UMONS

POLYTECH MONS

UNIVERSITE DE MONS
FACULTE POLYTECHNIQUE
Service de Mécanique Rationnelle, Dynamique et Vibrations
Boulevard Dolez, 31 | B - 7000 Mons (Belgium)

Thèse de doctorat déposée et acceptée en vue de l'obtention du grade de
Docteur en Sciences de l'Ingenieur
par

Quentin Bombled

Modeling and Control of Six-Legged Robots
Application to AMRU5

Membres du Jury :

Prof. Yvan Baudoin — ERM
Prof. Olivier Brüls — ULg
Prof. Pierre Dehombreux — UMONS (secrétaire)
Prof. Pascal Drazetic — UVHC
Prof. Paul Fisette — UCL
Prof. Dirk Lefeber — VUB
Prof. André Preumont — ULB
Prof. Marcel Remy — UMONS (président)
Prof. Olivier Verlinden — UMONS (promoteur)

May 2011

To my family
Aline, Eliott and Maureen

Acknowledgments

This thesis represents five years of work, during which several people have contributed to its success.

First of all, I want to thank Prof. Conti for welcoming me in 2005 in the Department of Theoretical Mechanics, Dynamics and Vibrations at the University of Mons.

I also want to warmly thank my supervisor Prof. Olivier Verlinden, for his guidance, implication in the work, permanent good humor and his careful listening.

Thank you to the Royal Military Academy, and especially to Prof. Y. Baudoin for the loan of the AMRU5.

Thank you Michel, for your precious help in the hardware architecture and the real-time control of the robot.

Thank you Denis, for your help in electronics. Without you the first leg motions would have been more difficult, certainly with more damage.

Thank you Cedric and Barthelemy, for your help in C/C++ programming.

Thank you Georges and David, for your experience and your help in the teaching tasks.

Thank you Regis and Kevin for your excellent work in the improvement of the mechanical and electrical design of the robot. Your work makes the use of the robot more comfortable.

Thank you to all my colleagues for their good mood; you all make the working environment a pleasure.

Finally, thank you to you, Aline, who gives me so much time to work and happily takes care of the daily life.

Abstract

Walking machines have been studied for more than thirty years. They provide a more important mobility potential than their wheeled counterparts, but require more complex control algorithms. In this thesis, we contribute to their study, by modeling the AMRU5 robot and implementing gait algorithms to walk on an uneven ground. The work is divided into two main axes.

The first contribution to the study of legged vehicles lies in the development of a comprehensive dynamic model of the whole robot. The mechanical parts of the robot are first modeled by means of a multibody approach based on the minimal coordinates. This model is completed by the DC actuators, controllers and friction in the joints. An important part of this work concerns friction: effectively, the transmissions of walking robots are high-geared, and composed of numerous elements. In this context, we tried to reproduce the irreversibility behavior of the vertical joint with the LuGre friction model. The modeling itself is not new: on the contrary, the validation of the model through comparisons of actuators currents and voltages is quite novel, anyway for a complete robot actuated by DC motors and walking in real conditions. Power expenditure is also considered, as an indicator of the degree of realism of the simulation. It is shown that friction is responsible for approximately 80 % of the power expenditure, and that the dynamic model is able to reproduce quite well this result.

The second contribution to the study of walking machines is the development of gait algorithms. In particular, the free gait algorithm we developed allows the robot to adapt to the terrain profile, while keeping the heading velocity imposed by the user. Dynamic simulations are again used to prove the efficiency of the algorithm. In parallel with simulations, a real implementation is performed. The parallel approach leads to the development of a "realistic" foot profile, which includes transitions between the swing and the support phases of the leg. These transitions are mandatory on the real robot, to avoid severe damages.

In parallel with simulations, gait algorithms have been implemented on the real robot. An important part of this work concerns the complete design of the control architecture.

Contents

Acknowledgments **iii**

Abstract **v**

Contents **vii**

List of figures **xi**

List of tables **xv**

Notations and symbols **xvii**

1 Introduction **1**

1.1 Context and contributions 1

1.2 Outline of the work 4

2 Legged robots **5**

2.1 A little History 6

2.1.1 From 19th Century to 1970 6

2.1.2 From 1970 to 1980 7

2.2 Since 1980 across the world... 8

2.2.1 United States of America 8

2.2.2 Canada 10

2.2.3 Japan 12

2.2.4 Europe 12

2.3 AMRU5 among these legged machines 14

2.3.1 Mechanical overview 14

2.3.2 Control architecture 18

3 Modeling of six-Legged robots 27
3.1 State of the art 27
3.2 Modeling issues 30
3.3 The EasyDyn framework 31
3.3.1 General overview 31
3.3.2 The sim module 32
3.3.3 The mbs module 36
3.3.4 Practical use 41
3.4 Application to AMRU5 44
3.4.1 Kinematic description 44
3.4.2 Ground contact forces 48
3.4.3 Joint friction 50
3.4.4 Actuation 56
3.5 Summary of the modeling 60

