in reports by Miller and Kun [8, 9].

5.5 Discussion

In research at the UNH Robotics Lab it was shown that the hypotheses listed in Section 5.2 could be used in creating a controller for dynamic bipedal walking. However the biped walking with the dynamic gait required human supervision, as failures occurred every few minutes, and the biped would fall without support. The lowest attainable walking speed was determined by the system dynamics: the biped could not walk at speeds that required sideways swinging at frequencies much below the natural frequency (recall that a detailed explanation of this result was given in Section 2.4). The top speed was limited by the highest possible swinging frequency, which in turn was limited by the available motor torques and the bandwidth of the serial communication link between the high and low level controllers. Step length was limited by the masses of the motors at the knee and ankle joints. Reaction forces resulting from the acceleration and deceleration of these masses during steps increased the coupling between the frontal and sagittal plane balancing problems, causing the controller to fail.
6. ADAPTIVE STATIC BALANCE

In this chapter we will look at the adaptive control mechanism used to perform static walking with the UNH biped, and the walking results obtained using this control architecture. Before we do that however, we need to introduce the biped gait that was implemented.

6.1 Static Walking Gait

As a result of the distribution of mass within the biped structure, in order to move a foot using a static balance gait, it is necessary to first counterbalance that foot by leaning the biped’s upper body toward the opposite side. The foot can then be lifted and moved to a new location. The control strategy has to deal with the resulting gravitational force when the foot is lifted. Note that, even at slow walking speeds, the dynamics of the system cannot be completely neglected and the walking is quasi-static, rather than static. However, for the sake of simplicity, in the following we will refer to quasi-static walking as static walking. The biped gait is shown in Figure 6.1. The biped goes through double support phases, when both feet are on the ground, and single support phases when only one foot is on the ground. The arrows show the direction in which the upper body is moving. The circles imply that the upper body is ideally not moving in the frontal plane.
Figure 6.1 *Static balance gait. The biped starts with bent knees, and leans to one side by extending the opposite leg.*

An important point is that leaning to one side is achieved by starting the biped with bent knees and extending the opposite leg. This method of leaning is illustrated by bent legs in Figure 6.1. Alternative ways of leaning the upper body are leaning it from the hips or from the ankles, but the biped’s DC motors are not strong enough for these.

The biped’s static walking gait is logically divided into the following six phases:

1. *extend left leg phase:* In this phase the biped leans from left to right by extending the left leg, in order to take weight off the left foot.

2. *lift left foot phase:* At the end of the *extend left leg phase* the left foot ideally has very little weight on it, and can be lifted off the ground.

3. *lower left foot phase:* Once the left foot reaches the highest point in its trajectory this phase starts, and the foot is lowered back onto the ground.

4. *extend right leg phase:* Symmetrical counterpart of the *extend left leg phase.*
5. *lift right foot phase*: Symmetrical counterpart of the *lift left foot phase*.

6. *lower right foot phase*: Symmetrical counterpart of the *lower left foot phase*.

### 6.2 Adaptive Control of Static Balance Walking

Now that we know what the biped gate has to look like, let us examine the control architecture of the biped for static walking. The control architecture consists of a high- and a low level controller. Figure 6.2 shows in block diagram form how the control architecture fits into the overall biped system.

![Overall biped system block diagram](image)

**Figure 6.2 Overall biped system block diagram**

The high level controller generates pre-planned, but adaptive, sensory triggered, smooth sequences of desired postures, called "commanded postures" in Figure 6.2. This involves posture sequence generation based on a simplified model of the biped, and the
modification of this initial posture sequence using neural networks and a PID controller. The low level controller performs three steps. First it transforms the posture sequences, received from the high level control, into desired actuator angle sequences. The actuator angle sequences are then modified as a result of reactive control of the right-left and front-back angles, and the active control of the foot contact in double support phase. This will be explained in Section 6.4. Finally, the corrected actuator angle sequences are implemented using PID control. Therefore, the overall control strategy (including both the high and the low level control) uses a combination of pre-planned, but adaptive, smooth motion sequences with sensory triggers, and reactive closed-loop control.

6.3 High Level Control

The block diagram of a very simplified model of the high level controller is given in Figure 6.3. Based on a crude model of the biped, we can design a sequence of postures that the biped has to go through in order to take steps. We can call these postures the "simple postures". However, a sequence of simple postures will not necessarily lead to stable walking, due to the fact that the sequence is generated based on an insufficiently accurate model of the system. In order to achieve stable walking we modify the simple sequences with the outputs of several CMAC neural networks [2, 3, 4], and a PID controller. The "simple postures" are changed into "commanded postures", such that the gait is more stable.
6.3.1 High Level Software Implementation

The high level control of the UNH biped is implemented as a digital controller on a 200 MHz PentiumPro-based personal computer, under the Microsoft NT operating system.

The structure of the high level control software is shown in Figure 6.4. The program has four threads. These threads perform keyboard input handling, data display, communication with the low level software, and control functions. The threads have access to a number of global variables. Information is passed between the threads by modifying these global variables.
The keyboard input thread runs continuously, and handles user commands that are coming from the keyboard. This thread runs at the normal priority level, and competes for processor time equally with other threads on the system. Since this thread only requires processor time when a key is hit, it will not use up too much processor time, and other threads will get a chance to execute too.

The communication thread maintains a communication link with the low level control software. This thread is responsible for sending desired postures to the low level control, and for receiving feedback about the status of the robot. The execution of the communication thread is triggered by a PC clock every 28 ms (this translates into a 35.71 Hz rate). The control thread performs the control calculations necessary to implement the function chosen by the user. The execution of this thread is triggered by the
communication thread when new information is available about the state of the biped, therefore the control rate of the system is 35.71 Hz.

The communication thread runs at the highest priority level of the four threads in order to achieve a constant control rate - if this thread could be interrupted by another thread the control rate would depend on the variable timing of other threads executing under Windows NT. The control thread runs at a priority level that is between the priority levels of the input thread and the communication thread.

The data display thread also runs continuously, and it provides information about the state of the biped on the PC screen. The data display thread runs at idle priority level, that is it executes when no other threads require processor time. Since the information about the biped state changes with every execution of the feedback thread, that is every 28 ms, the display thread constantly requires processor time, and if it had higher priority (normal for example) it would prevent other important threads from executing.

6.3.2 High Level Control Architecture

Figure 6.5 gives the outline of the high level controller. Variables in angle brackets are sampled physical measurements. Variables depicted in capital letters are parameters set by the user. Variables in lower case letters represent results of control calculations.
The high level controller has seven major components represented in the block diagram of Figure 6.5. The walking gait is initiated by the gait generator. The gait generator outputs seven posture parameters, which form posture1. These parameters are modified by CMAC neural nets and a PID controller to produce the output of the high level controller. This output is a posture command, consisting of eight posture parameters,
denoted as the *commanded posture* in Figure 6.5. Recall from Section 5.1 that there are seven posture parameters that define the posture of the biped. An eighth parameter, the *right-left target lean (rlt)*, was introduced that holds the value of the target (desired) right-left lean angle of the biped. Let us see why it is necessary to introduce this parameter. Both the high and the low level controllers need the value of the target right-left lean angle because they implement PID controllers to regulate the right-left lean angle. The controllers use the difference between the target and the measured right-left lean as the error signal. The value of the *right-left lean* parameter (*rll*) will never match the measured value of the right-left lean angle due to inaccuracies in the process of translating posture commands to angle commands. Therefore, the *rll* cannot be used to calculate the error signal for the PID controllers, because the error could never become zero. This is illustrated in the top block diagram of Figure 6.6. To solve this problem we introduce the *right-left target lean* parameter (*rlt*). The *rlt* is equal to the value of the *right-left lean* parameter (*rll*) modified by the output of CMAC 2. Its value is the true target value for the right-left lean angle. The *rll* is obtained by modifying the *rlt* such that, when the *commanded posture*, including the *rll*, is translated and implemented by the low level control, the actual (measured) right-left lean angle matches the *right-left target lean*. Now we can use the difference between the *right-left target lean* and the measured right-left lean angle in the PID controllers, because their difference can become zero. This idea is outlined in the bottom block diagram of Figure 6.6.
Figure 6.6 Right-left target lean parameter

Figure 6.7 shows values of \( rll \), \( rlt \), and the measured right-left angle logged during a biped walking experiment. The figure shows that the measured (implemented) right-left angle tracks the value of the right-left target lean parameter fairly well, while the \( rll \) value that is used in the commanded posture is significantly different from the measured lean.
Figure 6.7 The measured right-left angle tracks the rlt, not the rll.

The low level controller also implements a PID controller for the front-back lean angle. However, we do not need to pass the value of the desired front-back lean angle to the low level controller, because this value is constant - the desired front-back lean angle is always $9^\circ$ (see Section 6.3.3). This value is hard-coded in the low level controller.

Let us now look at each of the blocks of the high level controller individually.

6.3.3 The Gait Generator

The side-to-side and foot movement motions in the walking process are initiated by the gait generator, which is based on simple heuristics and an approximate model of the biped kinematics. The biped has to lean to one side in order to lift up the opposite foot. The gait generator has preprogrammed values for the necessary leans for lifting the feet.
CMAC 1 outputs corrections to these preprogrammed right-left lean angles. An insufficient lean causes the foot to lift for too short a duration (or not at all), while too much lean causes the foot to lift for too long a duration (or for the robot to fall over laterally). The proper lean is dependent on the state of the robot at the beginning of the step, and varies somewhat from step to step. It varies significantly for different desired step lengths and rates. In the static walking implementation described here it is assumed that the variations in the state of the robot at the beginning of steps taken by the same foot are negligible. In other words, it is assumed that the robot is always in the same state when it starts a step for a given foot. Therefore, CMAC 1 takes into account which foot is to be moved (that is which side the robot is leaning toward), and what the desired step length is. The desired stepping rate is kept constant, and it is not explicitly taken into account. Using these variables as state inputs, CMAC 1 is trained after each biped step, based on the foot pressure sensor readings. When a foot starts lifting up one of two things can happen. The first one is that, when the foot is commanded to be 1.5 cm off the ground, the foot is still on the ground, as detected by the pressure sensors on that foot. In this case the gait generator decides that the sideways lean was not sufficient to perform a step, the step is abandoned, and CMAC 1 is trained as though the biped fell onto the lifted foot. Let us first examine the other option, which is that the foot is successfully lifted off the ground by the point it is commanded to be off the ground by 1.5 cm. In this case the foot is lifted to the commanded height of \( \text{lift}_{\text{preparation}} + \text{lift magnitude} \), where \( \text{lift}_{\text{preparation}} \) is the value of the commanded lift at the moment the foot brakes contact with the ground, as detected by the pressure sensors, and \( \text{lift magnitude} \) is the desired actual height of the foot when it is fully lifted. Once the commanded foot lift reaches the
value of $\text{lift}_{\text{preparation}} + \text{lift magnitude}$, it is commanded to descend, until the commanded foot lift is zero. The gait generator registers the value of the foot lift command at the moment when the foot makes contact with the ground again, and this value is called $\text{lift}_{\text{land}}$. CMAC 1 is trained to help keep the feet in the air such that:

$$\text{lift}_{\text{preparation}} + \text{lift}_{\text{land}} = 1.6 \text{ cm}.$$ 

If the sum of the two values is less than $1.6 \text{ cm}$, CMAC 1 will be trained to lean the biped further away from the foot that will be lifted, and vice versa. We do not want the value of $\text{lift}_{\text{preparation}}$ to be zero, because this would mean that the biped is close to leaning out too much. We do not want the value of $\text{lift}_{\text{land}}$ to be zero for the same reason. Therefore we want the sum of the two values to be greater than zero.

Using the foot pressure sensors we can also detect if the biped fell over sideways. If the biped falls over sideways it has to rotate around the outer edge of one of the feet, in which case only the two pressure sensors on the edge of that foot will have non-zero readings. Similarly, it is easy to detect if the biped did not manage to lift up a foot, or it fell back on the lifted foot during the step. In this case some, or all, of the pressure sensors on the foot that should be lifted will have non-zero readings. For these two extreme cases CMAC 1 is trained to change the lean angle more dramatically. In the case of falling over sideways the lean angle is trained to be smaller by a preset value, and in the case the foot did not lift off, or fell back on the ground, the lean angle is trained to increase by a different preset value.
The output of the gait generator is depicted as posture\textsubscript{r} in Figure 6.5. In the software implementation of the high level control this is the closest we can get to the idealized "simple posture" shown in Figure 6.3. The pure "simple posture" would use the preprogrammed values for the right-left lean angles for lifting the feet.

Figure 6.8 shows three posture\textsubscript{r} parameter sequences. The three sequences are the desired right and left foot lift, and the desired right-left body lean, that is right foot lift, left foot lift, and right-left lean. The biped is first commanded to lean to the left (negative angle), and to lift its right foot approximately 6 cm. Note that the value the foot is lifted to is lift\textsubscript{preparation} + lift magnitude. Then the biped leans to the right (positive angle) and lifts its left foot about 6 cm, etc.

![Figure 6.8 Three posture, parameter sequences](Image)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Another two posture parameter sequences, the right foot forward and the left foot forward, are shown in Figure 6.9, along with the right foot lift and left foot lift sequences. The right foot forward and the left foot forward parameter sequences describe the desired position of the feet with respect to the hips. When a foot is in the air, that foot is commanded to go forward relative to the hips (by 12 cm in this case). After the foot lands the biped gets ready to lift the other foot by moving the previously lifted foot exactly under the body of the biped. This way it can provide the most stable support while the other foot is in the air. The supporting foot is kept steady under the body for the duration of the step. When the lifted foot lands both feet start moving backward relative to the biped hips. The foot that just landed is moving backward in order to get into position under the biped body to become the supporting foot for the next step. The foot that used to be the supporting foot has to keep up with this motion in order to keep the distance between the feet constant and thus prevent slipping. However, the two feet are not moving backward an equal amount, as can be seen in Figure 6.9 from the fact that the right foot forward and left foot forward curves are not parallel. The reason is that, without pushing the foot that will be lifted backward more than the other foot, this foot will drift forward as the biped leans away from it. This effect will be explained in detail in Section 6.3.7.

Recall that there are seven posture parameters. So far we have introduced five. The remaining two are the height and the front-back lean. These are kept constant throughout the walking - the desired hip height is always 41 cm, and the desired front-back lean is
always 9°. Keeping the hip height and the front-back lean constant simplifies the control algorithm and it produces good walking results.

**Figure 6.9** *Four posture, parameters*

The gait generator relies on sensory triggers to create posture sequences. Sensory triggers are the instances of each foot contacting or breaking contact with the ground, as detected via the foot force sensors, and the biped body reaching angles sufficient for successful foot lift, as detected by the virtual body angle sensors. The virtual body angle sensors use information from the accelerometers and the rate gyroscopes to determine the body lean angle in the frontal and the sagittal plane, and they are implemented in the low level controller. Their operation was explained in Section 4.1.3. Figure 6.10 shows how sensory triggers are used to commence lifting a foot: the biped leans to the left (negative angle) until a certain angle is reached: at that point it lifts up the right foot and keeps the sideways angle constant. The angle starts changing again after the right foot is not being
lifted any more, and it is on the floor - that is another sensory trigger is used, this time a reading from the pressure sensors on the sole of the right foot (the pressure sensor readings are not shown in Figure 6.10).

![Graph showing sensory triggered lifting of the right foot](image)

**Figure 6.10** *Sensory triggered lifting of the right foot*

### 6.3.4 Balance Control Using CMAC Neural Networks

Statically stable biped walking is achieved by keeping the robot’s normal projection of center of mass (*NPCM*) within the stability region, as defined in Section 2.2. We can position the *NPCM* in the middle of the foot in the single support phase and thus achieve equal error margins for falls to the right and to the left, and equal error margins for falls forward and backward. However, the biped cannot recover from falling sideways away from the lifted foot, while falling onto the lifted foot gives the robot a chance to recover. Therefore, in the single support phase, it makes sense to try to position the *NPCM*
slightly off center toward the lifted foot. CMAC 2 is used to predict the instantaneous side-to-side lean during walking, such that in single support phase the right-left component of the \( NPCM \) is positioned in the way described above. In double support phase the CMAC 2 output is trying to drive the right-left component of the \( NPCM \) to the center of the foot that will be the supporting foot in the next single support phase. This is an important goal because keeping the \( NPCM \) in the center of the foot that will become the supporting foot prevents rocking at the moment of lifting the opposite foot. Again, the proper lean for achieving the above is dependent on the state of the robot at any instant, and varies significantly for different desired step lengths and rates. CMAC 2 is trained during every execution of the control thread of the high level controller. The training is based on the difference between the desired and observed right-left component of the \( NPCM \) position. If, for a given state, the \( NPCM \) is too far to the right CMAC 2 is trained to help the biped lean more to the left, and vice versa. The state inputs of CMAC 2 are the foot lift magnitude, the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles.

CMAC 3 is used to control the instantaneous front-back position of the hips relative to the feet. In Section 5.3 we defined the \textit{front-back zero moment point} (ZMP\(_{FB}\)). If \( FR_1 \), \( FR_2 \), \( FL_1 \), and \( FL_2 \), are the readings from the pressure sensors in the front of the two feet (the "toes"), and \( FR_3 \), \( FR_4 \), \( FL_3 \), and \( FL_4 \) are the readings from the pressure sensors in the back of the two feet (the "heels"), he ZMP\(_{FB}\) is calculated as:

92

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The \( ZMP_{FB} \) is in the \([-40, 40]\) range. It reaches its maximum when all the force on the biped feet is on the "toes" of the feet, and it reaches its minimum when all the force is on the "heels". CMAC 3 provides adjustments to the position of the hips relative to the feet in order to achieve a value of \( ZMP_{FB} \) close to zero. In other words the output of CMAC 3 is used to achieve an equal overall distribution of force between the toes and the heels of the feet during the stepping motions. This has the effect of preventing the biped from falling forward or backward. The neural net is trained during every execution of the control thread of the high level controller, using the \( ZMP_{FB} \) as the training error signal.

For negative values of the \( ZMP_{FB} \) the CMAC is trained to help the biped position its hips further forward, and vice versa. The state vector of CMAC 3 contains the desired step length and foot lift magnitude, the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles.

6.3.5 Posture Command Correction Using CMAC Neural Networks and PID Control

The control architecture uses simplified kinematics to translate the posture commands into desired actuator angles - this will be explained in more detail in Section 6.4.3. Due to this fact, and other inaccuracies in the model, commanded postures will not be
implemented perfectly. This is a problem because, as will be explained in Section 6.4.5, the low level controller uses the difference between the target and measured lean angles as error signals for two PID controllers. CMAC 4 and CMAC 5 are used to learn adjustments to the desired right-left lean and front-back lean posture commands respectively, such that once these commands are implemented by the controller, the achieved angles match the desired angles better. A PID controller helps with achieving the correct right-left lean angle.

CMAC 4 and CMAC 5 are trained during every execution of the control thread of the high level controller. The training is based on the difference between the desired and observed lean angles. The state vector of CMAC 4 consists of two continuous functions of the phase variable, the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles. The state vector of CMAC 5 consists of the desired front-back lean angle, the desired foot lift magnitude, the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles.