4 Control design 61
4.1 Leg model 61
4.2 Inductance simplification 62
4.3 Coupling interactions 63
4.3.1 Mass matrix 63
4.3.2 Damping matrix 63
4.3.3 Stiffness matrix 65
4.3.4 Decoupling in 3 SISO loops 65
4.4 Controller synthesis 67
4.4.1 Open-loop transfer functions 67
4.4.2 The choice of PI controllers 67
4.4.3 Saturation as bandwidth limitation 67
4.4.4 Controller gains 68
4.4.5 Digital implementation and antiwindup 68
4.5 Simplification of the modeling 69

5 Friction identification 71
5.1 Preliminary assumptions 71
5.2 Steady-state curve identification for leg without load 73
5.2.1 Method 73
5.2.2 Measurements 74
5.2.3 Friction/velocities curves 77
5.3 Vertical joints 81
5.4 Horizontal joints 85
5.5 Remarks about previous work 88
5.6 Obstacle detection with current sensing 89
5.6.1 Principle 89
5.6.2 Torque estimation mechanism 90

5.6.3 Ground detection 92
5.7 Summary 96

6 Validation of the AMRU5 model 97
6.1 Discussion about friction models and parameters for the simulations 97
6.2 Simulation of the vertical irreversibility 100
6.3 Test case 100
6.4 Direct transposition of friction identification 103
6.4.1 Rotational joints 105
6.4.2 Vertical joints 109
6.4.3 Horizontal joints 114
6.4.4 Horizontal joint with variation of friction parameters 118
6.4.5 Specific Resistance 122
6.5 Extensive simulations with averaged friction 125
6.6 Summary 127

7 Gait control 129
7.1 Definitions 130
7.1.1 Typical parameters in legged locomotion 130
7.1.2 Classes of gaits 132
7.2 State of the art in gait generation 134
7.2.1 Bio-inspired approaches 134
7.2.2 Hybrid approaches 137
7.2.3 Engineering approaches 139
7.2.4 Gait generation principle for AMRU5 141
7.3 Kinematic control of a leg 143
7.4 Support phase 145
7.5 Swing phase 148
7.5.1 Periodic gait with constant straightforward velocity 148
7.5.2 Omnidirectional periodic gait 149
7.5.3 Results of omnidirectional motion 150
7.6 Enhancement of the periodic gait 156
7.7 Free gait 159
7.7.1 Posture control 160
7.7.2 Correction of global velocities 161
7.7.3 Swing phase for free gait 162
7.7.4 Free gait complete flowchart 165
7.7.5 Results of simulations 167
7.8 Summary 175

8 Conclusions and prospects 177

A Definition of elementary homogeneous transformation matrices 181

B Detailed description of AMRU5 **183**

B.1 Mechanics 183

B.2 Electronics 186

C Friction measurements for unloaded legs **189**

Bibliography **197**

List of Figures

2.1 Early legged robots . . . 6
2.2 Remarkable quadrupeds in the late sixties . . . 7
2.3 Remarkable hexapods in the late seventies . . . 8
2.4 ASV [CYBE] (1985) . . . 8
2.5 Legged robots that balance . . . 9
2.6 Robots with subsumption control architecture . . . 9
2.7 BigDog [BOST] (2008) . . . 10
2.8 *Ajax* [CRWU] (2005) . . . 10
2.9 CMU hexapods . . . 11
2.10 McGill legged robots . . . 11
2.11 Tokyo Institute of Technology quadrupeds . . . 12
2.12 Legged machines in Germany . . . 13
2.13 Legged machines from Europe . . . 14
2.14 AMRU5 hexapod . . . 15
2.15 Leg overview . . . 15
2.16 Focus on AMRU5 transmissions . . . 17
2.17 Global communication scheme . . . 19
2.18 Overview of the PIC-based slave board . . . 20
2.19 Timing diagram of the slave . . . 21
2.20 UDP data encapsulation [KURO08] . . . 22
2.21 Communication between master and slaves . . . 24
2.22 Timing for receptions of the slave data by the master . . . 24