The high level controller also implements a PID controller to adjust the value of the right-left lean parameter such that the measured right-left lean matches the target right-left lean parameter. This PID controller helps the system especially in the early stages of learning.
when CMAC 4 is not able to provide the correction needed to implement the desired right-left lean.

6.3.6 Overview of CMAC Neural Networks Used in the High Level Controller

The elements of the input vectors of the five CMACs are summarized in Table 6.1. Notice that CMACs 2, 3, 4, and 5 all take into account the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles. This is because these values contain the information about the instantaneous position of the biped and about its dynamics. Another variable that is used by multiple CMACs (CMACs 2, 3, and 5) is the desired foot lift magnitude. This variable influences the right-left stability of the biped, since changing the height of the lifted foot changes the moment around the NPCM in the frontal plane. Consequently, it is part of the state space of CMACs 2 and 5. For short steps the influence of this variable on the moment around the NPCM in the sagittal plane is small. However, for longer steps the moment in the sagittal plane also becomes larger. This is why this variable is also part of the state space of CMAC 3. Another important role of the desired foot lift magnitude variable is that it is used for step timing. The lifting and the lowering of the feet is done at a predetermined speed. Therefore, the length of time the lifting and lowering takes depends on the desired foot lift magnitude. This role of the variable further justifies its inclusion in the above CMACs.
CMAC uses two continuous functions of the *gait phase* variable as state inputs. As we mentioned in Section 6.1 the gait is divided into six phases. The *gait phase* variable is generated by the gait generator, and it keeps track of the progress of the stepping motion through these six phases. The gait phase is incremented during each execution of the control thread. Its value goes from zero to 36000 - it starts out with zero at the beginning of the *extend left leg* phase, and ends up with 36000 at the end of the *lower right foot* phase. Once it reaches 36000 it is reset to zero, and it is incremented again. This means that there is a discontinuity in the value of the *gait phase* variable at the point where the above two phases meet. If this discontinuity was propagated to the state input of the

---

Table 6.1 *CMAC input vector elements*

<table>
<thead>
<tr>
<th>State Vector Element</th>
<th>CMAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>desired step length</td>
<td>✓</td>
</tr>
<tr>
<td>desired foot lift magnitude</td>
<td>✓</td>
</tr>
<tr>
<td>desired position of right and left foot with respect to the upper body (two variables)</td>
<td>✓</td>
</tr>
<tr>
<td>front-back and right-left lean angle (two variables)</td>
<td>✓</td>
</tr>
<tr>
<td>difference between the current and the last measured values of the front-back and right-left lean angle (two variables)</td>
<td>✓</td>
</tr>
<tr>
<td>which foot is to be lifted next</td>
<td>✓</td>
</tr>
<tr>
<td>continuous functions of the <em>phase</em> variable</td>
<td></td>
</tr>
<tr>
<td>desired front-back lean angle</td>
<td></td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
CMAC it would cause a discontinuous CMAC output at the moment the biped switches from the lower right foot phase to the extend left leg phase. This is undesirable, since the output of the CMAC should be smooth in order for the gait to be smooth. Thus it is necessary to use a smooth function of the gait phase variable. Two such functions are:

\[
\begin{align*}
    f_1(\text{gait phase}) &= C_1 \sin\left(\frac{\text{gait phase}}{100}\right), \\
    f_2(\text{gait phase}) &= C_2 \cos\left(\frac{\text{gait phase}}{100}\right),
\end{align*}
\]

where the arguments of the sine and cosine are in degrees, and \(C_1\) and \(C_2\) are constants. These functions are used as state inputs to CMAC 4. Notice that the functions are smooth, and that the gait phase maps uniquely to a pair of values \((f_1(\text{gait phase}), f_2(\text{gait phase}))\).

A state of imbalance at a given time during walking generally results from incorrect postures at earlier times, rather than from an incorrect current posture. Thus, in the training of CMACs 2, 3, 4, and 5, the general technique of temporal difference learning [38] is used to distribute the information from the delayed supervised learning over sequential time steps.
6.3.7 Treating the Three-Dimensional Walking Problem as Two Two-Dimensional Problems

The problem of bipedal walking is a three-dimensional problem. However, the design of the controller presented here started with the assumption that the three-dimensional problem can be solved by treating the problem as two two-dimensional problems: one in the frontal plane and one in the lateral plane. As a consequence of this assumption motion in the plane of the floor is ignored. Another consequence is that each CMAC in the controller is designed to influence motion in only one plane - the frontal plane or the lateral plane. This approach creates problems because the motions in the frontal, lateral, and floor planes are coupled. The coupling gets more pronounced as the biped leans farther to one side in the frontal plane, and in the case when it is takes relatively long steps. Let us examine these two cases.

6.3.7.1 Ankle-Y Coupling

We shall first look at what we can call the "Ankle Y coupling". When the biped leans to one side it extends the opposite leg, while the "Ankle Y" joint of the supporting leg turns in the direction the biped is leaning. As mentioned in Section 6.1 the biped stands with bent knees in its initial position. Figure 6.11 shows the projection of the rotational motion of the "Ankle Y" joint on four planes:

a) the plane of the "Ankle Y" joint (joint plane view),

b) the sagittal plane of the biped (side view),
c) the frontal plane of the biped (front view),

d) the plane of the floor (top view).

The motion in Figure 6.11 has a non-zero projection in the frontal plane, and the plane of the floor. Therefore, even though the rotation of the "Ankle Y" joint is performed in order to lean the biped to one side, it also rotates the biped in the floor plane. Thus motions in these two planes are coupled. Notice that the biped does not have a joint that would rotate it around a vertical axis. This means that the projection of the "Ankle Y" joint rotation on the floor plane rotates the whole biped around a vertical axis going through the supporting foot, as shown in Figure 6.12. In other words, the rotation of the "Ankle Y" joint couples the motions in the frontal and the lateral planes as well. Thus, we can say that when the biped leans to one side, motions in all three planes are coupled.
An important effect of the "Ankle-Y coupling" is that the foot the biped is leaning away from is going to drift forward, as shown in Figure 6.12. In order to avoid this drift the foot that will be lifted has to be pushed back while the biped is leaning. This is the reason why during double support phases (when both feet are on the ground) the two feet are not moving backward the same amount (see Section 6.3.3). The backward motion is intended to offset the forward drift caused by leaning sideways. The approximate amount the foot has to be pushed back $d$, can be calculated from the following formula:

$$
 d \approx 2 \cdot \sin\left(\frac{tg^{-1}(tg_{\gamma_m} \cdot tg\alpha)}{2}\right) \cdot l,
$$

where $l$ is the distance between the biped’s legs, $\alpha$ is the angle between the floor and the shin link of the supporting leg, and $\gamma_m$ is the target right-left lean at the moment the foot is lifted. The derivation of this equation is given in Appendix 2.

**Figure 6.12** Rotation of the biped while leaning and the associated drift of the foot which the biped is preparing to lift

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
6.3.7.2 Long Step Coupling

Another important coupling occurs between the motions in the lateral and the frontal planes when the biped is taking relatively long steps. We can call this coupling the "long step coupling". If the biped could stand upright with straight legs the forces due to gravity would be trying to make it fall in either the forward-backward direction or the right-left direction. There would be no coupling between these forces - one set would be acting on the biped in the lateral plane while the other would be acting in the frontal plane. However, when the biped takes a step the forces acting on it are coupled. This coupling is due to the structure of the robot and its dynamics - mainly the heavy legs which carry five DC motors each. The longer the steps are the more pronounced the coupling is. Consequently, if the biped tends to be falling, it is falling both sideways and in the forward-backward direction. The uncoupled and coupled cases are shown in Figure 6.13. What we defined here as "long step coupling" has to be taken into account when training the controller CMACs. For example, CMAC 4 is used to learn adjustments to the right-left lean posture command such that once this command is implemented by the controller, the achieved right-left angle matches the target right-left lean better. The training is based on the difference between the desired and observed lean angles. However, what happens if the biped starts falling backward? As we can see from Figure 6.13, along with the front-back angle, the right-left angle is also going to change. This means that CMAC 4 will be trained to modify the desired right-left angle posture command differently than before. However, in this case CMAC 4 "was not at fault" for the biped falling - the fall was caused by instability in the lateral plane, and only the
projection of this fall affected the biped’s behavior in the frontal plane. Therefore, in this case CMAC 4 should not be trained.

![Diagram](image_url)

**Figure 6.13 Long step coupling**

### 6.4 Low Level Control

The block diagram of a simplified model of the low level controller is given in Figure 6.14. The low level controller receives commanded posture sequences from the high level controller, and translates these into desired actuator angle sequences. Next, the desired angles are modified in order to improve the walking performance. The resulting commanded angles are the inputs to PID controllers, which in turn output voltages to the joint motors.
6.4.1 Low Level Control Software Implementation

The structure of the low level control software is shown in Figure 6.15. The program has a command processing routine and four interrupt service routines. The interrupt service routines perform pulse width modulation and control operations, and handle communication with the high level logic. All the above routines have access to a number of global variables, and information is passed between them by modifying these global variables.
The command processing routine runs continuously. It handles commands coming from the high level control. The control interrupt service routine implements the translation of posture commands into the joint angle space, the modification of the joint angles to achieve better walking performance, and PID control of the motors. It is triggered by a SAB80C166 internal clock every 5 ms (this translates into a 200 Hz rate). The control routine receives feedback about the state of the robot from the biped hardware.

The biped’s motors are driven by PWM drive signals. The PWM interrupt service routine is triggered by another SAB80C166 internal clock every 0.8 ms (that is at a 1250 Hz rate).
The remaining two interrupt service routines handle serial communication with the high level control software. The interrupt routine handling transmission is triggered when the data in the *serial data transmit buffer register* of the SAB80C166 has been transmitted, while the service routine handling reception is triggered by data received and stored in the *serial data receive buffer register* of the SAB80C166.

The interrupt service routines handling the communication with the high level control have the highest level of priority. This ensures that data is not lost due to another service routine taking control of the processor while serial transmission is in progress. Communication between the low level and the high level control software is synchronized in such a manner that only one side transmits at a given time. Therefore, even though the reception handling routine executes at a slightly higher priority level than the transmission handling routine, these two routines will never try to interrupt each other and the difference in the levels of priority of these two routines is unimportant.

The interrupt service routine running at the third highest level of priority is the PWM handling routine, while the control routine runs at the lowest level of priority. The PWM service routine has to run every *0.8 ms* in order to successfully implement the PWM scheme. The execution time of the communication routines is short, but the execution time of the control routine is relatively long, therefore the PWM routine has to have higher priority in order to continuously control the ten biped motors.
6.4.2 Low Level Control Architecture

Figure 6.16 shows the block diagram of the low level controller. Again, variables in angle brackets are sampled physical measurements, and variables in lower case letters represent results of control calculations. The low level controller receives the commanded posture sequences from the high level controller, and translates them into actuator angles, using simplified kinematics. These angles are further modified by two blocks. The *Foot Contact Control* block is an integral controller, driven by the perceived centers of force on the feet. It changes the ankle actuator angles such that the feet are flat on the ground during the double support phase. The *Reactive Control* block is a modified PD controller, driven by the perceived body angle. It changes the ankle actuator angles to provide reactive lean angle control. This block is active throughout the gait, but it is much more important during the single support phase. Finally, the corrected actuator angles are implemented by the *Actuator PID Control and PWM* block. The output of this block are *pulse-width modulated (PWM)* voltages that drive the biped’s motors.
Let us now look at each of the blocks of the low level controller individually.

6.4.3 Simplified Kinematics

The simplified kinematics translates the commanded posture sequences into commanded actuator angles by assuming that there is no coupling between the frontal plane and the sagittal plane. As we saw in Section 6.3.7, this assumption does not hold too well, however it allows the calculations to be greatly simplified. The resulting implemented postures do not match the commanded postures, but the errors are repeatable.
6.4.4 Foot Contact Control

The *Foot Contact Control* block in Figure 6.16 represents an algorithm that has the very important role of preventing the rocking of the biped after a foot is lifted. Figure 6.17 shows the positions of the force sensors on the biped feet from the top view. The numbered circles represent the positioning of the force sensors. Let us define $FR_i$ ($i=1,...,4$) as the forces measured by the sensors on the right foot, and $FL_i$ ($i=1,...,4$) as the forces measured by the sensors on the left foot, where the subscripts correspond to the positions of the sensors. The figure also defines the meaning of the "right-left direction" and of the "front-back direction" in terms of the edges of the feet.

Let us now define the right-left, and front-back component of the Center of Force (CF) on each foot as:

![Figure 6.17 Foot with numbered pressure sensors - top view](image)
• right-left component of CF on right foot:

\[ RLCF_R = \frac{FR_2 + FR_4 - FR_1 - FR_3}{\sum_{i=1}^{4} FR_i} \cdot 40 \]

• front-back component of CF on right foot:

\[ FBCF_R = \frac{\sum_{i=1}^{2} FR_i - \sum_{i=3}^{4} FR_i}{\sum_{i=1}^{4} FR_i} \cdot 40 \]

• right-left component of CF on left foot:

\[ RLCF_L = \frac{FL_2 + FL_4 - FL_1 - FL_3}{\sum_{i=1}^{4} FL_i} \cdot 40 \]

• front-back component of CF on left foot:

\[ FBCF_L = \frac{\sum_{i=1}^{2} FL_i - \sum_{i=3}^{4} FL_i}{\sum_{i=1}^{4} FL_i} \cdot 40 \]

The values \( RLCF_R \), \( FBCF_R \), \( RLCF_L \), and \( FBCF_L \) are in the \([-40, 40]\) range. For the right-left components a positive value indicates that the CF is closer to the right edge of the foot. For the front-back components a positive value indicates that the CF is closer to the front edge of the foot.

During the extend left leg and the extend right leg phases the \( RLCF_R \) and \( RLCF_L \) move away from the center of the feet in the direction of leaning. The reasons for this
phenomenon is that the rigid body kinematics employed in the controller idealizes the biped's kinematics by not taking into account the flexing of the links and the foot plates, joint play, the fact that the feet are constructed with two plates with rubber pads in between, etc. As a result of the \( RLCF_R \) and \( RLCF_L \) moving in the direction of leaning, when the biped is ready to lift one foot up its feet are not parallel to the ground. This situation is illustrated as case a) in Figure 6.18. Consequently, when the foot is lifted up the biped loses the support the lifted foot provided, and it rocks back inward toward the lifted foot. The high level control recognizes this event as falling back on the lifted foot and interprets it as a case when the biped did not lean far enough out. Next time the biped leans further out and again falls back on the lifted foot. Eventually the biped tries to lean too far out and falls sideways away from the lifted foot. The high level controller interprets this event as leaning too far out, and decreases the lean angle for the next step, and the same course of events happens again. The biped never successfully lifts up the foot.

The biped feet can also be in a position where both feet are pressing harder with their outer or inner edge. These two situations are illustrated as case b) of Figure 6.18 for outer edges and as case c) of Figure 6.18 for inner edges.

In the front-back direction the biped feet can end up in a position where one foot is pushing harder with its "toes" while the other is pushing with its "heel". Again the feet are not flat on the ground, and this position creates instabilities during the double support phase. The illustration of this situation is case d) of Figure 6.18. Notice that if both feet
are in a position where either the "toes" or the "heels" are pushing harder, the biped's front back angle will change and the feet will be parallel to the ground.

\[ \text{Figure 6.18 Four cases when the feet are not parallel to the ground} \]

The Foot Contact Control algorithm changes the ankle actuator angles in such a way that the feet are closer to being flat on the ground during the double support phase. The algorithm uses integral control to move the feet to a position parallel to the ground. The scaled outputs of the algorithms integrators are added to the drive signals of the four ankle motors.

In the right-left direction the algorithm numerically integrates the sum and the difference of $RLCF_R$ and $RLCF_L$. The integral of the sum of $RLCF_R$ and $RLCF_L$ is a relatively large numerical value if the two variables have the same sign. This happens when the $RLCF_R$ and $RLCF_L$ move away from the center of the feet in the direction of leaning, as illustrated in case a) of Figure 6.18. The difference of the $RLCF_R$ and $RLCF_L$ is

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
significant if the two outer edges are pressing harder than the two inner edges, as in case b) of Figure 6.18, or if the two inner edges are pressing harder than two the outer edges, as in case c) of Figure 6.18. In the front-back direction the algorithm integrates the difference of $FBCF_R$ and $FBCF_L$. The value of the difference is significant if one foot is pushing harder with its “toes” while the other is pushing with its “heel”, as in case d) of Figure 6.18.

6.4.5 Reactive Lean Angle Control

The Reactive Lean Angle Control block in Figure 6.16 is an algorithm responsible for improving the tracking of the desired right-left and front-back body lean angles.

Consider the case when the biped is in the single support phase with the right foot on the ground. Let us assume that the biped is leaning further to the right than the target right-left lean angle generated by the high level controller. One way to correct the lean angle is to move the pivot point of the robot to the right. This in turn would push the biped body to the left. The pivot point can be moved to the right if the center of force on the right foot ($RLCF_R$) has not already reached the right edge of the right foot, that is if $RLCF_R < 40$. In order to move the pivot point to the right we can rotate the right foot in the counterclockwise rotation as viewed from the front of the biped. If the biped was not leaning far enough to the right the $RLCF_R$ would have to be moved to the left by rotating the foot in the clockwise direction as viewed from the front of the biped.
The mechanism described above is in effect a Proportional Controller. The input of the controller is the error in the lean angle, and the output is a correction signal that changes the position of the Right Ankle_Y joint such that the RLCF moves to the right or to the left. Of course, it would be better to move the pivot point in anticipation of a lean angle error, rather than to have to correct the error. For this purpose we could include derivative control. Pure derivative control would require finding the derivative of the right-left lean angle. An alternative is to use the filtered reading of the right-left rate gyroscope, since ideally the gyro’s output is proportional to the derivative of the right-left lean angle. In reality we first need to filter out the DC offset of the gyro signal, and this yields a signal that is approximately proportional to the derivative that we need. Using the low-pass filtered gyro reading introduces a shorter time delay than finding the derivative of the lean angle.

The above example describes the single support case when the right foot is on the ground. If the left foot was the supporting foot, or if both feet were on the ground, the same technique could be applied to correct the right-left lean angle. Furthermore the technique can be used to correct the front-back lean angle. The only difference for the latter case is that we have to change the position of the front-back component of the Center of Force (CF) on the feet.

The Reactive Lean Angle Control algorithm, shown in block diagram form in Figure 6.19, is a modified PD controller based on the above ideas. It tracks the target right-left and front-back lean angles by controlling the positions of the CFs on the two feet. The
algorithm monitors the values of the right-left and front-back lean angles. It finds the
differences between the target and the measured lean angles and uses these as inputs to
the proportional part of a modified PD controller. The inputs for the derivative part of the
controller are the high-pass filtered rate gyroscope readings. The algorithm interprets
scaled values of the modified PD controller outputs as target centers of force (TCF). We
can denote the right-left component of the TCF as $TCF_{RL}$, and the front-back component
of the TCF as $TCF_{FB}$. Note that the TCF is the same for both feet. Next, the algorithm
applies PD control to the difference between the current position of the CF, as measured
by the foot force sensors, and the position of the TCF. The outputs of these PD
controllers are then used to modify the drive signals for the motors of the ankle joints.
The corrections result in the joints turning in such a way that the differences between the
CFs and the TCFs are reduced. The above described mechanism is implemented for both
feet in single and double support phase.
6.4.6 Actuator PID Control and PWM

The commanded actuator angles are implemented through ten PID controllers - one for each motor. As explained in Section 4.1.4 the outputs of the PID controllers have to be pulse-width modulated. Note that this approach reduces losses in the motor driver chips.
6.5 Neural Network Training and Qualitative Results for the Static Walking Controller

The previous sections introduced the design of the Static Walking Controller. Let us now look at how the neural networks of the controller were trained and what kind of qualitative results the controller achieved.

Training of the biped typically proceeded in a similar fashion to training in the case of dynamic walking. The five CMAC neural networks were first trained during repetitive foot lift motions similar to marching in place (i.e. no attempts were made to translate the lifted foot). This was typically carried out for five minutes, with different settings for the desired foot lift height (in the 2 to 5 cm range). Frequent human support was required to keep the biped from falling during the first half of this training, and occasional support was required during the second half. Then, training of the five CMAC neural networks was carried out during attempts at walking (translating the lifted foot forward), for increasing step lengths, at a constant step rate. Again, frequent human support was required during early training for each new parameter setting, while less frequent support was required after 2 or 3 minutes of training at a given setting. After about 30 minutes of total training time, the biped was able to shift body weight from side-to-side while maintaining good foot contact, and to lift a foot off of the floor for a desired length of time, during which the foot could be moved to a new location relative to the body. The biped could start and stop on demand.
6.6 Quantitative Results

All data about the state of the biped was taken using a function of the high level control program. Data were logged after every execution of the high level control thread, that is every 28 ms. Desired values logged this way represent the complete set of values created by the high level digital controller during the logging period. The recorded measured values represent the sampled feedback about biped states received by the high level controller, since the low level software performs measurements every 5 ms.

Two important parameters used in the high level controller are the desired distance of the feet from the ground (foot lift), and the right-left component of the Zero Moment Point (ZMP<sub>RL</sub>). If FR<sub>i</sub> (i=1,...,4) are the forces measured by the four force sensors on the right foot, and FL<sub>i</sub> (i=1,...,4) are the forces measured by the four force sensors on the left foot then the ZMP<sub>RL</sub> is:

\[
ZMP_{RL} = \frac{\sum_{i=1}^{4} FR_i - \sum_{i=1}^{4} FL_i}{\sum_{i=1}^{4} FR_i + \sum_{i=1}^{4} FL_i} \cdot 40
\]

The ZMP<sub>RL</sub> is in the [-40, 40] range. It reaches its maximum when the weight of the biped is on the right foot, and its minimum when the weight is on the left foot. Note that, since the walking is static, the position of the ZMP is very nearly the same as the position of the normal projection of the center of mass (NPCM).
As mentioned in Section 6.5, the biped is trained at increasing step lengths. Figure 6.20 and Figure 6.21 show results of walking with 12 cm strides, obtained after the biped had been successfully trained at stride lengths of up to 10 cm. The top graphs in both figures show results logged shortly after the stride length had been increased to 12 cm, and the bottom graphs show results after approximately 5 minutes of training with 12 cm stride length. Therefore, the results show incremental changes in the walking performance. There are no results showing performance with 12 cm strides without training with shorter strides - the UNH biped would not be able to walk without first going through training with shorter stride lengths.

Figure 6.20 shows the relationship between the desired lift of the feet and the $ZMP_{RL}$ early in training, with 12 cm stride length and after 5 minutes of training with 12 cm stride length. The desired lifts of the two feet are combined into a single variable (desired right-left lift) by subtracting the desired lift of the left foot from the desired lift of the right foot. This way the desired right-left lift is positive when the desired right foot lift is greater than zero, and negative when the desired left lift is greater than zero. Note that the two desired lift values can never be greater than zero at the same time.

The high level controller uses another variable called the front-back zero moment point ($ZMP_{FB}$), introduced in Section 6.3.4. Figure 6.21 shows the relationship between the $ZMP_{FB}$ and the desired right-left lift early in training with 12 cm stride length and after 5 minutes of training with 12 cm stride length.
Figure 6.20 Desired right-left lift and measured ZMP\textsubscript{RL} early in training with 12 cm stride length (top graph) and after 5 minutes of training with 12 cm stride length (bottom graph)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 6.21 Measured ZMP_{FB} and the desired right-left lift early in training with 12 cm stride length (top graph) and after 5 minutes of training with 12 cm stride length (bottom graph)
6.7 Discussion

From the desired right-left lift in both graphs of Figure 6.20 we can see that the feet are commanded to be in the air for the same amount of time. However early in training they spend unequal time in the air. The top graph also shows unsuccessful attempts at lifting the left foot (short negative triangles) - the CMAC responsible for learning the necessary lean angle at the moment of lifting the foot was not sufficiently trained at this point in the walking. After about 5 minutes of training the feet are roughly spending equal time in the air, as desired, and the $ZMP_{RL}$ curve is smoother than early in training at this stride length. Also, there are no unsuccessful attempts at lifting a foot. All this shows that the neural nets were trained to augment the pre-planned motion control.

Figure 6.21 shows the relationship between the $ZMP_{FB}$ and the desired right-left lift. Ideally we would like the $ZMP_{FB}$ to be close to zero, however its value makes excursions as far as -12 and +40 early in training. After 5 minutes of training the excursions are noticeably reduced both in the single- and the double support phase, which means that the biped front-back balance has improved with training.

The biped could walk with continuous motion on flat surfaces at a rate of up to 2.2 steps per minute, with step lengths of up to 6 cm per step (12 cm stride lengths). However the biped required human supervision, as failures occurred every few minutes, and the biped would fall without support. The top forward progression velocity of the biped was limited by the fact that the gait and the controller were designed for static walking, and therefore the controller had difficulties dealing with effects due to the dynamics of the
system. The dynamic effects became more prominent at higher forward velocities. There
was no lower bound on the forward progression velocity - the biped can stand on one foot
indefinitely. Step length was limited by the kinematics of the robot.
This chapter presents the unified walking control algorithm. The main hypothesis of this research, which was presented in Section 2.1, proposed that identification of the characteristics of dynamic and static walking, and the examination and comparison of these characteristics, could be used to create the new algorithm. Thus the first step in the development of this algorithm was to identify the major characteristics of the dynamic and static walking gaits.

7.1 Characteristics of Dynamic Walking

During the analysis of the dynamic walking controller seven important characteristics of the gait were identified:

1. The controller utilized the natural dynamics of the robot.
2. The controller lacked reactive control.
3. The biped would continue moving in the right-left direction after it lifted a foot.
4. Centers of force (CFs) measured on the feet were not utilized extensively.
5. The supporting foot had to be roughly static under the upper body during stepping with the other foot.
6. The desired front-back lean angle was constant.
7. The controller used simplified frontal and lateral plane kinematics to translate posture commands into joint position commands.
Let us now look at each of the above characteristics individually.

### 7.1.1 Utilization of Natural Dynamics

The most important characteristic of the dynamic walking gait implementation, which was presented in Chapter 5, is that the controller tried to make use of the natural dynamics of the biped hardware. The biped's gait required a right-left swinging motion in order to lift a foot. The controller was programmed to keep the rate of the right-left swinging of the biped in the vicinity of the natural frequency of this swinging motion. This way, instead of "fighting" the biped's mechanical hardware, the controller was making use of its properties. This approach reduced the need to accurately model the biped dynamics. However, the lack of an accurate knowledge of the dynamics represented a problem when the controller tried to reduce the stepping rate.

In single support phase the biped can roughly be modeled as an inverted pendulum. As explained in Section 2.4, for a two-dimensional inverted pendulum model of the biped, the relationship between the time period the foot spends in the air ("foot lift period"), the lean angle at the moment the foot is lifted ("lean at foot lift"), and the sideways speed at this moment ("v"), is represented by the curves in Figure 7.1 (same as Figure 2.4). The dynamic walking controller produced walking rates that resulted in the lifted foot remaining in the air for approximately 0.3 sec. The corresponding pairs of lean at foot lift, and foot lift period were in the natural frequency region outlined in Figure 7.1. When the controller was driving the robot at a stepping rate higher than the natural frequency the "foot lift period" became shorter. The controller did not have a problem
with these stepping rates because they required less accurate control than walking in the vicinity of the natural frequency. This can be seen from the fact that the curves have a smaller slope for shorter "foot lift period" values, and the curves corresponding to different sideways speeds "v" are closer together. Conversely, for longer "foot lift period" values, the curves have a larger slope, and curves corresponding to different values of "v" are further apart. Consequently, slow walking requires a more precise control of the biped's position and speed. The dynamic walking controller was not able to provide the accuracy required for slow walking.

![Figure 7.1 Relationship between the time period the foot spends in the air, and the lean angle and the sideways speed at the moment the foot is lifted](image)

**Figure 7.1** Relationship between the time period the foot spends in the air, and the lean angle and the sideways speed at the moment the foot is lifted
7.1.2 Lack of Reactive Control

The dynamic walking controller implemented adaptive closed loop control without reactive control. The controller got better by learning not to repeat mistakes, however it did not have a mechanism that would react to mistakes when they occurred.

7.1.3 Right-Left Motion Continued After Lifting a Foot

In the case of the dynamic walking gait, a foot was moved by first generating a lateral momentum toward the opposite side. The foot could then be lifted and moved to a new location. The biped's right-left motion would continue in single support phase without explicit control by the walking algorithm. At the moment a foot was lifted the biped body was at a certain right-left angle. The first part of the sideways motion in single support phase was due to the biped body's inertia, which carried it past its position when the foot was lifted. However, the resulting gravitational force when the foot was lifted, broke the inertia and allowed the biped to fall back onto the lifted foot. Thus the second part of the motion was due to the biped falling back toward the lifted foot.

7.1.4 Centers of Force (CFs) on the Feet not Utilized Extensively

The dynamic walking controller utilized the CFs measured on the feet only to train one of its CMAC neural networks. This CMAC adjusted the positions of the "Ankle Y" joint of the supporting foot to achieve better foot contact in the right-left direction in single support phase.
7.1.5 The Supporting Foot has to be Under the Upper body

The dynamic gait controller was designed in such a way that the gait generator produced a predefined sequence of posture commands, which were then modified by CMAC neural networks. The predefined sequence was very simple: the biped had to lean to one side, pick up a foot and put it forward, and then repeat the motion to the other side and with the other foot. One important preprogrammed feature of the gait was that, while one foot was lifted, the supporting foot had to be held steady under the biped upper body. The biped was not pushing off with the supporting foot. Moving the supporting foot did not produce stable walking.

7.1.6 Constant Desired Front-Back Lean Angle

The dynamic gait controller aimed to keep the front-back lean angle constant. The controller was designed with this characteristic as a requirement. This meant that the front-back angle was not used to try to achieve better front-back stability. Instead, a constant front-back angle was a criterion of stability.

7.1.7 Simplified Kinematics

The system used simplified kinematics to translate commanded posture sequences into commanded joint angle sequences. The simplified kinematics did not take into account the coupling between the frontal and lateral plane motions. This reduced the complexity of the necessary calculations, while providing good results.
7.2 Characteristics of Static Walking

During the analysis of the static walking controller ten important characteristics of the gait were identified:

1. The biped was moving slow enough that the dynamics could be neglected.
2. Reactive lean angle control was used along with adaptive control.
3. The preprogrammed part of the gait assumed that the biped stopped the right-left motion in single support phase.
4. Right-left lean posture command corrections were performed by both a CMAC neural network and a PID controller.
5. The CFs measured on the feet were used both in the high level and the low level controller.
6. The supporting foot had to be under the upper body during stepping with the other foot.
7. Integral control was used to provide good foot contact in double support phase.
8. The desired front-back lean angle was constant.
9. The desired front-back lean angle was $9^\circ$ which was less than the desired front-back angle in the case of dynamic walking ($15^\circ$).
10. The controller was not successful at stepping rates of more than 2.2 step per minute.
11. The controller used simplified frontal and lateral plane kinematics to translate posture commands into joint position commands.

Let us now take a look at each of the above characteristics individually.
7.2.1 Slow Motion Allowed Dynamics to be Neglected

The biped’s stepping rate did not exceed 2.2 steps/minute. This rate allowed the controller to be designed without having to model the dynamics very accurately.

7.2.2 Reactive Control Used Along with Adaptive Control.

The low level controller included an algorithm for reactively adjusting the right-left and front-back lean angles to the target lean angles. The adaptive part of the control, based on CMAC neural networks, was not able to implement the exact angles required for lifting a foot. With the presence of reactive control, the lean angles were implemented with sufficient accuracy for static walking.

7.2.3 Right-Left Lean Angle Commanded to be Constant in Single Support Phase

In the case of static walking, in order to move a foot, it is necessary to first counterbalance that foot by leaning the biped’s upper body toward the opposite side. The foot can then be lifted and moved to a new location. The right-left lean angle was commanded to be constant in single support phase.

7.2.4 Right-Left Lean Posture Command Corrections Performed by Both a CMAC and a PID Controller

The right-left lean posture command was affected by both a CMAC neural net and a PID controller, as described in Section 6.3.5. The idea was to make the job of the CMAC easier - the CMAC would “learn” to implement the lean angle “almost right”, and the PID
controller would provide small corrections. The problem with this setup was that the PID control caused instability when the stepping rate was increased.

7.2.5 Extensive Use of Measured Centers of Force (CFs)

In the static gait controller the CFs measured on the feet were used in three places:

- the integral control of the position of the biped ankle angles;
- a CMAC neural network responsible for right-left balance control;
- the reactive control algorithm.

This extensive use of the CFs was important because they are the most reliable measure of the biped's stability. If the biped is falling in one direction the CF readings on the feet will clearly show this. For example, if the biped is falling sideways to the right, the value of the right-left component of the CF on the right foot will be $RLCF_R = 40$. The biped's stability cannot be deduced unequivocally from the body lean angles - in different situations different body angles could lead to a fall.

7.2.6 The Supporting Foot has to be Under the Upper Body

As explained in Section 6.3.3, the static gait controller's gait generator produced a predefined sequence of posture commands, which were then modified by CMAC neural networks. The predefined sequence commanded the biped to lean to one side, pick up a foot and put it forward, then repeat the motion to the other side and with the other foot. As in the case of the dynamic gait, an important feature of the static gait was that while one foot was lifted, the other (supporting) foot had to be held steady under the biped's
upper body. The biped was not pushing off with the supporting foot. When the supporting foot was under the biped's upper body the metal structure (the "bones") carried a large portion of the weight. When the supporting foot was used to push the biped forward the load on the "Hip" and "Ankle" joint motors of the supporting leg increased to the point where the motors were not able to implement the commanded angles. This is illustrated in Figure 7.2.

Figure 7.2 It is important to keep the supporting foot under the upper body

7.2.7 Integral Control Used to Provide Good Foot Contact in Double Support Phase

As discussed in Section 6.4.4 the low level controller implemented integral control to provide good foot contact in double support phase. Good foot contact was essential in achieving stable walking - without it the biped tended to rock back onto the lifted foot.
One problem with having an integral controller in the system was stability. The biped’s walking rate was limited by the delay introduced by the integral control of foot contact.

7.2.8 Desired Front-Back Lean Angle Constant

As in the case of dynamic walking, the front-back lean angle was not variable - the controller did not adjust this angle continuously in order to try to improve stability. Rather, the value of the front-back angle was a predefined requirement.

7.2.9 Desired Front-Back Lean Angle Less than for Dynamic Walking

The dynamic walking was implemented with a constant front-back lean angle of 15°. For static walking the value of the desired front-back lean angle was decreased from 15° to 9°. When the lean angle was 15° in single support phase the “Hip” joint motors could not counteract the torque created by the weight of the upper body and the lifted leg. Figure 7.3 illustrates how the different lean angles result in different torques on the “Hip” joint. In the extreme case, when the biped’s front-back lean angle is zero, the torque is also zero. If the biped leans forward the torque is non-zero, and the more the biped leans the larger the torque gets - this is true up to the point where the center of mass of the upper body and the lifted leg (CM) are in the same line as the “Hip Y” joint.

In the dynamic walking case the length of time the lifted foot spent in the air was less than 0.5 sec, while in the case of static walking it was on the order of 5 sec. The “Hip” motors were strong enough to implement the desired angles with relatively high accuracy.
for short periods of time, even when the biped was leaning 15°. However, for longer periods they were not able to supply the necessary torque. Therefore, for static walking the lean angle was reduced to 9°.

\[ T = 0 \quad T = r \cdot F_n \]

**Figure 7.3** Torque (T) on "Hip" joint due to body lean

### 7.2.10 The Controller Could Only Implement Slow Walking Rates

The controller was not successful at walking rates of more than 2.2 steps/minute. The controller was designed to work at low stepping rates. Mechanisms that performed well at these low rates failed at higher rates. The CMAC neural nets were not geared toward higher speeds, or toward allowing motion at various speeds. The system also had integral and PID controllers in two subsystems - these controllers worked well for low step rates, but became unstable for higher rates.
7.2.11 Simplified Kinematics

As in the case of dynamic walking, the posture commands were translated into joint angle commands by simplified kinematics, which ignored the coupling between motions in the frontal and the lateral planes. As in the case of dynamic walking, this reduced the complexity of the necessary calculations, while providing good results.

7.3 Unified Controller Design Hypotheses

The previous two sections described the most important characteristics of the static and dynamic gaits. This section discusses how the examination and comparison of these characteristics led to the formulation of two of the major hypotheses that were presented in Chapter 2, and seven secondary hypotheses.

7.3.1 Major Hypotheses

The first similarity between the two gaits that was found in the process of this research was described in Section 2.2. It was the property of both static walking and full foot contact dynamic walking that the ZMP has to be within the stability region, as illustrated in Figure 2.1. The second major hypothesis, introduced in Section 2.2 was based on this similarity.

The fourth major hypothesis of this research, introduced in Section 2.4, proposed that the unified bipedal walking controller should be based on adaptive closed-loop control. This hypothesis was initially based on the property of the dynamic gait controller that slow walking cannot be implemented because the controller utilizes the dynamics of the
mechanical hardware and lacks reactive control. The hypothesis was given more credibility when it was found that the low level reactive lean angle control was essential in achieving static walking.