3.1 `EasyDyn` framework . . . 31
3.2 `EasyAnim` window with the AMRU5 hexapod . . . 32
3.3 Structure of the `sim` module . . . 34
3.4 Homogeneous transformation matrices between two frames . . . 37
3.5 Structure of the `mbs` module and interactions with `sim` . . . 40
3.6 Slider-scrank mechanism . . . 42
3.7 Kinematic description of a leg . . . 45

3.8 Kinematic study of the pantograph 45
3.9 Penetration between two bodies [VERL07] 48
3.10 Friction model for the ground contact 49
3.11 General Kinetic Friction model 51
3.12 Steady-state characteristic of friction for different lubrications (adapted from [ARMS94]) 51
3.13 Presliding regime 52
3.14 Bristles deflection 53
3.15 Coupling between electrical part of the motor and multibody model 57
3.16 Global overview of the modeling for one DOF 60

4.1 Mass matrix contributions with respect to q_v and q_h 64
4.2 Damping matrix contributions with respect to q_v and q_h 64
4.3 Tangent stiffness matrix contributions with respect to q_v and q_h 65
4.4 Position control loop for one joint 65

5.1 Current measurements for leg 0 75
5.2 Current measurements for leg 2 76
5.3 Friction model for rotational joints (unloaded legs) 79
5.4 Friction model for vertical joints (unloaded legs) 79
5.5 Friction model for horizontal joints (unloaded legs) 79
5.6 Loading the leg with 10 kg for stiction identification 82
5.7 Stiction identification (leg 0) 83
5.8 Flexbility in horizontal transmission 85
5.9 Influence of foot load on current in the horizontal actuators 85
5.10 Measured evolution of dry friction force versus foot load (horizontal joint) . 86
5.11 Leg and joint spaces 90
5.12 Model-based estimation of the joint torques 91
5.13 Torque estimation at joint q_r 92
5.14 Torque estimation at joint q_v 92
5.15 Torque estimation at joint q_h 93
5.16 Experimental setup for ground detection 93
5.17 Comparison of estimated and real ground contact force for leg 0 94

6.1 Simulation of AMRU5 no powered 101
6.2 Internal variables of the LuGre model (leg 4) 102
6.3 Illustration of the test case for model validation 103
6.4 Results for leg 0 (rotational joint) 106
6.5 Results for leg 2 (rotational joint) 107
6.6 Instantaneous power consumption for the rotational actuator 108
6.7 Results for leg 0 (vertical joint) 111
6.8 Results for leg 2 (vertical joint) 112
6.9 Instantaneous power consumption for the vertical actuator 113
6.10 Results for leg 0 (horizontal joint) 115

6.11 Results for leg 2 (horizontal joint) 116
6.12 Instantaneous power consumption for the horizontal actuator 117
6.13 Results for leg 0, horizontal joint (with load influence) 119
6.14 Results for leg 2, horizontal joint (with load influence) 119
6.15 Instantaneous power consumption for the horizontal actuator - load influence120
6.16 Bad estimation of friction/load influence for leg 4 121
6.17 Bad estimation of friction/load influence for leg 5 121
6.18 Gabrielli-Von Karman diagram (adapted from [GREG97]) 123
6.19 Energetic indicator for different friction models 124
6.20 Specific resistance for different friction models 126

7.1 Basic leg step cycle (main body point of view) 130
7.2 Support pattern, configuration polygon and stability margin 131
7.3 Workspace defined for AMRU5 132
7.4 Tripod gait pattern 133
7.5 Forward periodic gait patterns with $\beta = 3/4$ (normalized step cycle time) . 134
7.6 Cruse's et al. researches on stick insect 135
7.7 Pearson's neural circuit [FERR95] 136
7.8 Albiez behaviors network [ALBI03a] 138
7.9 Reactive free gait (adapted from [CELA98]) 138
7.10 Inverse kinematics algorithm 143
7.11 Determination of the foot motion from high level commands 145
7.12 Basic cycle of a foot 148
7.13 Example of paths requiring an omnidirectional ability 151
7.14 Body motion for a back and forth motion 151
7.15 Foot trajectory adaptation for a heading velocity reversal (foot 1) 152
7.16 XY references and tracking error magnitude vs. time (straight lines) 152
7.17 Foot 0 trajectory adaptation in case of a motion change of 90 degrees . . . 153
7.18 Tracking error magnitude with/without navigation system (straight lines) . 154
7.19 Body trajectory in the ground plane (spline) 154
7.20 Tracking error with/without navigation system (spline) 155
7.21 Foot zero stroke and swing adaptation for the spline motion 155
7.22 Enhanced cycle of the foot 156
7.23 Detail of the enhanced leg cycle for legs 2 and 3 157
7.24 Tracking of a straight line on soft ground 159
7.25 Posture adaptation : 2-D case 160
7.26 Correction of global velocities 161
7.27 Illustration of rules 2 and 3 163
7.28 Research of target point in swing phase 164
7.29 Free gait algorithm (starting with deadlock condition) 166
7.30 Example of paths requiring a free gait 168
7.31 Screenshots of the animation (case 1) 169
7.32 X reference and tracking error vs. time (case 1) 169