7.3.2 Secondary Hypotheses

The examination and comparison of the characteristics of the two implemented gaits resulted in the following seven secondary hypotheses, denoted SH:

SH #1. In single support phase the body's right-left position should be actively controlled to help smoothly break any existing momentum which is swinging the biped away from the lifted foot. In the case of dynamic walking, the gait was designed such that the biped relied on the lateral momentum being broken by lifting the foot off the ground. The right-left angle was not actively controlled. In the case of static walking, the sideways momentum was always negligibly small, thus the biped could be commanded not to move sideways in single support phase. The new controller had to be able to deal both with gaits that have lateral momentum at the beginning of the single support phase, and with those that do not. This hypothesis proposed that the important consideration was how to smoothly break the sideways momentum if it does exist, and it said that this should be done by actively controlling the right-left angle in single support phase, such that its value helps break the momentum in a smooth manner.

SH #2. If only a CMAC neural network is used to perform corrections of the right-left lean posture command, the biped's performance at higher stepping rates will be satisfactory. As explained in Section 6.3.2, it is necessary to perform
corrections of the right-left lean and the front-back lean posture commands in order to implement reactive control of the lean angles in the low level controller. In the case of static walking this was done in the high level controller by a PID controller in combination with a CMAC neural net. However, the PID control caused instability when the stepping rate was increased. This hypothesis proposed that by using only a CMAC neural network the lean angles would be corrected successfully, while avoiding instability at higher stepping rates.

SH #3. Providing good foot contact in double support phase is necessary for successful walking at variable speeds. Securing good foot contact in the double support phase was essential to achieving static walking. In the case of dynamic walking good foot contact in the double support phase was less important because, due to relatively large sideways speeds, the biped did not have a problem of rocking back onto the lifted foot. Instead, variations in foot contact resulted primarily in variations in single support duration. Since the new controller would have to deal with both slow and fast walking, it seemed reasonable to expect that good foot contact in double support phase will be a prerequisite for successful walking.

SH #4. Good foot contact in double support phase can be achieved by correcting the positions of the ankle joint positions by using a CMAC neural network. In the case of static walking good foot contact is achieved with the help of an integral controller. This controller was not stable at higher stepping rates. This hypothesis says that the integral controller can be replaced by a CMAC neural network, that will perform well at various stepping rates.
SH #5. *The preprogrammed part of the gait should keep the supporting foot under the upper body.* For both the implemented static and the dynamic gaits the walking was successful only when the supporting foot was commanded to be steady under the upper body. It was expected that the new controller would perform in the same manner.

SH #6. *Keeping the commanded front-back angle constant at 9° will result in stable walking.* Both implemented controllers used a constant front-back lean angle. In the case of dynamic walking this angle was 15°, however the static gait walking required a front-back lean angle of 9°. In order to be able to implement both static and dynamic walking the front-back angle was set to 9°.

SH #7. *The new controller can use simplified kinematics to translate posture commands into commanded joint angles.* Both the dynamic and the static controller used simplified kinematics with good results. It seemed reasonable to expect that the new controller would perform well with it as well.

The gait and the control architecture of the unified biped walking scheme was based on the major hypotheses presented in Chapter 2 and the secondary hypotheses presented in this section. Let us first introduce the biped gait, and then take a look at the control architecture.
7.4 Walking Gait

The walking gait of the robot using the unified control algorithm combined elements of both previously implemented gaits. It also took into account the first and the fifth secondary hypotheses (SH #1 and SH #5).

As in the case of dynamic and static walking, due to the distribution of mass within the biped structure, in order to move a foot it is necessary to first lean the biped’s upper body toward the opposite side. Since the unified controller has to deal with both static and dynamic walking, it was assumed that the sideways motion will create a momentum that cannot be neglected. Following SH #1, the biped gait was designed to help smoothly break this momentum after the foot is lifted, by smoothly decelerating the sideways motion while the foot is being raised. If the gait generator was programmed to stop the sideways motion while the lifted foot is in the air (as was the case for static walking), the upper body would still continue moving away from the lifted foot, with a decelerating motion. The gait generator creates a commanded sequence of postures that takes into account the fact that the upper body will need some time to decelerate. This approach makes the control less abrupt, and results in smoother walking motions. Once the foot reaches the highest point of its trajectory, the right-left angle starts changing to bring the biped back towards the lifted foot. When the foot lands the biped’s sideways speed levels off. The right-left speed profile of the sideways motion implementing this idea is outlined in Figure 7.4. Notice that the profile is designed to be continuous. This is important since a discontinuous speed profile corresponds to jerky motion, which in turn can easily lead to instability, due to its high frequency content.
The biped gait is shown in Figure 7.5. The biped goes through double support phases, when both feet are on the ground, and single support phases when only one foot is on the ground. The arrows on the upper body show the direction in which the upper body is moving (right or left). The arrows next to the lifted feet show the direction in which those feet are moving (up or down).
As for both dynamic and static walking, leaning to one side is achieved by starting the biped with bent knees and extending the opposite leg. This method of leaning is illustrated by bent legs in Figure 7.5. Alternative ways of leaning the upper body are leaning it from the hips or from the ankles, but the biped’s DC motors are not strong enough for these. Notice also that, as outlined in Figure 7.5, in single support phase the biped can roughly be modeled as in inverted pendulum.

The biped’s walking gait in the unified control algorithm is logically divided into the same six phases as the static walking gait:

1. **extend left leg phase**: In this phase the biped leans from left to right by extending the left leg, in order to take weight off the left foot. The sideways speed is preprogrammed to be constant. In the front-back direction the biped feet are moving relative to the upper body such that at the end of this phase the right foot is under the upper body, and the distance between the right and left foot is the same as at the beginning of this phase.

2. **lift left foot phase**: At the end of the extend left leg phase the position of the biped (in the case of static walking), or both the position and the sideways momentum generated by the left-to-right motion (in the case of dynamic walking), allow the left foot to be lifted, and the lift left foot phase starts. While in this phase, the upper body is moving away from the lifted left foot with speed decreasing to zero. The right foot is kept under the upper body, while the left foot moves forward.

3. **lower left foot phase**: Once the left foot reaches the highest point of its trajectory this phase starts, and the foot is lowered back onto the ground. At the start of this phase
the sideways speed is zero. As time progresses the biped is moving towards the lifted left foot with increasing speed, until the left foot hits the floor. When the left foot makes contact with the ground this phase ends, and the _extend right leg phase_ starts. While the left foot is in the air the right foot is kept under the upper body, and the left foot is moved forward.

4. _extend right leg phase_: Symmetrical counterpart of the _extend left leg phase_.

5. _lift right foot phase_: Symmetrical counterpart of the _lift left foot phase_.

6. _lower right foot phase_: Symmetrical counterpart of the _lower left foot phase_.

Note that the motion of the feet is based on the requirement that the biped has to move forward and on SH #5, which proposes that the supporting foot should be preprogrammed to be under the upper body.

### 7.5 Adaptive Control of Walking

Now that we have introduced the biped gate for the unified control algorithm, let us examine the unified control architecture of the biped. As in the cases of dynamic and static walking, the control architecture consists of a high- and a low level controller. Figure 7.6 shows in block diagram form how the control architecture, including both the low and the high level control, fits into the overall biped system. Comparing this block diagram to the block diagram of the overall system in the case of static walking (Figure 6.2), we can see that the unified controller architecture is very similar to the static controller architecture.
The high level controller generates sequences of commanded joint angles. This is done in four steps. First, the controller generates posture sequences based on a simplified model of the biped. Next, these sequences are modified using neural networks. The modified sequences are pre-planned, but adaptive, sensory triggered, smooth sequences of desired postures, called "commanded postures" in Figure 7.6. The commanded posture sequences are transformed into sequences of desired actuator angles. The resulting "desired angles" are modified using neural networks to obtain the "commanded angles".

The low level controller performs two steps. First, the actuator angle sequences received from the high level controller are modified as a result of reactive control of the right-left and front-back angles. Next, the modified actuator angle sequences ("modified angles") are implemented using PID control.

Figure 7.6 Overall biped system block diagram
As in the case of static walking, the overall control strategy (including both the high and the low level control) uses a combination of pre-planned, but adaptive, smooth motion sequences with sensory triggers, and reactive closed-loop control.

### 7.6 High Level Control

The block diagram of a simplified model of the high level controller is given in Figure 7.7. As in the case of static walking, based on a crude model of the biped, the high level controller creates a sequence of "simple postures". Sequences of simple postures will not necessarily lead to stable walking, due to the insufficient accuracy of the model of the system. In order to achieve stable walking the simple sequences are modified by the outputs of several CMAC neural networks, and the "simple postures" are changed into "commanded postures", such that the gait is more stable. The "commanded postures" are translated into desired actuator angles, and these "desired angles" are modified using CMAC neural networks in order to further improve stability.

![Simplified model of the high level control](image)

**Figure 7.7** Simplified model of the high level control
7.6.1 High Level Software Implementation

The high level control is implemented as a digital controller on a 200 MHz PentiumPro-based personal computer, under the Microsoft NT operating system.

The structure of the high level control software is identical to the structure of the static walking control software. The communication and control threads execute every 28 ms, which translates into a 35.71 Hz control rate. The difference between the two controllers is in the functionality of the control thread, which was changed in order to implement the new control architecture.

7.6.2 High Level Control Architecture

Figure 7.8 gives the outline of the high level controller. Variables in angle brackets are sampled physical measurements. Variables depicted in capital letters are parameters set by the user. Variables in lower case letters represent results of control calculations.
The high level controller evolved from the static walking high level controller, and it was designed taking into account the seven secondary hypotheses.

The high level controller has seven major components represented in the block diagram of Figure 7.8. The walking gait is initiated by the gait generator. The gait generator outputs seven posture parameters, which form \textit{posture}_1. These parameters are modified...
by CMAC neural nets to produce a posture command consisting of eight posture parameters, which in turn form the commanded posture. Again, eight posture commands are used in order to include the right-left target lean (rlt), which is used by the low level controller in the modified PD control of the right-left lean angle. As in the case of static walking, the low level controller also implements a modified PD controller for the front-back lean angle. However, the value of the desired front-back lean angle (9°) is hard-coded in the low level controller, and does not have to be passed to it. The commanded posture parameters are transformed into actuator angle commands by the simple kinematics block of the controller. At the output of the simple kinematics block are ten desired biped joint angles, denoted joint angles in Figure 7.8. Two of the joint angles values, ray and lay, that is the positions of the right an left Ankle Y joints, are modified by the outputs of CMAC 5. CMAC 5 corrects the positions of the two joints such that better foot contact is achieved in double support phase. Finally, the output of the controller consists of ten desired joint angles, denoted commanded joint angles, and the right-left target lean.

Let us now look at each of the blocks of the high level controller individually.

7.6.3 The Gait Generator and Learning to Lift the Feet (CMAC 1)

The gait generator of the controller is very similar to the gait generator of the static walking controller: it initiates the side-to-side and foot movement motions in the walking process, and it is based on the same heuristics and approximate biped model as the static
walking gait generator. It also has preprogrammed values for the sideways lean angles at the moment the feet are to be lifted. In the following, the lean angles that have to be reached in order to lift a foot will be called (right or left) lift lean angles.

In Figure 7.8 CMAC 1 outputs corrections of the preprogrammed values of the lift lean angles. An insufficient lean causes the foot to lift for too short a duration (or not at all), while too much lean causes the foot to lift for too long a duration (or for the robot to fall over laterally). Let us examine the factors that the proper lean depends on. In the case of static walking we assumed that the biped moves from one extreme sideways position to the other in a smooth motion, and that the CMAC only has to take into account which foot will be lifted. All other dynamic and kinematic factors that could influence the required lean were neglected. Therefore, the desired lean angle was calculated at the beginning of the leg extension phases (extend right leg phase and extend left leg phase).

In the case of the unified control algorithm the (right and left) lift lean angle is calculated at the beginning of every leg extension phase (extend right leg phase and extend left leg phase respectively). However, the first thing that walking experiments showed was that the sideways motion was not smooth, especially in the beginning of training with new settings. The biped’s right-left angle had local minima and maxima, which meant that during the sideways motion the biped sometimes stopped and reversed direction for short periods of time. These jerky motions had a large effect on the magnitude of the sideways angle that was needed for successful stepping. For example, if the biped was in the process of leaning to the right and for some reason this motion was stopped for a short
period of time, the momentum that had built up in the process of leaning disappeared. This meant that the biped now had to lean further out to compensate for the lost momentum. Keeping the previous value of the required lean angle would have meant that the biped would not be able to lift the foot off the ground. This in turn would be interpreted by CMAC 1 as a case when the biped did not lean far enough, and the neural net would be trained to increase the lean angle. However, if the short stop in the sideways motion disappeared the biped would now be leaning too far out. A similar problem would occur if the biped reversed direction for a short time, and when it resumed moving in the original direction, the momentum was greater than before the reversal of direction. In this case the biped needed to reduce the \textit{lift lean angle}.

In order to deal with the problems created by the jerky sideways motion of the biped the gait generator can detect local extremums. When an extremum is detected the necessary correction of the preprogrammed \textit{lift lean angle} is updated using CMAC 1. This is illustrated in Figure 7.9. The figure shows the measured right-left lean of the biped, and the output of CMAC 1, which is used to calculate the necessary lean angle. The right-left lean angle does not change smoothly. During leaning from the right to the left, that is while the value of the right-left lean angle is decreasing, the lean angle has local maxima that correspond to the biped reversing its motion for short periods of time. The controller detects local maxima by monitoring the trend of the right-left lean - if the right-left lean is increasing for at least five control cycles, that is for $5 \cdot 28ms = 140ms$, a local maximum is detected. If a local maximum is detected CMAC 1 is interrogated to find a new correction of the \textit{lift lean angle}. This is reflected in Figure 7.9 as changes in the value of
the output of CMAC 1. During leaning from left to right, that is while the value of the right-left lean angle is increasing, the lean angle has local minima that are detected if the lean angle is decreasing for at least five cycles. If a local minimum is detected CMAC 1 provides a new correction of the required lean angle. Note that a local extremum is detected only if the measured lean angle is monotonically changing in the direction opposite from the preprogrammed direction, for five consecutive control cycles. This assures that noise does not cause the controller to falsely identify a local extremum. Figure 7.9 also shows that the gait generator takes into account the output of CMAC 1 every time the biped starts a new extend leg phase just as the static controller’s gait generator did.

![Figure 7.9 CMAC 1 outputs a new correction at the beginning of every extend leg phase. Local minima and maxima in the right-left lean angle cause the reevaluation of the CMAC 1 correction.](image)

149
The state vector of CMAC 1 has the following nine elements:

- foot to be lifted (represented by a numerical value);
- desired sideways speed;
- maximum foot lift magnitude;
- relative position of the right foot with respect to the hip;
- relative position of the left foot with respect to the hip;
- measured front-back angle;
- measured right-left angle;
- difference between current and last measured front-back angle;
- difference between current and last measured right-left angle.

The above variables define the state of the biped in sufficient detail for CMAC 1 to be able to predict the necessary lift lean angle. CMAC 1 is trained after every completed step, that is twice during a gait cycle. The state vector used for training is the last input state vector, which is either the state vector at the beginning of the extend leg phase, or the vector at the moment a local extremum was detected. This means that the success of CMAC 1 is evaluated according to the performance of the biped after the last correction of the preprogrammed lift lean angles.

The training procedure applied for CMAC 1 is very similar to the procedure described for training CMAC 1 of the static gait controller (see Section 6.3.3). Here is a short summary of the training procedure. As in the case of static walking, the training error is derived from the foot pressure sensor readings. If a foot is commanded to be 1.5 cm off
the ground, it is still on the ground, as detected by the pressure sensors on that foot, the gait generator decides that the sideways lean was not sufficient to perform a step. The step is abandoned, and CMAC 1 is trained as though the biped fell onto the lifted foot. If the foot is successfully lifted off the ground by the point it is commanded to be off the ground by 1.5 cm the foot is lifted to the commanded height of lift$_{\text{preparation}}$ + lift magnitude, where lift$_{\text{preparation}}$ is the value of the commanded lift at the moment the foot brakes contact with the ground, as detected by the pressure sensors, and lift magnitude is the desired actual height of the foot when it is fully lifted. In the latter case CMAC 1 is trained to help keep the feet in the air such that:

$$\text{lift}_\text{preparation} + \text{lift}_\text{land} = 1.5 \text{ cm},$$

where lift$_{\text{land}}$ is the value of the foot lift command at the moment when the foot makes contact with the ground again. If the biped falls over sideways CMAC 1 is trained to make the lift lean angle smaller by a preset value, and in the case the foot did not lift off, or fell back on the ground shortly after being lifted, CMAC 1 is trained to make the lift lean angle larger by a different preset value.

We can see that the way CMAC 1 is used in combination with the gait generator in the unified controller is more complex than in the case of the static walking controller. Another difference between the two gait generators is the fact that the unified controller’s gait generator is programmed to continue the sideways motion of the biped after a foot is lifted, in order to implement the gait described in Section 7.4. The biped is
preprogrammed to continue moving away from the lifted foot in the *lift foot phases* (*lift left foot* and *lift right foot*), and to move towards the lifted foot in the lower foot phases (*lower left foot* and *lower right foot*). This is illustrated in Figure 7.10 which shows three *posture* parameters: the *right foot lift* (*rfl*), the *left foot lift* (*lfl*) and the *right-left lean* (*rll*). Notice that *rll* does not level off when the foot lift parameters are not zero, as it did in the case of static walking (see Figure 6.8).

**Figure 7.10** *The sideways motion continues during single support*

Another important feature of the gait generator is that it is designed to reduce the shock caused by the landing of the lifted foot. If a foot that is descending with a constant speed hits the ground, and continues its downward motion at the same constant speed, this will break the smooth sideways motion of the biped. The biped will rock from one foot to the other, and possibly even fall. However, if the motion of the descending foot is slowed
down after it hits the floor, the impact of this hit will not be as severe as in the previous case. As illustrated in Figure 7.11, the gait generator slows down the descent of the lifted foot after it hits the floor. Figure 7.11 shows the relationship between the right foot lift and left foot lift posture parameters, and the right-left ZMP. The right-left ZMP was defined in Section 6.6 as:

\[
ZMP_{RL} = \frac{\sum_{i=1}^{4} FR_i - \sum_{i=1}^{4} FL_i}{\sum_{i=1}^{4} FR_i + \sum_{i=1}^{4} FL_i} \cdot 40,
\]

where \( FR_i, (i=1,...,4) \), are the readings from the pressure sensors on the right foot, and \( FL_i, (i=1,...,4) \), are the readings from the pressure sensors on the left foot. The \( ZMP_{RL} \) is in the \([-40, 40]\) range. When the right foot is lifted, the weight of the biped is on the left foot, and \( ZMP_{RL} \) is \(-40\). During its descent the right foot makes contact with the ground, as can be seen from the fact that the value of \( ZMP_{RL} \) increases to about \(-20\). At this point the descent of the right foot is slowed down - this is reflected in the change of the slope of the right foot lift curve. In a similar fashion the descent of the left foot is slowed down after it hits the ground.
Figure 7.11  The extension of the lifted foot slows down after it contacts the foot. This reduces the shock caused by landing.