7.33 Body height vs. X coordinate, for different body clearances (case 1) 170
7.34 Evolution of the roll and pitch vs. X coordinate (case 1) 171
7.35 XY references and tracking error magnitudes vs. time (case 2) 171
7.36 Evolution of the body height vs. X coordinate, for different body clearances (case 2) 172
7.37 Evolution of the roll and pitch vs. X coordinate (case 2) 172
7.38 XY references and tracking error magnitudes vs. time (case 3) 173
7.39 Yaw reference and tracking error vs. time (case 3) 174
7.40 Evolution of the body height vs. X coordinate, for different body clearances (case 3) 174
7.41 Evolution of the roll and pitch vs. X coordinate (case 3) 175

B.1 Geometrical configuration of the leg 186
B.2 Slave board organization 187
B.3 LEM acquisition schematics 187

C.1 Current measurements for leg 0 190
C.2 Current measurements for leg 1 191
C.3 Current measurements for leg 2 192
C.4 Current measurements for leg 3 193
C.5 Current measurements for leg 4 194
C.6 Current measurements for leg 5 195

List of Tables

2.1 Relevant geometrical characteristics of the pantograph 16

3.1 Reduction ratios of transmission elements 44
3.2 Simulation times for different kinematic improvements 48
3.3 Parameters of ground contact interaction 49
3.4 Motors parameters 57
3.5 Summary of the modeling in terms of bodies and degrees of freedom 60

4.1 Controller gains and closed-loop performances 68

5.1 Summary of the friction parameters identification 73
5.2 Velocities setup for steady-state curve identification 74
5.3 Dry and viscous parameters for the 18 joints (unloaded leg) 80
5.4 Stiction parameters for the 6 vertical joints (N), for different loading cases 84
5.5 Polynomial coefficients expressing the friction/load dependency 87
5.6 Profile of each joint for joint torque estimation 91
5.7 Measurements with U9B force transducer 95

6.1 LuGre parameters 99
6.2 Test case for the validation 100
6.3 Averaged power for the rotational actuators (V_x=1.5 cm/s) 108
6.4 Averaged power for the vertical actuators (V_x=1.5 cm/s) 114
6.5 Averaged power for the horizontal actuators (V_x=1.5 cm/s) 117
6.6 Improvement of power consumption for the horizontal joints (V_x=1.5 cm/s) 122
6.7 Mean and standard deviation of the friction parameters used in the Monte Carlo method 126
6.8 Mean and standard deviation for specific energy 127

B.1 Geometrical data [m] 184
B.2 Inertial data (mass in [kg] and inertia in [kg.m^2]) 185

Notations and symbols

Symbols

(x_0, y_0, z_0)	global coordinates system	
(x_b, y_b, z_b)	robot body coordinates system	
(x_l, y_l, z_l)	leg coordinates system	
t	time	[s]
$\boldsymbol{q}$	vector of configuration parameters	[rad] or [m]
$\dot{\boldsymbol{q}}$	first time derivative of $\boldsymbol{q}$	[rad/s] or [m/s]
$\ddot{\boldsymbol{q}}$	second time derivative of $\boldsymbol{q}$	[rad/s^2] or [m/s^2]
$\boldsymbol{\tau}$	vector of joint generalized forces	[N] or [Nm]
$\boldsymbol{F}$	force in the global coordinates system	[N]
(q_r, q_v, q_h)	rotational, vertical and horizontal leg joint coordinates	[rad] or [m]
(τ_r, τ_v, τ_h)	leg joint generalized forces related to (q_r, q_v, q_h)	[Nm] or [N]
(i_r, i_v, i_h)	motor currents related to (q_r, q_v, q_h)	[A]
(u_r, u_v, u_h)	motor voltages related to (q_r, q_v, q_h)	[V]
(n_r, n_v, n_h)	reduction ratios related to (q_r, q_v, q_h)	[rad/rad] or [rad/m]
Δt	control time slice	[s]
h	time step of the numerical integration	[s]
$\boldsymbol{T}_{i,j}$	homogeneous transformation matrix between frame i and frame j	
$\boldsymbol{T}^{R_i}$	homogeneous transformation matrix (pure rotation) around axis i	
$\boldsymbol{T}^{d}$	homogeneous transformation matrix (pure displacement)	
$\boldsymbol{R}_{i,j}$	rotation tensor between frame i and frame j	
$\boldsymbol{p}_{P/i}$	position vector of point P with respect to the frame i center of gravity	[m]