In Figure 7.10 we illustrated three posture parameters generated by the gait generator - the right foot lift (rfli), the left foot lift (lfli) and the right-left lean (rlti). Let us now take a look at the remaining four posture parameters. The right foot forward (rffi) and the left foot forward (lffi) parameters are illustrated in Figure 7.12, along with rfli and lfli. The figure shows that, following the proposition of the fifth secondary hypothesis (SH #5), the supporting foot is preprogrammed to be under the upper body in single support phase. This can be seen from the fact that the foot forward posture parameter of the supporting foot is zero while the biped is in single support phase. In single support phase the lifted foot is moved forward (by approximately 5-6 cm in this case). In double support phase both feet are pushed back relative to the hips. The foot that just landed is getting into position to be the supporting foot. The backward motion
of the foot that will be lifted next has two components. One component pushes the foot back the same amount as the other foot is pushed, in order to keep the distance between the feet constant. The second component prevents this foot from drifting forward. The cause of drifting is explained in Section 6.3.7.1.

![Figure 7.12 Four posture, parameters](image)

**Figure 7.12** *Four posture, parameters*

The remaining two *posture* parameters are held constant - the *height* is always 40 cm, and the *front-back lean* is set to be 9°.

As in the case of static walking, the gait generator relies on sensory triggers to create posture sequences. Sensory triggers are the instances of each foot contacting or breaking contact with the ground, as detected via the foot force sensors, and the biped body reaching angles sufficient for successful foot lift, as detected by the virtual body angle.
sensors. The foot lift phases are commenced when the biped reaches a certain sideways lean angle, as detected by the virtual sensors. The foot lowering phases end, and the leg extension phases start, when a foot hits the ground, as detected by the pressure sensors on the feet.

Note that the gait generator creates gaits with different step lengths, foot lift magnitudes, foot lifting speeds, and sideways leaning speeds depending on user input.

7.6.4 Front-Back Balance Control (CMAC 2)

CMAC 2 is used to control the instantaneous front-back position of the hips relative to the feet. CMAC 2 is the counterpart of CMAC 3 of the static walking controller. In Section 5.3 we defined the front-back zero moment point ($ZMP_{FB}$). If $FR_1$, $FR_2$, $FL_1$, and $FL_2$, are the readings from the pressure sensors in the front of the two feet (the “toes”), and $FR_3$, $FR_4$, $FL_3$, and $FL_4$ are the readings from the pressure sensors in the back of the two feet (the “heels”), the $ZMP_{FB}$ is calculated as:

$$ZMP_{FB} = \frac{\sum_{i=1}^{2} FR_i + \sum_{i=3}^{4} FL_i - \sum_{i=1}^{2} FR_i - \sum_{i=3}^{4} FL_i}{\sum_{i=1}^{4} FR_i + \sum_{i=1}^{4} FL_i} \cdot 40$$

The $ZMP_{FB}$ is in the [-40, 40] range. CMAC 2 provides adjustments to the relative body position in order to achieve a value of $ZMP_{FB}$ close to zero. In other words, the output of CMAC 2 is used to achieve an equal overall distribution of force between the “toes” and the “heels” of the feet during the stepping motions. This has the effect of preventing the
biped from falling forward or backward. The neural net is trained during every execution of the control thread of the high level controller, using the $ZMP_{FB}$ as the training error signal. For negative values of the $ZMP_{FB}$, that is when there is more force on the “heels”, CMAC 2 is trained to help the biped position its hips further forward, and vice versa. The state vector of CMAC 2 has the following eight elements:

- desired step length;
- desired foot lift magnitude;
- relative position of the right foot with respect to the hip;
- relative position of the left foot with respect to the hip;
- measured front-back angle;
- measured right-left angle;
- difference between current and last measured front-back angle;
- difference between current and last measured right-left angle.

7.6.5 Posture Command Correction Using CMAC Neural Networks (CMAC 3 and CMAC 4)

As in the case of the static gait controller, the control architecture uses simplified kinematics to translate the posture commands into desired actuator angles. Due to this fact, and other inaccuracies in the model, commanded postures will not be implemented perfectly. This is a problem because the low level controller uses the difference between the target and measured lean angles as error signals for two modified PD controllers. CMAC 3 and CMAC 4 are used to learn adjustments to the desired right-left lean and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
front-back lean posture commands respectively, such that once these commands are implemented by the controller, the achieved angles match the desired angles better. CMACs 3 and 4 are the counterparts of CMACs 4 and 5 of the static walking controller.

CMAC 3 and CMAC 4 are trained during every execution of the control thread of the high level controller. For CMAC 3 the training error ($\text{error}_{\text{CMAC}3}$) is proportional to the difference between the target right-left angle from the previous control cycle ($\text{rlt}[1]$) and the current observed right-left lean angle ($\text{rl\_angle}$):

$$\text{error}_{\text{CMAC}3} = K \cdot (\text{rlt}[1] - \text{rl\_angle}).$$

By using the target right-left angle from the previous control cycle, the training error for CMAC 3 partially takes into account the delay in the implementation of desired joint positions. There is a delay of three high level control cycles between the time the high level controller calculates a set of desired joint positions, and the time the information is available to the high level control about these positions being implemented. This is illustrated in Figure 7.13. If a set of joint positions is available at the end of cycle 0, it will be written out to the low level controller by the end of cycle 1. The low level controller will implement the desired angles by the end of cycle 2, and they will be read back to the high level controller at the beginning of cycle 3.
cycle 0       cycle 1       cycle 2       cycle 3
calculate desired positions  send position to low level control  low level control implements desired positions  high level control receives feedback

Figure 7.13 Control delay

For CMAC 4 the training error ($error_{CMAC4}$) is proportional to the difference between the target front-back angle, which is always $9^\circ$, and current observed front-back lean angle ($fb\_angle$):

$$error_{CMAC4} = K \cdot (9^\circ - fb\_angle).$$

The state vector of CMAC 3 has the following nine elements:

- sideways speed during the last control cycle;
- difference between the values of the lift right foot and lift left foot posture parameters ($lrf - llf$);
- relative position of the right foot with respect to the hip;
- relative position of the left foot with respect to the hip;
- measured front-back angle;
- measured right-left angle;
- difference between current and last measured front-back angle;
- difference between current and last measured right-left angle;
- right-left target lean (rlt) posture parameter during the last control cycle.
The state vector of CMAC 4 has the following eight elements:

- desired front-back angle;
- desired foot lift magnitude;
- relative position of the right foot with respect to the hip;
- relative position of the left foot with respect to the hip;
- measured front-back angle;
- measured right-left angle;
- difference between current and last measured front-back angle;
- difference between current and last measured right-left angle;
- right-left target lean (rlt) posture parameter during the last control cycle.

Note that the right-left lean (rll) posture command is corrected using only CMAC 3. The PID correction used in the case of the static walking algorithm was not implemented for the unified controller. This design decision was based on the second secondary hypothesis (SH #2), which proposed that using only a CMAC to correct the rll posture command will result in satisfactory walking performance at higher stepping rates.

7.6.6 Simplified Kinematics

The simplified kinematics translates the commanded posture sequences into commanded actuator angles by assuming that there is no coupling between the frontal plane and the sagittal plane. As we saw in Section 6.3.7, this assumption does not hold too well,
however it allows the calculations to be greatly simplified. The resulting implemented postures do not match the commanded postures, but the errors are repeatable.

In the static walking controller the simplified kinematics block was implemented as part of the low level controller. In the unified control algorithm it became part of the high level control. The reason for moving the block to the high level control was to reduce the load of the low level hardware, which in turn allows future expansions of the low level software. The simplified kinematics block only needs to be executed with every execution of the high level control thread - this is when new posture commands are available. The low level software executes several times during one high level control cycle. Therefore, it is very useful for handling time-critical responses. A good example is the modified PD controller implemented in the low level software. Moving the simplified kinematics block to the high level control allows future expansion of the low level software with time-critical blocks.

7.6.7 Foot Contact Control Through Ankle Position Correction (CMAC 5)

The third secondary hypothesis (SH #3) proposes that providing good foot contact in double support phase is necessary for successful walking at variable speeds. The fourth secondary hypothesis (SH #4) says that good foot contact in double support phase can be achieved by correcting the positions of the ankle joints by CMAC neural network. Recall that, in the case of static walking, good foot contact was achieved with the help of an integral controller (see Section 6.4.4), however this controller was not stable at higher stepping rates.
CMAC 5 corrects the positions of the Right Ankle_Y, and the Left Ankle_Y joints in order to achieve better foot contact in the double support phase. The CMAC has two outputs, one for the Right Ankle_Y joint, and one for Left Ankle_Y joint. The state vector of CMAC 5 has the same eight elements as CMAC 2:

- desired step length;
- desired foot lift magnitude;
- relative position of the right foot with respect to the hip;
- relative position of the left foot with respect to the hip;
- measured front-back angle;
- measured right-left angle;
- difference between current and last measured front-back angle;
- difference between current and last measured right-left angle.

CMAC 5 is trained during every execution of the high level control thread. For different phases of the gait, the training for the two outputs of the CMAC is based on the error values given in Table 7.1.
The training of the output responsible for correcting the Right Ankle Y joint is based on the right-left center of force measured on the right foot ($RLCF_R$). The training of the output responsible for correcting the Left Ankle Y joint is based on the right-left center of force measured on the left foot ($RLCF_L$). Recall that $RLCF_R$ and $RLCF_L$ are defined in Section 6.4.4 as:

$$RLCF_R = \frac{FR_2 + FR_4 - FR_1 - FR_3}{\sum_{i=1}^{4} FR_i} \cdot 40,$$

and

$$RLCF_L = \frac{FL_2 + FL_4 - FL_1 - FL_3}{\sum_{i=1}^{4} FL_i} \cdot 40.$$  

While the biped is extending the left leg, it is leaning to the right, and preparing to lift the left foot. In single support phase the right foot will play the role of the supporting foot, therefore keeping the value of $RLCF_R$ in the current double support phase close to zero will prevent rocking when the foot is lifted. Conversely, when the biped is extending the
right leg the value of $RLCF_L$ has to be kept near zero. CMAC 5 is trained to keep the value of $RLCF_R$ at -5, and the value of $RLCF_L$ at 5. These offsets were added because without them CMAC 5 did not manage to keep the above values close to zero. Note that due to the offsets CMAC 5 tries to push the $RLCF$ of the given foot toward the other foot.

7.6.8 Overview of CMACs Used in the High Level Controller

The elements of the input vectors of the five CMACs are summarized in Table 7.2. Notice that all five CMACs take into account the position of the right and left foot with respect to the upper body, the front-back and the right-left lean angle of the biped, and the difference between the current and the last measured values of these angles. This is because these values contain important information about the instantaneous position of the biped and about its dynamics.

Another variable that is used by multiple CMACs (CMACs 1, 2, 4, and 5) is the desired foot lift magnitude. This variable influences the right-left stability of the biped, since changing the height of the lifted foot changes the moment around the ZMP in the frontal plane. Hence, it is part of the state space CMACs 1, 4 and 5. For short steps the influence of this variable on the moment around the ZMP in the sagittal plane is smaller, however it increases for longer steps. This is why it is also included in the input space of CMAC 2. The desired foot lift magnitude variable also influences step timing. The lifting and the lowering of the feet is done at a user-determined speed. Once this speed is set the length of time the lifting and lowering takes depends on the desired foot lift magnitude. Timing is important in both right-left and front-back balance control,
therefore the inclusion of this variable in the above CMACs is appropriate. Timing is especially important in the case of CMAC 1, which has a role in determining the *lift lean angle*. The value of the *lift lean angle* will determine the time the foot spends in the air (this is illustrated in Sections 2.4 and 7.1.1, for the case of a two-dimensional inverted pendulum model of a biped).

CMAC 3 deals with right-left stability in an indirect fashion, through adjusting the right-left lean posture parameter such that the right-left target lean matches the measured right-left lean. However the desired foot lift magnitude is not part of the state space of this CMAC. Instead, CMAC 3 relies on the value of the difference between the right foot lift and the left foot lift posture parameters. This in effect gives the CMAC information about where in the single support phase the biped is at any moment.
Table 7.2 CMAC input vector elements

CMAC 1 uses the value of the current sideways speed of the biped. This variable is very important because, for different sideways speeds, the magnitude of the sideways lean is different (see model in Sections 2.4 and 7.1.1). CMAC 3 uses the sideways speed of the previous control cycle. Again, this variable is very important because, depending on the value of the sideways speed in the previous cycle the dynamic properties of the biped will be different.
As in the case of static walking, a state of imbalance at a given time during walking generally results from incorrect postures at earlier times, rather than from an incorrect current posture. Thus, in the training of CMACs 2, 3, and 4 the general technique of temporal difference learning [38] is used to distribute the information from the delayed supervised learning over sequential time steps. In the case of determining the lift lean angle, the cause for not leaning the right amount is the last output of CMAC 1, thus temporal difference learning is not applicable. CMAC 5 corrects kinematics errors. Current errors are not due to incorrect angles at some previous time, but incorrect angles at the present moment. Therefore temporal difference learning was not applied in the case of CMAC 5.

7.7 Low Level Control

The outline of the low level controller is given in Figure 7.14. The low level controller receives desired joint angles and desired body angles, denoted as desired angles in Figure 7.14, from the high level controller. The desired joint angles, are corrected by the modified body angle PD control, in order to improve the walking performance. The resulting commanded angles are inputs to PID controllers that output voltages to the joint motors. The low level controller implemented for the unified walking controller is almost identical to the low level controller for the static walking. There are two differences between the two controllers. One is that the new controller receives desired angles from the high level controller, and does not have to translate desired posture commands into angles, like the static low level controller did (see Figure 6.14). The second difference is that the new controller does not implement foot contact control,
because in the unified walking algorithm foot contact control is implemented as part of the high level controller.

![Diagram of low level control outline](image)

**Figure 7.14 Low level control outline**

The low level control software implementation is very similar to the software implementation of the static low level software, described in Section 6.4.1. The difference is that the control interrupt service routine has a reduced load, namely it does not have to translate incoming posture commands, and it does not implement foot contact control. Due to the reduced load the control interrupt service routine can run every 4 ms, instead of every 5 ms, as in the case of the static walking controller. This translates into a 250 Hz control rate.

The block diagram of the low level control architecture is shown in Figure 7.15. Variables in angle brackets are sampled physical measurements, and variables in lower case letters represent results of control calculations.
From the control architecture block diagram we can see that the low level controller concentrates on two tasks. The first one is to provide lean angle reactive control, based on the measurements obtained from the virtual sensors. The second one is to implement the corrected actuator angles using PID control and pulse-width modulation (PWM). The low level controller performs both of these functions identically to the low level controller of the static walking algorithm (see Section 6.4.5 and Section 6.4.6). The reactive lean control block uses the outputs of the virtual sensors. The virtual sensors of the unified controller are implemented identically to the virtual sensors of the static walking controller (see Section 4.1.3).
8. RESULTS

In order to evaluate the performance of the new unified walking control algorithm several trials of three walking experiments were performed. This chapter presents the test procedures for the experiments, and the results collected during these walking experiments.

The three walking experiments were the following:

- the variable gait speed experiment,
- the walking velocity and stride length measurement experiment, and
- the reliability experiment.

The variable gait speed experiment was designed to test the ability of the biped to walk at different speeds, and to switch between these speeds in real time. It also tested the ability of the adaptive algorithm to improve its performance over time, and the motor drive signals. The gait speeds were picked in a wide-enough range to span static and dynamic walking. The walking velocity and stride length measurement experiment tested the forward progression velocity of walking the biped achieved at different parameter settings, and the corresponding stride length the biped executed. Finally, the reliability experiment tested the reliability of the unified controller for different gait speeds. Still shots of the biped walking are given in Appendix 3.
8.1 Variable Gait Speed Experiment

The goal of the variable gait speed experiment was to test the performance of the new controller for different gait speeds, and for cases when the speeds were changed on the fly. Trials of the experiment were conducted as follows. First, the biped's neural networks were cleared of all previous experiences. Then, the neural nets were trained for different values of the following four parameters:

- sideways leaning speed;
- stride length;
- foot lift height;
- foot lifting speed.

The biped started marching in place with the sideways leaning speed set to approximately 3.5 °/sec, and the stride length set to zero. The desired foot lift height was set to 2 cm, and the foot lifting speed was approximately 8.5 cm/sec. Note that the desired foot lift height and the foot lifting speed determine the duration of the single support phase. If we denote the single support phase duration as \( T_{SS} \), the desired foot lift as \( h \), and the lifting speed as \( v_l \), then ideally the following applies:

\[
T_{SS} = \frac{2 \cdot h}{v_l}.
\]

Recall from Section 2.4 that the longer \( T_{SS} \) is, the harder it is to control the biped. In the next five minutes the foot lift height was increased to 3 cm and then to 4 cm. Over approximately the following 90 minutes the above settings were varied in steps, one at a
time, to cover the ranges given in Table 8.1. The values of the step sizes that the parameters were changed by, and the number of different settings of each parameter that were tested, are also given in the table.

<table>
<thead>
<tr>
<th>parameter</th>
<th>low setting</th>
<th>high setting</th>
<th>setting step size</th>
<th>settings in range</th>
</tr>
</thead>
<tbody>
<tr>
<td>sideways leaning speed</td>
<td>3.6 °/sec</td>
<td>14.3 °/sec</td>
<td>1.5 °/sec</td>
<td>7</td>
</tr>
<tr>
<td>stride length</td>
<td>0 cm</td>
<td>11 cm</td>
<td>1 cm</td>
<td>12</td>
</tr>
<tr>
<td>foot lift height</td>
<td>2 cm</td>
<td>4 cm</td>
<td>1 cm</td>
<td>3</td>
</tr>
<tr>
<td>foot lifting speed</td>
<td>8.6 cm/sec</td>
<td>12.1 cm/sec</td>
<td>1.8 cm</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 8.1 Walking parameter settings

The values of the \textit{sideways leaning speed} and the \textit{foot lifting speed} could not be picked independently. The allowed pairs of values of these parameters are listed in Table 8.2.

<table>
<thead>
<tr>
<th>sideways leaning speed [°/sec]</th>
<th>foot lifting speed [cm/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6</td>
<td>8.6</td>
</tr>
<tr>
<td>5.4</td>
<td>8.6</td>
</tr>
<tr>
<td>7.1</td>
<td>8.6</td>
</tr>
<tr>
<td>8.9</td>
<td>10.4</td>
</tr>
<tr>
<td>10.7</td>
<td>12.1</td>
</tr>
<tr>
<td>12.5</td>
<td>12.1</td>
</tr>
<tr>
<td>14.3</td>
<td>12.1</td>
</tr>
</tbody>
</table>

Table 8.2 Allowed pairs of sideways leaning speed and foot lifting speed

172
A record of how the different settings were changed with time for one trial of the variable gait speed experiment is given in Appendix 4.

All data about the state of the biped collected during trials of the variable gait speed experiment were taken using a function of the high level control program. Data were logged after every execution of the high level control thread, that is every 28 ms. Desired values logged this way represent the complete set of values created by the high level digital controller during the logging period. The recorded measured values represent the sampled feedback about biped states received by the high level controller, since the low level software performs measurements every 4 ms.

In performing the variable gait speed experiment, we wanted to accomplish the following five goals:

- test the change in the control system's performance with training;
- test the biped's performance for different gait speeds;
- test the biped's performance for cases when the biped changes gait speeds on the fly;
- test the walking performance for gait speeds that the controller could not handle consistently;
- observe the motor drive signals for various allowed gait speeds.

The following sections show data that pertain to these testing goals.
8.1.1 Change in Performance with Training

In order to obtain the results shown in this section, the biped was first trained to walk with the settings given in Table 8.3. When the biped achieved stable walking with these settings, the foot lift height was changed from 3 cm to 4 cm. Data were logged immediately after the foot lift height was changed, that is at the beginning of training with the new setting, and after approximately 5 minutes of walking with the new setting.

Figure 8.1 and Figure 8.2 show the right-left target lean angle (rlt posture parameter), the measured right-left angle, and the difference between the two angles (error) at the beginning of training, and after 5 minutes of training respectively. Figure 8.3 and Figure 8.4 show the right-left and front-back ZMP (ZMP_{RL} and ZMP_{FB}) at the beginning of training, and after 5 minutes of training respectively.

<table>
<thead>
<tr>
<th>parameter</th>
<th>setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>sideways leaning speed</td>
<td>3.6 °/sec</td>
</tr>
<tr>
<td>stride length</td>
<td>0 cm</td>
</tr>
<tr>
<td>foot lift height</td>
<td>3 cm</td>
</tr>
<tr>
<td>foot lifting speed</td>
<td>8.6 cm/sec</td>
</tr>
</tbody>
</table>

Table 8.3 Settings before change
Figure 8.1 Right-left angles at the beginning of training

Figure 8.2 Right-left angles after 5 minutes of training

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 8.3 Right-left and front-back ZMP at the beginning of training

Figure 8.4 Right-left and front-back ZMP after 5 minutes of training

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
8.1.2 Walking Performance with Constant Gait Speed

The data presented in this section were logged after the biped was trained for about 90 minutes at various settings, to cover the ranges given in Table 8.1. The biped was then tested at several constant gait speeds. Table 8.4 shows the settings for the data presented in the figures of this section.

<table>
<thead>
<tr>
<th>parameter</th>
<th>settings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Figure 8.5, Figure 8.6, and Figure 8.7</td>
</tr>
<tr>
<td>sideways leaning speed [°/sec]</td>
<td>7.1</td>
</tr>
<tr>
<td>stride length [cm]</td>
<td>9</td>
</tr>
<tr>
<td>foot lift height [cm]</td>
<td>4</td>
</tr>
<tr>
<td>foot lifting speed [cm/sec]</td>
<td>8.6</td>
</tr>
</tbody>
</table>

Table 8.4 Parameter settings for Figures 8.5 through 8.9

Figure 8.5 shows the commanded right and left foot lift, and the measured right-left and front-back angles. Figure 8.6 shows the right-left and front-back ZMP. Figure 8.7 and Figure 8.8 show the commanded and measured right-left angles, and the absolute value of the difference between them (error). Figure 8.9 shows the commanded right and left foot lift and the $RLCF_{L}$. Note that the value of $RLCF_{L}$ reflects how good the contact is between the left foot and ground.
Figure 8.5 Right and left foot lift and the measured right-left and front-back angles for the sideways leaning speed of 7.1 °/sec

Figure 8.6 Right-left and front-back ZMP for the sideways leaning speed of 7.1 °/sec
Figure 8.7  Commanded and measured right-left angles, and the absolute value of the error for the sideways leaning speed of 7.1 °/sec

Figure 8.8  Commanded and measured right-left angles, and the absolute value of the error for the sideways leaning speed of 12.5 °/sec
8.1.3 Performance with Gait Speeds Changed on the Fly

The data presented in this section were logged after the biped was trained for about 90 minutes at various settings, to cover the ranges given in Table 8.1. The biped’s ability to change gait speeds on the fly was tested extensively. Three types of change in gait speed were tested:

- large changes in the sideways leaning speed,
- gradual changes of the sideways leaning speed (both increasing and decreasing), and
- “random” changes of the sideways leaning speed.

Figure 8.10, Figure 8.11, and Figure 8.12 show results for large changes in the sideways leaning speed: the sideways leaning speed was changed from 3.6 °/sec to 12.5 °/sec and
back to 3.6 °/sec. The data in the three figures was taken with the stride length set to 9 cm, and the step height to 4 cm. The foot lifting speed was varied with the sideways leaning speed according to the allowed pairs listed in Table 8.2. Figure 8.10 shows the right and left foot lift and the measured right-left and front-back angles. Figure 8.11 and Figure 8.12 show the right and left foot lift and the measured right-left ZMP ($ZMP_{RL}$), and the measured front-back ZMP ($ZMP_{FB}$) respectively.

Figure 8.13, Figure 8.14, and Figure 8.15 show results for walking with gradually decreasing sideways leaning speed: the sideways leaning speed was changed from 12.5 °/sec to 3.6 °/sec. Figure 8.16 shows results for walking with gradually increasing sideways leaning speed: the sideways leaning speed was changed from 3.6 °/sec to 12.5 °/sec. The data presented in these four figures was taken with the stride length set to 9 cm, and the step height to 4 cm. The foot lifting speed was varied with the sideways leaning speed according to the allowed pairs listed in Table 8.2. Figure 8.13 shows the absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. Figure 8.14 and Figure 8.15 show the absolute value of the commanded sideways leaning speed and the measured value of the right-left ZMP ($ZMP_{RL}$), and the measured front-back ZMP ($ZMP_{FB}$) respectively. Figure 8.16 shows the absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles.
Figure 8.17, Figure 8.18, and Figure 8.19 show results for walking with "random" changes in the sideways leaning speed: the sideways leaning speed was changed in the range of 3.6 °/sec to 12.5 °/sec. By "random" changes we mean that the operator did not deliberately follow a pattern in changing the value of the sideways leaning speed. The data presented in the three figures was taken with the stride length set to 9 cm, and the step height to 4 cm. The foot lifting speed was varied with the sideways leaning speed according to the allowed pairs listed in Table 8.2. Figure 8.17 shows the absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. Figure 8.18 and Figure 8.19 show the right and left foot lift and the measured value of the right-left ZMP ($ZMP_{RL}$), and the measured front-back ZMP ($ZMP_{FB}$) respectively.

![Graph showing foot lift and angles](image)

**Figure 8.10** Right and left foot lift and the measured right-left and front-back angles for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec.
Figure 8.11  Right and left foot lift and the measured right-left ZMP (ZMP_{RL}) for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec

Figure 8.12  Right and left foot lift and the measured front-back ZMP (ZMP_{FB}) for the sideways leaning speed changing from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec
Figure 8.13 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is decreasing.

Figure 8.14 Absolute value of the commanded sideways leaning speed and the measured value of the right-left ZMP (ZMP$_{RL}$). The sideways leaning speed is decreasing.
Figure 8.15 Absolute value of the commanded sideways leaning speed and the measured value of the front-back ZMP ($\text{ZMP}_{FB}$). The sideways leaning speed is decreasing.

Figure 8.16 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is increasing.
Figure 8.17 Absolute value of the commanded sideways leaning speed and the measured value of the right-left and front-back angles. The sideways leaning speed is changing "randomly".

Figure 8.18 Right and left foot lift and the measured right-left ZMP (ZMP_{RL}) for the sideways leaning speed changing randomly.
8.1.4 Performance at Speeds That Were Too High for the Controller

During the trials of the variable gait speed experiment it was found that the biped could walk with a sideways leaning speed of 14.3 °/sec, however the rate of failures was much higher than for lower sideways leaning speeds that were tested. The performance became much worse for sideways leaning speeds over 14.3 °/sec. This section presents data logged during walking with a sideways leaning speed of 14.3 °/sec, without failures. The data was taken with the stride length set to 9 cm, and the step height to 4 cm. The foot lifting speed was set to 12.1 cm/sec. The data presented in this section were logged after the biped was trained for about 90 minutes at various settings, to cover the ranges given in Table 8.1. Figure 8.20 shows the right and left foot lift and the measured value of the right-left and front-back angles. Figure 8.21 and Figure 8.22 show the right and left foot lift.
lift and the measured value of the right-left ZMP \( (ZMP_{RL}) \), and the measured front-back ZMP \( (ZMP_{FB}) \) respectively.

![Graph of foot lift and ZMP angles](image)

**Figure 8.20** Right and left foot lift and the measured right-left and front-back angles for sideways leaning speed of 14.3 °/sec

![Graph of foot lift and ZMP](image)

**Figure 8.21** Right and left foot lift and the measured right-left ZMP \( (ZMP_{RL}) \) for sideways leaning speed of 14.3 °/sec
8.1.5 Motor Drive Signal Observations During Walking

The data presented in this section were logged after the biped was trained for about 90 minutes at various settings, to cover the ranges given in Table 8.1. The drive signals of the motors were observed for constant and variable gait speeds. As explained in Section 4.1.4, the joint motors are driven using Pulse Width Modulation (PWM). The drive signals are the modulating signals in the PWM multiplied by +1 or -1 depending on the desired direction of motor rotation. Their values can be in the range of [-100, 100]. The value of the drive signal for a given motor can be found according to the following formula:

\[ \text{Drive Signal} = \begin{cases} 
+1 & \text{for forward rotation} \\
-1 & \text{for backward rotation} 
\end{cases} \times \text{Modulating Signal} \text{[V]} \]
\[ \text{drive}[i] = \frac{\text{direction}[i] \cdot t_{\text{pulse}}[i]}{T_{\text{PWM}}} \cdot 100, \]

where \( i \) is a number corresponding to one of the ten motors \( (i = 1, \ldots, 10) \), \( \text{drive}[i] \) is the drive signal for motor \( i \), \( \text{direction}[i] \) is the desired direction of rotation for motor \( i \) \( (\text{direction}[i] = +1 \) or \( \text{direction}[i] = -1 ) \), \( t_{\text{pulse}}[i] \) is the desired width of the driving pulse for motor \( i \) \( (0 \leq t_{\text{pulse}}[i] \leq T_{\text{PWM}}) \), and \( T_{\text{PWM}} \) is the PWM period \( (T_{\text{PWM}} = 0.8 \text{ ms}) \).

It was found that the drive signals required for stable walking depended primarily on the current gait speed and not on whether the gait speed was constant or variable. Therefore, the data presented in this section were logged while the sideways leaning speed was gradually increased from \( 3.6 \text{ °/sec} \) to \( 12.5 \text{ °/sec} \). This way the required drive signals could be evaluated for the whole range of the allowed gait speeds.

Figure 8.23, Figure 8.24, Figure 8.25, and Figure 8.26 show the values of the drive signals of the Left Ankle, Left Ankle-Y, Left Knee, and Left Hip and Hip-Y joints respectively. The figures also show the absolute value of the sideways leaning speed, which was gradually increased from \( 3.6 \text{ °/sec} \) to \( 12.5 \text{ °/sec} \). Note that the figures show the drive signals for all five motors of the left leg. The drive signals for the motors of the right leg were also observed, and they were found to be very similar to the ones presented here.
Figure 8.23 Left Ankle drive signal. The absolute value of the sideways leaning speed is gradually increasing.

Figure 8.24 Left Ankle-Y drive signal. The absolute value of the sideways leaning speed is gradually increasing.
Figure 8.25 *Left Knee drive signal.* The absolute value of the sideways leaning speed is gradually increasing.

Figure 8.26 *Left Hip and Hip-Y drive signal.* The absolute value of the sideways leaning speed is gradually increasing.
8.2 Walking Velocity and Stride Length Measurement Experiment

This experiment was designed to determine the average implemented walking velocity, and the corresponding average stride length of the biped, for one given set of parameters. For the experiment the biped traversed approximately 100 cm. The number of steps and the time needed to cover this distance were measured in three trials. Let us denote the three trials $T_1$, $T_2$, and $T_3$, the numbers of steps counted in each trial as $s_1$, $s_2$, and $s_3$, and the times measured as $t_1$, $t_2$, and $t_3$. The average stride length ($L$) was found using the following formula:

$$
L = 2 \cdot \frac{1}{3} \left( \frac{1}{s_1} + \frac{1}{s_2} + \frac{1}{s_3} \right) \cdot 100 \text{cm}.
$$

The average velocity ($\bar{v}$) was found using the following formula:

$$
\bar{v} = \frac{100 \text{cm}}{3} \left( \frac{1}{t_1} + \frac{1}{t_2} + \frac{1}{t_3} \right).
$$

The foot lift height was 4 cm, and the commanded stride length was 9 cm. The other two settings were varied to cover the allowed options listed in Table 8.2. The results of three trials, and the calculated values of $L$, and $\bar{v}$ are presented in Table 8.5.
Table 8.5 Average stride length (\( \bar{L} \)) and average walking velocity (\( \bar{v} \)) measurement results for three trials (T₁, T₂, and T₃).

### 8.3 Reliability Experiment

This experiment tested the reliability of the unified walking control algorithm for different settings of the desired sideways leaning speed. In the experiment the biped was commanded to walk and the number of steps taken by the biped, and the number of falls were observed. The biped continued to walk until it either ran out of room to walk, or it fell. A trial ended either with the biped falling, in which case the number of falls was one, or with the biped running out of room to walk, in which case the number of falls was zero. Due to the size of the laboratory the maximum distance the biped could cover was approximately 5-6 m, and it varied from trial to trial because the biped would not follow the same path during each trial. The biped could not follow the same path during
different trials because it had no way of detecting a change in direction of progress, and no mechanism to compensate for this change. Changes in the direction of progression were caused by the slipping of the feet on the floor.

For this experiment the foot lift height was 4 cm, and the commanded stride length was 9 cm. The other two settings were varied to cover the allowed options listed in Table 8.2. The experiment was repeated three or four times for each setting. The results of the trials, denoted as $T_1$, $T_2$, $T_3$, and $T_4$, are given in Table 8.6.

<table>
<thead>
<tr>
<th>max. sideways speed [°/sec]</th>
<th>steps taken</th>
<th>number of falls</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$T_1$</td>
<td>$T_2$</td>
</tr>
<tr>
<td>3.6</td>
<td>38</td>
<td>156</td>
</tr>
<tr>
<td>5.4</td>
<td>185</td>
<td>82</td>
</tr>
<tr>
<td>7.1</td>
<td>60</td>
<td>88</td>
</tr>
<tr>
<td>8.9</td>
<td>154</td>
<td>117</td>
</tr>
<tr>
<td>10.7</td>
<td>80</td>
<td>90</td>
</tr>
<tr>
<td>12.5</td>
<td>190</td>
<td>170</td>
</tr>
<tr>
<td>14.3</td>
<td>50</td>
<td>26</td>
</tr>
</tbody>
</table>

Table 8.6 Number of steps taken and corresponding number of falls
9. DISCUSSION

This chapter discusses the results of the three experiments presented in Chapter 0, and the relevance and validity of these experiments.

All results in Chapter 0, except for the results that illustrate the change in performance with training (Section 8.1.1) were logged when the biped was walking with 9 cm long strides and 4 cm foot lift height. The 9 cm stride length was chosen because in multiple trials of the experiments, with different sideways leaning speed settings, it was shown that commanded stride lengths of more than 9 cm did not consistently produce stable walking. For such stride lengths the biped ran into problems with stubbing the toe of the landing foot, which caused instability. The 4 cm foot lift height was chosen because lifting the feet 4 cm off the ground is clearly observable by looking at the biped walk, and the trajectory of the lifted foot can be completed in approximately 1 second. Making the foot lift height higher than 4 cm, and thus the foot lift period considerably longer than 1 second, created stability problems at higher gait speeds.

9.1 Discussion of Variable Gait Speed Experiment Results

The variable gait speed experiment tested the following:

- the change in the control system's performance with training;
• the biped’s performance for different gait speeds;
• the biped’s performance for cases when the biped changed gait speeds on the fly;
• the walking performance for a gait speed that the controller could not handle consistently;
• the motor drive signals for various allowed gait speeds.

The following sections discuss the data that pertain to the above five topics.

9.1.1 Discussion of Change in Performance with Training

One important desired feature of the new walking controller was that the algorithm should be adaptive. When the controller was implemented it had to be tested to prove that the algorithm did indeed improve its performance with time, specifically that the CMAC neural networks were being trained such that the walking was more stable.

Figure 8.1 and Figure 8.2 show the commanded and measured right-left angles of the biped at the beginning of training with a new setting of 4 cm for the foot lift height, and after approximately 5 minutes of training with the new setting, respectively. The figures also show the absolute value of the difference between the two angles, denoted as error. One sign of improved walking performance is a smoother curve of the measured right-left angle. If we inspect the measured right-left angle at the beginning of training we can see that the curve is not smooth. Local minima and maxima occur frequently, and they do not happen in the vicinity of the same values of the measured angle as time progresses. If we look at the right-left angle curve in Figure 8.2, we see that the it is much smoother. It
still has local minima and maxima, however they are always located around the values of the measured angle of $+\delta^\circ$ to $+9^\circ$ and $-7^\circ$ to $-9^\circ$. The location of these extremums is very predictable. The reason for this is that they are caused by the landing of a lifted foot. The controller has improved its performance such that the feet spend very nearly the same amount of time in the air every time they are lifted, and they predictably land when the value of the right-left angle is either around $+\delta^\circ$ to $+9^\circ$ (left foot landing) or around $-7^\circ$ to $-9^\circ$ (right foot landing).

The value of the error in Figure 8.2, which is the absolute value of the difference between the desired and the measured right-left angle, has decreased overall, compared to its value shown in Figure 8.1. At the beginning of training the peaks of the error were around $5^\circ$, while most of the time the error was between $2-3^\circ$. After 5 minutes of training the peaks were around $3^\circ$, while most of the time the value of the error was around $1-2^\circ$. We can conclude that, with respect to tracking the desired right-left lean angle, the adaptive controller has improved its performance over the 5 minutes. However, the measured lean angle still "leads" the target lean by approximately 15 control cycles, and the error is rather large ($2-3^\circ$) when the feet are in the air. Note that the feet are in the air around the peaks of the right-left angle curve.

Figure 8.3 and Figure 8.4 show the right-left and front-back ZMP ($ZMP_{RL}$ and $ZMP_{FB}$) at the beginning of training and after 5 minutes of training. The improvement in the smoothness of the $ZMP_{RL}$ after training is obvious after visual inspection of the curves.
Large fluctuations in the value of the $ZMP_{RL}$ are a sign of bouncing between the two feet - this happens a lot at the beginning of training, and hardly at all after 5 minutes of training. In fact, after 5 minutes of training bouncing occurs when a biped foot lands. This can be seen from the fact that the value of the $ZMP_{RL}$, after being +40 or -40, changes toward zero, and then suddenly jumps back to either +40 or -40, only to continue changing relatively smoothly toward zero.

The value of the $ZMP_{FB}$ is fairly low, both at the beginning of training, and after 5 minutes of training, indicating that the front-back balance is good throughout this time period. However, the $ZMP_{FB}$ makes frequent excursions past +10 at the beginning of training, while after training it does not go past +10, that is the controller improved the front-back balance.

### 9.1.2 Discussion of Walking Performance with Constant Gait Speed

In the process of creating a controller that would allow the biped to change the gait speed on the fly, it was important to first determine that the individual gait speed settings resulted in stable walking.

The results in Figures 8.5, 8.6, and 8.7 reflect stable walking at the sideways leaning speed of 7.1 °/sec. The right-left angle curve is smooth, and the absolute value of the difference between the target and measured right-left lean is mostly in the 2-3° range,
with the peak around $3^\circ$. The measured front-back angle is centered around $9^\circ$, with excursions of less than $\pm2.5^\circ$.

Let us observe the right and left foot lift curves in Figure 8.5. At around 1 second on the time axis, the right foot is commanded to lift at the constant speed of $8.6 \text{ cm/sec}$. This means that the right leg starts contracting such that, in the vertical direction, the right foot is moving toward the hip at $8.6 \text{ cm/sec}$. Recall from Section 7.6.3 that, when a foot is successfully lifted off the ground by the point it is commanded to be off the ground by $1.5 \text{ cm}$, it is lifted to the commanded height of $\text{lift}_{\text{preparation}} + \text{lift magnitude}$, where $\text{lift}_{\text{preparation}}$ is the value of the commanded lift at the moment the foot brakes contact with the ground, as detected by the pressure sensors and, in this case, $\text{lift magnitude} = 4 \text{ cm}$. Once the commanded foot lift reaches the value of $\text{lift}_{\text{preparation}} + 4 \text{ cm}$, it is commanded to descend, again at the speed of $8.6 \text{ cm/sec}$. If we wanted to be precise we could say that the speed of descent is $-8.6 \text{ cm/sec}$, to account for the direction of the motion. However, for the sake of simplicity we will omit the negative sign. The foot descends at this speed until it makes contact with the ground again. At this point the foot lift value is $\text{lift}_{\text{land}}$, and the descent is slowed down in order to reduce the shock the contact causes, as explained in Section 7.6.3. In Section 7.6.3 we also said that CMAC 1 is trained to help to keep the feet in the air such that:

$$\text{lift}_{\text{preparation}} + \text{lift}_{\text{land}} = 1.5 \text{ cm}.$$
If we now look at Figure 8.5 we see that the biped feet are successfully lifted every time. The value of \( \text{lift}_{\text{preparation}} \) is greater than zero, which can be seen from the fact that the peaks of the right and left foot lift curves are greater than 4 cm. The feet also land before the foot lift command is zero - this can be seen from the leveling-off of the foot lift curves. By visual inspection we can conclude that the value of \( \text{lift}_{\text{preparation}} + \text{lift}_{\text{land}} \) is in the range of approximately 0.5 cm to 2.5 cm. This performance can be considered satisfactory, because the biped walks in a stable manner.

The stability of the gait can also be seen from Figure 8.7. This figure shows that the absolute value of the difference between the commanded and measured right-left angle of the biped is usually between 0° and 2°, with occasional excursions to about 3°.

Figure 8.8 shows results logged when the biped was walking with a sideways leaning speed of 12.5 °/sec and with 9 cm long strides, and 4 cm foot lift height. The absolute value of the difference between the commanded and measured right-left angle of the biped was between 0° and 2°, with occasional excursions to about 3°, that is the walking was stable for this constant setting of the sideways leaning speed.

Figure 8.9 illustrates the success of the adaptive foot contact control scheme implemented in the high level controller (see Section 7.6.7). The biped was walking with a sideways leaning speed of 7.1 °/sec. The figure shows the left foot \( RLCF \) (\( RLCF_L \)), and the commanded right and left foot lift. The \( RLCF_L \) has to be close to zero before the left foot
is lifted, in order to avoid the biped rocking back onto the lifted right foot. The figure shows that this is achieved fairly successfully. During the extend left leg phase, which is in progress when the left foot lift curve has a smaller slope, the absolute value of the $RLCF_L$ decreases, and it is somewhere around 0-10 when the right foot is starting to lift. We can see from the fact that the right foot lift curve increases over 1.5 cm, that the lifting was successful. A similar result was found for the value of the $RLCF_R$.

In conclusion we can say that various constant gait speed settings resulted in stable walking. Let us now look at the performance of the controller with variable speed gaits.

**9.1.3 Discussion of Performance with Gait Speeds Changed on the Fly**

Figures 8.10 through 8.12 show results obtained when the sideways leaning speed was changed abruptly from 3.6 °/sec to 12.5 °/sec and back to 3.6 °/sec. Figure 8.10 shows the commanded right and left foot lift, and the measured right-left and front-back angles. The right-left angle curves are smooth during the double support phases (these are the parts of the graph where either both foot lifts are zero, or one is zero, and the other’s slope has decreased). The right-left angle curve has local maxima and minima which are lined up with the points where the foot lift curves change slope - these extremums are due to the shock of the lifted foot landing. The front-back angle is centered around 9°, however it makes excursions of approximately ±2.5°. Notice that the value of the front-back angle changes at a faster rate when the sideways leaning speed is higher.
Another interesting phenomenon that we can observe in Figure 8.10 is that, in single support phase, the front-back angle always dips while a foot is being lifted, and it increases while the foot is moving down. Notice that this phenomenon is pronounced whenever the biped is walking with non-zero length strides, and can be observed in other graphs in Chapter 8. The cause of this phenomenon is that when the foot is first lifted off the ground it is behind the biped's upper body. The loss of support creates an additional momentum around the ZMP on the supporting foot, and this momentum tends to pull the biped body backward. Consequently the biped's front-back angle decreases. Approximately when the lifted foot reaches the highest point of its trajectory it moves under the biped's upper body, and in the remaining part of the stride the momentum associated with it will tend to move the biped forward. As a result the biped's front-back angle increases. This is illustrated in Figure 9.1, where the commanded forward position of the right foot is plotted along with the right foot lift, and the front-back angle. Figure 9.2 shows the forces that create the additional torque around the ZMP on the supporting foot in single support phase.
**Figure 9.1** Front-back angle dip when the foot is lifted

**Figure 9.2** Torque around the ZMP created by the lifted foot

Figure 8.11 and Figure 8.12 show the ZMP$_{RL}$ and the ZMP$_{FB}$, along with the right and left foot lift commands, when the sideways leaning speed is abruptly changed from 3.6
°/sec to 12.5 °/sec and back to 3.6 °/sec. Both the $ZMP_{RL}$ and the $ZMP_{FB}$ indicate stable walking, although for slow sideways leaning the bouncing after the lifted foot hits the ground is quite pronounced. The $ZMP_{FB}$ also has fairly large peaks reaching values of approximately ±20°. However, comparing the behavior of the $ZMP_{RL}$ and the $ZMP_{FB}$ in Figures 8.11 and 8.12 to the results shown in Figure 8.6, which shows the $ZMP_{RL}$ and the $ZMP_{FB}$ for the sideways leaning speed of 7.1 °/sec, we can see that the performance during abrupt changes of the leaning speed is very similar to performance when the leaning speed is kept constant.

Figures 8.13, 8.14 and 8.15 show results for the case when the sideways leaning speed is gradually decreasing, while Figure 8.16 deals with the case when the sideways leaning speed is gradually increasing. All four figures reflect stable walking. Note that the absolute value of the sideways leaning speed is constant during double support phase, decreases to zero during the first part of a single support phase, and increases again during the second part of the single support phase. The absolute value of the leaning speed is decreasing while a foot is being lifted, and it increases again while the lifted foot is being lowered.

Figures 8.17, 8.18, and 8.19 show results for the case when the sideways leaning speed is changed “randomly”. By randomness we mean that the operator did not change the leaning speed according to some pre-planned pattern. The results in the figures show that the walking was stable. The right-left angle was smooth, the front-back angle was
centered around 9°, with excursions of less than ±2°. The $ZMP_{RL}$ indicates that the transition of the biped’s weight from one foot to the other was smooth. The $ZMP_{FB}$ was centered around zero, and for most of the time its value did not exceed ±10. The largest peaks of the value of the $ZMP_{FB}$ were approximately +20 and -15.

The results of Section 8.1.3 show that we can be satisfied with the performance of the controller for the cases when the gait speed changes on the fly. A very important point to make regarding the results shown in Section 8.1.3 is that the various sideways leaning speeds that were tested implemented both static and dynamic walking. The slow sideways leaning speeds resulted in static, or quasi static walking, while the faster speeds produced dynamic walking. Therefore, the controller was proven to be able to implement variable speed gaits, where the slower gaits implemented static walking, and the faster gaits implemented dynamic walking.

9.1.4 Discussion of Performance at Speeds That Were Too High for the Controller

As mentioned in Section 8.1.4, during the trials of the variable gait speed experiment it was found that the biped could walk with a sideways leaning speed of 14.3 °/sec, however the rate of failures was much higher than for lower sideways leaning speeds that were tested. The performance became much worse for sideways leaning speeds over 14.3 °/sec.
Figures 8.20, 8.21, and 8.22 show data logged when the biped was walking with a sideways leaning speed of $14.3^\circ/sec$, without falling. Figure 8.20 shows the right-left and front-back angles, along with the right and left foot lift commands. Notice that, after the left foot step that is finished around the 14 second mark, the front-back angle goes up to about $13^\circ$. At the same time, as shown in Figure 8.21, the $ZMP_{RL}$ stays at +40 after the foot was commanded to be on the ground ($left\ foot\ lift = 0$). From the two figures we conclude that the biped leaned forward by approximately $4^\circ$ more than the target front-back angle, and that the left foot did not make contact with the ground until about 0.5 seconds after it was commanded to. Going back to Figure 8.20 we can see that just before the 14 second mark the right-left angle reached $18^\circ$, which is approximately $3^\circ$ more than in the previous two left foot steps shown in the graph. From all the above information we can infer that the controller either made a mistake in estimating the necessary lean angle for this step, or the lean angle was not implemented correctly, and as a consequence the biped stayed in single support phase longer than desired. This in itself is not a reason for the biped to fall. However, Figure 9.3 shows that the biped rocked up on the right edge of the right foot. The figure shows that the value of the right-left component of the center of force on the right foot ($RLCF_R$) reaches +40 just before the 14 second mark, and stays at that value for approximately a second. This means that the biped was very close to falling (a record of visual observation states the biped did not actually fall). Since the unified controller relies on the position of the ZMP on the supporting foot to prevent falls during single support phase, once the value of the $RLCF_R$
reached $+40$ the biped was uncontrollable. The fact that it did not fall was only due to circumstances not under the control of the unified algorithm.

![Graph showing foot lift and RLCF](image)

**Figure 9.3** Left foot lift and $RLCF_R$ showing the biped rocked up on the right edge of the right foot

Figure 8.22 shows the right and left foot lift commands and the $ZMP_{FB}$ for the case when the sideways leaning speed is $14.3^\circ$/sec. We can see that the value of the $ZMP_{FB}$ reaches $-20$ several times. Looking at Figure 8.20 we can see that when the value of the $ZMP_{FB}$ is around $-20$ the front-back angle of the biped is in the vicinity of $5-6^\circ$, which is considerably less than the desired $9^\circ$. This data shows that the algorithm has problems controlling the front-back balance.

The reason for the poor performance of the unified control algorithm at the higher leaning speeds can be found by turning to the inverted pendulum model of the biped, that was

208

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
introduced in Section 2.4. If the sideways leaning speed is increased the operating curve of the inverted pendulum shifts to the left. For a fixed foot lift period the slope of the curve will increase, requiring more precise control. In the case of the UNH biped the precision of the control is limited by the fact that the control is performed at a fixed control rate. The physical biped is affected by disturbances caused by the uneven floor the biped was walking on, and non-repetitive disturbances caused by the biped’s mechanical structure, such as slop in the joints, gear stiction, shifting of the foot plates, and flexing of the links. As a result of the limited control precision some combinations of these disturbances cause falls.

9.1.5 Discussion of Motor Drive Signal Observations During Walking

Once the unified controller was implemented, it was important to analyze the data that showed how close the motors were to the limits of their performance capacity. Figure 8.23, Figure 8.24 and Figure 8.26 show that the drive signals for the Left Ankle, Ankle-Y, Hip, and Hip-Y joints rarely reach the $+100$ and $-100$ values. These motors are able to produce the torque outputs that are necessary to implement the desired angles of the given joints. However, the drive signal of the Left Hip-Y motor (Figure 8.26) is around $-80$ during a large portion of the gait cycle. This means that the motor is very close to the point of not being able to provide sufficient torque for stable walking.

In Figure 8.25 the value of the drive signal for the Left Knee motor reaches $+100$ a few seconds before the 45 second mark, and stays at this value for 2-3 seconds. This is the sign that the Left Knee motor was probably not able to produce the torque output that was
required to implement the desired Left Knee joint angles during that period. Figure 9.4 shows the Left Knee drive signal that was shown in Figure 8.25 and the corresponding right foot lift command. We can see that the 2-3 second period when the drive signal is +100 happens at about the same time when the right foot is lifted. This means that the Left Knee motor had problems outputting the required torque when most of the weight of the biped was supported by the left leg.

Figure 9.4 Left Knee drive signal and the right foot lift. The gait speed is gradually increasing.

Looking at Figure 8.25 we can also note that the drive signal reaches its maximum value while the biped is walking with a fast gait. During tests of the biped’s performance with constant sideways leaning speeds of 10.7 °/sec or more, it was found that the knee motors would sometimes have to be driven with maximum drive signals. The periods when maximum drive signals were required coincided with periods when most of the biped's
weight was supported by the leg of the given knee motor. For lower sideways leaning speeds the knee motors did not require maximum drive signals for extended periods of time. The other eight motors never required maximum drive signals (+100 or -100) for extended periods of time.

Why do the knee motors sometimes require maximum drive signals? At higher gait speeds the lifted foot often lands too early, before the knee had a chance to fully extend. With both feet on the ground the extension of the knee joint takes a relatively long time, and when the legs switch roles and the lifting leg becomes the supporting leg, the resulting angle of the knee joint requires the knee motor to output a larger torque. For slower gait speeds the feet land at more regular intervals, giving the knee joints enough time to extend. Consequently the knee joints do not need maximum drive signals.

9.2 Discussion of the Results of the Walking Velocity and Stride Length Measurement Experiment

From Table 8.5 we can see that the commanded stride length of 9 cm was not implemented - the average stride length of the gait was between 6.5 and 6.7 cm. There are two reasons for this. The first reason is that the inverse kinematics, which translates the desired posture commands into joint angle commands, is not completely accurate. The inverse kinematics does not take into account the coupling between motions in the frontal and the lateral planes, and it also does not model some characteristics of the mechanical structure that affect the implemented posture, such as slop in the gears, flexing of the links, etc. The second reason is that the mechanism that is responsible for
compensating the drifting of the foot that will be lifted tends to overcompensate the forward drift. As a result the foot that will be lifted slides back about 0.5-1 cm before it is lifted. Adding this distance to the distance covered by the lifted foot makes the average stride length approximately 7 cm.

Presenting the relationship between the average forward progression velocity of the gait and the maximum sideways leaning speed of the gait in graphical form makes it obvious that the relationship between the two is close to linear, as shown in Figure 9.5.

![Graph showing the relationship between average walking velocity and maximum sideways leaning speed.](image)

**Figure 9.5** The relationship between the maximum sideways leaning speed and the walking velocity is nearly linear.

Let us see where the nearly linear relationship shown in Figure 9.5 comes from. The forward progression velocity depends on the stride length and the stepping rate. From Table 8.5 we can see that the average stride length was virtually equal for all tested...
sideways speeds. Ideally, the stepping rate is linearly dependent on the maximum sideways leaning speed. This can be shown as follows. During one gait cycle, the biped has to lean from right to left, and back to right again (or from left to right, and back to left). Ideally, the right-left lean angles at the beginning and at the end of the cycle are the same, and so are the angles when the right foot is lifted and when it lands, as well as the angles when the left foot is lifted and when it lands. Let us denote the right-left lean angle at the beginning and at the end of the cycle as $\Theta_{\text{cycle}}$, the angles at the instances when the right foot is lifted and when it lands as $\Theta_{\text{right}}$, the angles at the instances when the right foot is lifted and when it lands as $\Theta_{\text{left}}$, and the maximum sideways leaning speed as $v_{\text{max}}$. Now the period of the gait cycle $T$, can be found as follows:

$$T = 2 \cdot \frac{\Theta_{\text{cycle}}}{v_{\text{max}}} + 2 \cdot \frac{\Theta_{\text{right}}}{v_{\text{max}}} + 2 \cdot \frac{\Theta_{\text{left}}}{v_{\text{max}}} = \frac{K}{v_{\text{max}}},$$

where $K$ is a constant. The second and the third term in the first equation are due to the fact that the sideways leaning speed is linearly reduced from $v_{\text{max}}$ to zero while the foot is being lifted, and it is linearly increased from zero to $v_{\text{max}}$ while the foot is being lowered. Since the stepping rate $f$, is the inverse of the cycle period, the stepping rate is linearly related to the maximum sideways leaning speed:

$$f = \frac{v_{\text{max}}}{K}.$$
In conclusion, the stride length is basically constant over different sideways leaning speeds, and there is ideally a linear relationship between the stepping rate and the leaning speed. Thus, the relationship between the forward progression velocity and the leaning speed is also close to linear.

### 9.3 Discussion of the Reliability Experiment Results

The goal of the reliability experiment was to determine how reliable the new unified control algorithm was. The obvious conclusion that we can draw from looking at the data in Table 8.6 is that the controller is reasonably reliable for sideways leaning speeds of up to 12.5°/sec.

In order to quantify the results obtained in this experiment we can calculate the empirical probability that the biped will fall during a step taken when the sideways leaning speed is \( v_i \). Let us denote this probability as \( p(v_i) \). If the number of steps for trial \( T_j \) is denoted as \( s_j \), and the number of falls during trial \( T_j \) is denoted as \( f_j \), \( p(v_i) \) can be calculated using the following formula:

\[
p(v_i) = \frac{\sum_{j=1}^{N} f_j}{\sum_{j=1}^{N} s_j}, \quad i = 1, 2, \ldots, 7
\]
where \( N_i \) is the number of trials performed with the sideways leaning speed \( v_i \). Note that seven different sideways leaning speeds were tested, thus the value of \( i \) can be one through seven. The probabilities found using the above formula are given in Table 9.1.

<table>
<thead>
<tr>
<th>( v_i [^\circ/sec] )</th>
<th>number of trials, ( j )</th>
<th>( \sum_{j=1}^{N_i} s_j )</th>
<th>( \sum_{j=1}^{N_i} f_j )</th>
<th>( p(V_i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6</td>
<td>3</td>
<td>374</td>
<td>1</td>
<td>0.0027</td>
</tr>
<tr>
<td>5.4</td>
<td>3</td>
<td>476</td>
<td>1</td>
<td>0.0021</td>
</tr>
<tr>
<td>7.1</td>
<td>4</td>
<td>268</td>
<td>4</td>
<td>0.0149</td>
</tr>
<tr>
<td>8.9</td>
<td>4</td>
<td>399</td>
<td>3</td>
<td>0.0075</td>
</tr>
<tr>
<td>10.7</td>
<td>3</td>
<td>340</td>
<td>2</td>
<td>0.0059</td>
</tr>
<tr>
<td>12.5</td>
<td>3</td>
<td>430</td>
<td>1</td>
<td>0.0023</td>
</tr>
<tr>
<td>14.3</td>
<td>4</td>
<td>133</td>
<td>4</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Table 9.1 Empirical probability \( p(V_i) \), that the biped will fall during a step for a given sideways leaning speed, \( v_i \)

The results of Table 9.1 are show in graphical form in Figure 9.6. From the figure we can see that the empirical probability of falls increases dramatically when the sideways leaning speed is greater than 12.5 \(^\circ/sec\).
Figure 9.6 Empirical probability \( p(v_i) \), that the biped will fall during a step for a given sideways leaning speed, \( v_i \).

9.4 Discussion of the Relevance and the Validity of the Test Procedures

It is important to determine whether the test procedures were relevant, that is if they really measured what needed to be measured in order to test the hypotheses of the research. It is also important to determine if the results were valid, that is if the test procedures measured the performance of the system correctly.

It is easy to see that the variable gait speed experiment was relevant to finding out whether the new control strategy was successful. The variable gait speed experiment tested the controller under all the conditions that it was supposed to perform under. It tested the performance of the controller when the gait speed was constant, and when the
gait speed was changing. Thus, the experiment directly or indirectly tested all the major and secondary hypotheses.

The walking velocity and stride length measurement experiment was relevant because it quantified the performance of the controller. Once the variable gait speed experiment showed that the new controller could implement static and dynamic gaits, it was important to gain quantitative data about the characteristics of these gaits.

During trials of the variable gait speed experiment, and the walking velocity and stride length measurement experiment the biped covered relatively short distances (not longer than 1-1.5 meters). The reliability experiment was relevant because it gave the author confidence that the results of the other two experiments were not results of "good luck", but rather represented repeatable performance.

The validity of the above experiments was proven by performing several trials for each experiment. The variable gait speed experiment was performed several times, always resulting in good walking performance. The walking velocity and stride length measurement experiment and the reliability experiment required performing several trials by design. The consistency of the results proves that the experiments were valid.
10. CONCLUSION

In order to evaluate the research presented in this dissertation let us first evaluate the major hypotheses of the research, that were presented in Chapter 2.

In Section 2.2 we said that the ZMP has to be within the stability region for both static and full foot contact dynamic walking. The second hypothesis proposed that this property of the two gaits can be used to create a unified biped walking control algorithm. According to the hypothesis the position of the ZMP can be measured, and the control algorithm should position the ZMP such that the biped is stable. Since the ZMP has to be in the same area for all gait speeds, the controller should not need to distinguish between static and dynamic walking. The measured value of the ZMP is used in two parts of the unified control algorithm. The adaptive control of front-back balance uses the front-back component of the ZMP as the training signal for CMAC 3. The CMAC outputs corrections to the commanded positions of the biped’s feet with respect to the hips. By moving the feet relative to the hips in the front-back direction the front-back balance is improved. The low level controller also uses the ZMP in single support phase to provide reactive control of the right-left and front-back lean angles. The controller uses the ZMP in both the high- and the low level controller without having to distinguish between static and dynamic walking. Therefore, the second hypothesis was proven to be correct.
The third hypothesis of the dissertation proposed that the new control algorithm can be made adaptive, and that it can improve its performance over time based on feedback from the biped sensors. The high level controller of the unified control algorithm has five CMAC neural networks. The output of the high level controller is a combination of pre-planned elements, and outputs of the CMAC neural networks. The neural networks are trained based on sensory feedback about the biped states. The performance of the system improves over time, as shown in Section 8.1.1. All this means that the third hypothesis was correct.

The fourth hypothesis of the research proposed that the unified bipedal walking controller should be based on adaptive closed-loop control. The control output should have a pre-planned, but adaptive, component, and a reactive component. The reactive component should compensate small errors of the controller and some disturbances. The low level controller of the unified control algorithm implements a modified PD controller, which reactively controls the right-left and front-back lean angles. The modified PD controller compares the lean angles, measured by the virtual body angle sensors, with the desired lean angles and uses the difference between them as the error signal. The modified PD controller is essential to the successful operation of the unified controller, which confirms the fourth hypothesis.

The main hypothesis of this research proposed that, if the static and dynamic walking gaits were analyzed and their important features were identified, one could base the design of a new walking control algorithm on the acquired knowledge. The new
algorithm would allow the biped to walk at a range of gait speeds, and switch from any
gait speed to any other gait speed within the range, at any given instant of time. The slow
velocities would be implementing static walking, and the faster velocities dynamic
walking. As explained in Section 7.3, the examination and comparison of the static and
dynamic gait characteristics led to the formulation of two major hypotheses, and seven
secondary hypotheses. These major and secondary hypotheses were the basis of the
design process that led to the implementation of the new unified biped walking control
algorithm. The new controller was tested through several trials of three experiments, and
it was found that, with the new controller, the biped can walk with variable gait speeds,
and change gait speeds on the fly. The slow gait speeds implement statically balanced
walking, while the faster speeds implement dynamically balanced walking. Therefore,
the main hypothesis of the research was correct.

In Section 1.2 the specific goals of this research were listed. Let us now list the
accomplishments of this research:

• The static and dynamic walking gaits were analyzed and their important features were
  identified.

• Based on the analysis of the two gaits a new biped walking controller was
  theoretically designed.

• The new control algorithm was implemented on the UNH biped. The new controller
  has the following characteristics:
- The biped can walk with forward progression velocities in the range of 21 cm/min to 72 cm/min. The goal was to implement walking in the 25 cm/min to 100 cm/min range. The implemented walking velocities are satisfactory, because the slow velocities implement static walking, and the faster ones dynamic walking.

- The gait speed can be changed on the fly. The biped’s gait speed can be changed from any allowed speed to any other allowed speed, at any instant of time. The allowed speeds correspond to sideways leaning speeds in the range of 3.6 °/sec to 12.5 °/sec (see Section 8.1). This achievement shows that the controller does not explicitly distinguish between static and dynamic walking.

- As explained in Chapter 9, in multiple trials of walking experiments, with different sideways leaning speed settings, it was shown that commanded stride lengths of more than 9 cm did not produce stable walking. With a commanded stride length of 9 cm the biped’s maximum stride length was approximately 9 cm. The biped’s average stride length with a commanded stride length of 9 cm was approximately 6.5 cm. The biped feet are 12 cm long, therefore the biped’s average stride length was approximately one half of the foot length.

- The biped can walk 5-6 m without falling. This result should not be interpreted as saying that the biped will be able to cover 5-6 meters in every walking trial. However, the system has consistently shown the ability to traverse these distances, as explained in Chapters 0 and 9.
In our evaluation of the research we can say that the goals listed in Section 1.2 were met. The research produced new scientific knowledge that can be used to create biped walkers that can walk with variable speed gaits.
11. FUTURE RESEARCH

There are many possible areas of future research connected to bipedal walking. Any future research could either use the existing biped hardware as a testbed, it could require adding new hardware to the existing biped, or it could incorporate creating a new physical biped. Let us first list a few research ideas that would use the existing UNH biped:

- The foot placement in the current controller is fixed. Future research could investigate ways to adaptively change the placement of the landing foot in order to improve right-left and front-back balance.

- The implemented front-back angle of the biped could be used to achieve better walking stability. In the existing controller the target front-back angle is fixed. In a future controller the target front-back angle could be actively controlled with the goal of improving the stability of the biped.

- Sometimes the biped stubs the toe of the landing foot. This reduces the implemented step length, and causes instability. A new controller could incorporate control of the position of the right and left "Ankle" joints in order to reduce the stubbing of the toes.

- A new controller could implement sideways walking. This maneuver would allow the biped more flexibility in tight spaces, and it would be another step towards creating a walker that could operate in human environments.
• The new unified biped walking control algorithm does not achieve gait speeds as high as the dynamic gait controller did. New input spaces could be tested for the controller's CMACs in order to push the limit of the gait speed higher.

Here are some research ideas that would incorporate either adding hardware to the existing biped or creating a new physical biped:

• The number of sensors could be increased on the UNH biped, and a new controller could be built that would use the increased amount of sensory information to improve the biped's performance. The new sensors could include more pressure sensors on the feet, and a gyroscope to measure rotation around the yaw axis.

• The execution rate of the control software could be increased by implementing new hardware for low level control and for feedback collection.

• A new biped could be built that would have arms. The walking performance could now be improved by using the arms to create additional momentums such that they improve walking stability.
BIBLIOGRAPHY


225

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


226

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


The block diagram of a virtual sensor is shown in Figure A.1.1. This appendix shows the equations and transfer functions implementing the two matched filters, and the integrator.

![Block Diagram of Virtual Sensor](image)

**Figure A.1.1 Virtual sensor**

The transfer function of the high-pass filter, denoted as $H_{HP}$ in Figure A.1.1, is selected to have zero gain at $0$ Hz in combination with the integrator, denoted as $H_I$ in Figure A.1.1. Since the transfer function of the integrator is:

$$H_I = \frac{dt}{1 - z^{-1}},$$

where $dt$ is the time step of integration, $H_{HP}$ should have one or more zeros at $z = 1$. The following transfer function satisfies this requirement:
where $p$ is the pole of the filter. We can multiply $H_{hp}$ and $H_I$ to get the transfer function of these two blocks, $H_{IHP}$:

$$H_{hp} = \frac{\left(\frac{1}{2} + p\right)^2 \cdot \left(1 - z^{-1}\right)^2}{\left(1 - p \cdot z^{-1}\right)^2},$$

For numerical reasons we can implement $H_{IHP}$ as two cascaded first order filters $H_{IHP1}$ and $H_{IHP2}$:

$$H_{hp} = \frac{dt}{1 - z^{-1}} \cdot \left(\frac{1}{2} + p\right)^2 \cdot \left(1 - z^{-1}\right)^2 = dt \cdot \left(\frac{1}{2} + p\right)^2 \cdot \frac{1 - z^{-1}}{(1 - p \cdot z^{-1})^2}$$

Let us denote the input of the $H_{IHP1}$ as $X$, the output of $H_{IHP1}$ as $W$, and the output of $H_{IHP2}$ as $Y$. Note that since $H_{IHP1}$ and $H_{IHP2}$ are transfer functions of cascaded filters, the input of $H_{IHP2}$ is $W$. Using this notation the equations implementing the above transfer functions are:
\[ W[0] = \frac{1+p}{2} \cdot X[0] - \frac{1+p}{2} \cdot X[1] + p \cdot W[1] \]

\[ Y[0] = dt \cdot \frac{1+p}{2} \cdot W[0] + p \cdot Y[1] . \]

In order to speed up the calculations necessary to solve the above equations we can pick the value of the pole \( p \), as a sum of powers of 2. This way, instead of executing multiply instructions, the code can execute shift instructions which are faster. Let \( p = 1 - 2^{-N} \).

Then the following applies:

\[ \frac{1+p}{2} = \frac{2-2^{-N}}{2} = 1-2^{-(N+1)}. \]

Using the above equation we can rewrite the \( H_{HP} \) equations as follows:

\[ W[0] = (1-2^{-(N+1)}) \cdot X[0] - (1-2^{-(N+1)}) \cdot X[1] + (1-2^{-N}) \cdot W[1] \]

\[ Y[0] = dt \cdot (1-2^{-(N+1)}) \cdot W[0] + (1-2^{-N}) \cdot Y[1] . \]

Since the two filters in the virtual sensor are matched, the transfer function of the low-pass filter has to be chosen such that the following applies:

\[ H_{LP} = 1 - H_{HP}. \]
Therefore, the following is true:

\[
H_{LP} = 1 - \left( \frac{1 + p}{2} \right)^2 \cdot \left( \frac{1 - z^{-1}}{1 - pz^{-1}} \right)^2
\]

\[
H_{LP} = \frac{(1 - pz^{-1})^2 - \left( \frac{1 + p}{2} \right)^2 \cdot (1 - z^{-1})^2}{(1 - pz^{-1})^2}
\]

\[
H_{LP} = \left( \frac{1 - p}{2} \right) \cdot \left( \frac{3 + p}{2} \right) \cdot \left( 1 - \frac{1 + 3p}{3 + p} z^{-1} \right).
\]

Again, for numerical reasons we can implement \( H_{LP} \) as two cascaded first order filters, \( H_{LP1} \) and \( H_{LP2} \):

\[
H_{LP} = H_{LP1} \cdot H_{LP2} = \left[ \frac{1 - p}{2} \cdot \frac{1 + z^{-1}}{1 - pz^{-1}} \right] \cdot \left[ \frac{3 + p}{2} \cdot \frac{1 - \frac{1 + 3p}{3 + p} z^{-1}}{1 - pz^{-1}} \right].
\]

Let us denote the input of the \( H_{LP1} \) as \( X \), the output of \( H_{LP1} \) as \( W \), and the output of \( H_{LP2} \) as \( Y \). Note that since \( H_{IHP1} \) and \( H_{IHP2} \) are transfer functions of cascaded filters, the input of \( H_{IHP2} \) is \( W \). Using this notation the equations implementing the above transfer functions are:
\[ W[0] = \frac{1-p}{2} \cdot X[0] + \frac{1-p}{2} \cdot X[1] + p \cdot W[1] \]

\[ Y[0] = \frac{3+p}{2} \cdot W[0] - \frac{1+3p}{2} \cdot W[1] + p \cdot Y[1] \cdot \]

Again, picking \( p = 1 - 2^{-N} \) will speed up the execution of the calculations for the above equations. The following is true:

\[
\frac{1-p}{2} = \frac{2^{-N}}{2} = 2^{-(N+1)}
\]

\[
\frac{3+p}{2} = \frac{4 - 2^{-N}}{2} = 2 - 2^{-(N+1)}
\]

\[
\frac{1+3p}{2} = \frac{4 - 3 \cdot 2^{-N}}{2} = 2 - 3 \cdot 2^{-(N+1)}
\]

Substituting the above equations we can rewrite the equations implementing the low-pass filter as follows:

\[ W[0] = 2^{-(N+1)} \cdot X[0] - 2^{-(N+1)} \cdot X[1] + (1 - 2^{-N}) \cdot W[1] \]

\[ Y[0] = (2 - 2^{-(N+1)}) \cdot W[0] - (2 - 3 \cdot 2^{-(N+1)}) \cdot W[1] + (1 - 2^{-N}) \cdot Y[1] \]
APPENDIX 2

The biped leans to one side by extending the opposite leg. During this process the Ankle-Y joint of the leg that will become the supporting leg turns by \( \theta \) degrees in the plain of the joint. Figure A.2.1 illustrates the motion of the link connecting the ankle and the knee joints of the leg that will become the supporting leg. The OA line represents this link when the biped is standing upright and the OB line represents the link when the biped is leaning sideways. The OC line represents the orientation of the axis of the biped when the biped is leaning sideways. The angle denoted \( \gamma_m \) is the angle that the right-left virtual sensor will measure. As a side-effect of leaning sideways the biped will also rotate around the z-axis by \( \beta \) degrees. The angle between the link connecting the Ankle-Y and the knee joints and the ground is denoted as \( \alpha \). This angle is constant during leaning sideways.

The following is true:

\[
\beta = \tan^{-1}\left(\frac{OB \cdot \sin \theta}{OB \cdot \cos \theta \cdot \cos \alpha}\right) = \tan^{-1}\left(\frac{ tg \theta }{ \cos \alpha }\right) \\
\tan \theta = \tan \beta \cdot \cos \alpha .
\]

(A.2.1)
Let us turn our attention to the measured lean angle, \( \gamma_m \). The following applies:

\[
\begin{align*}
tg_{\gamma_m} &= \frac{y'_i}{z_i} \\
tg_{\gamma} &= \frac{y_1}{z_1} \\
y_1 &= y'_i \cdot \cos \beta \\
tg_{\gamma} &= \frac{y'_i \cdot \cos \beta}{z_1} = tg_{\gamma_m} \cdot \cos \beta.
\end{align*}
\] (A.2.2)
Note that $\gamma$ is an auxiliary angle introduced for the sake of convenience. Through this angle we can find a relationship between the measured angle $\gamma_m$, and the rotation around the z-axis, $\beta$:

$$z_1 = OA \cdot \sin \alpha$$

$$tg\theta = \frac{y_1}{OA - (OA - x_1 \cdot \cos \beta)}$$

$$\cos \beta \approx 0 \Rightarrow tg\theta \approx \frac{y_1}{OA} \Rightarrow OA \approx \frac{y_1}{tg\theta} \quad (A.2.3)$$

$$z_1 = \frac{y_1 \cdot \sin \alpha}{tg\theta}$$

$$tg\gamma = \frac{y_1}{z_1} = \frac{tg\theta}{\sin \alpha}.$$  

The approximation in equations A.2.3 assumes that the angle of rotation around the z-axis $\beta$, is small. Combining the results from equations A.2.1, A.2.2, and A.2.3 we have the following:

$$tg\gamma = \frac{tg\theta}{\sin \alpha} = \frac{tg\beta \cdot \cos \alpha}{\sin \alpha} = \frac{tg\beta}{tg\alpha} = tg\gamma_m \cdot \cos \beta$$

$$tg\beta = tg\gamma_m \cdot tg\alpha \cdot \cos \beta \quad (A.2.4)$$

$$\beta \approx 0 \Rightarrow \beta = tg^{-1}(tg\gamma_m \cdot tg\alpha)$$
Figure A.2.2 shows the position of the feet before and after the biped leans sideways. The angle of rotation around the z-axis is denoted as $\beta$, and the drift of the foot due to this rotation is denoted as $d$. The distance between the feet is $l$.

Using the result derived in equations A.2.4 we can find the approximate value of the drift to be:

$$d \approx 2 \cdot \sin \left( \frac{\beta}{2} \right) \cdot l \approx 2 \cdot \sin \left( \frac{t g^{-1} (t g_{m} \cdot t g_{a})}{2} \right) \cdot l.$$
APPENDIX 3

Figure A.3.1 shows still shots of the biped walking (see next page).
Figure A.3.1 The UNH biped walking
Appendix 4

Table A.4.1 gives a record of how the different settings were changed with time for one trial of the variable gait speed experiment.

<table>
<thead>
<tr>
<th>training time</th>
<th>sideways leaning speed [°/sec]</th>
<th>foot lifting speed [cm/s]</th>
<th>stride length [cm]</th>
<th>foot lift height [cm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>20</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>20</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>20</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>25</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>30</td>
<td>30</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>35</td>
<td>35</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>20</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>20</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>20</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>15</td>
<td>20</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>20</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>25</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>30</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
Table A.4.1 Record of how the different settings were changed with time for one trial of the variable gait speed experiment.

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>20</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>20</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>25</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td>30</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>20</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>15</td>
<td>20</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>20</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>25</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>30</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>20</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>20</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>20</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td>25</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>30</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>20</td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>