$\boldsymbol{d}_{i,j}$	contribution of DOF j velocity in the linear velocity of body i	[m/m] or [m/rad]
$\boldsymbol{J}_{Si}$	velocity transformation matrix of frame i (linear contribution)	[m/m] or [m/rad]
$\boldsymbol{\delta}_{i,j}$	contribution of DOF j velocity in the rotational velocity of body i	[rad/rad] or [rad/m]
$\boldsymbol{J}_{\omega i}$	velocity transformation matrix of frame i (rotational contribution)	[rad/rad] or [rad/m]
(X, Y, Z)	position of the robot main body in (x_0, y_0, z_0)	[m]
(Φ, Θ, Ψ)	yaw, pitch and roll angle of the robot main body in (x_0, y_0, z_0)	[rad]
(V_x, V_y, V_z)	robot global linear velocities (equivalent to $\boldsymbol{v}_{b/0}$)	[m/s]
$(\Omega_x, \Omega_y, \Omega_z)$	robot global rotational velocities (equivalent to $\boldsymbol{\omega}_{b/0}$)	[rad/s]
$(\omega_x, \omega_y, v_z)$	posture velocities in the robot frame	[m/s] or [rad/s]
β	leg duty cycle	
T	leg cycle time for one step	[s]
$\Delta\phi$	parameter determining landing, takeoff and overlap	
u	motor voltage	[V]
i	motor current	[A]
L	motor inductance	[H]
R	motor resistance	[Ω]
k_t	motor torque constant	[Nm/A]
τ_f	friction force/torque at joint	[N] or [Nm]
$\tau_{f,d}$	coulomb friction force/torque	[N] or [Nm]
$\tau_{f,s}$	breakaway force/torque	[N] or [Nm]
f_v	viscous friction coefficient	[Ns/m] or [Nms]
z	bristles average deformation	[m] or [rad]
$g()$	function of the LuGre model	[m] or [rad]
σ_0	bristles stiffness	[N/m] or [Nm]
σ_1	bristles damping	[Ns/m] or [Nms]
$\dot{q}_{st}$	Stribeck's velocity	[m/s]
$\boldsymbol{k}()$	direct kinematics function of the leg	
μ	average	
σ	standard deviation	

Conventions and mathematical operators

$\{\}_i$	vector projection in frame i
$[]_i$	matrix projection in frame i
$\circ$	operator for homogeneous transformation between a homogeneous transformation matrix and a vector

List of abbreviations

n_{cp}	number of configuration parameters
DOF	degree of freedom
TMDV	Theoretical Mechanics, Dynamics and Vibrations
HTM	homogeneous transformation matrix
MBS	multibody system
AEP	anterior extreme position
PEP	posterior extreme position

Subscripts and superscripts

r	relative to the rotational leg joint
v	relative to the vertical leg joint
h	relative to the horizontal leg joint
$/l$	with respect to the leg frame
$/b$	with respect to the main body frame
$/0$	with respect to the global frame
gnd	relative to the ground
$fric$	relative to friction
$iner$	relative to inertia
$grav$	relative to gravity
$\hat{}$	estimated variable
$*$	setpoint (reference) variable

CHAPTER 1

Introduction

For more than twenty years, the Department of Theoretical Mechanics, Dynamics and Vibrations (TMDV) of the University of Mons (Faculty of Engineering) has worked in the field of multibody dynamics: one way to extend and apply this field of research is to move towards mechatronics. According to H. Van Brussel, *"mechatronics encompasses the knowledge base and the technologies for the flexible generation of controlled motion"* [VANB96]. Ancestors of mechatronics were purely mechanical systems. Their lack of flexibility in generating more complex motions made them gradually abandoned since the second half of the 20th century. Mechatronics represents the synthesis of engineering systems, covering a wide range of topics: mechanical and control engineering, microelectronics and computer science. These components are unified to result in reactive and intelligent systems. In this context, the purpose of this work is twofold:

- proceed to the *integrated* modeling and the simulation of a mechatronic system by using a multibody modeling as a basis for the whole system;
- implement an "intelligent" behavior to the system and enhance it with the help of simulation.

The choice of the system is a six-legged robot. This is a relevant illustration of the H. Van Brussel's mechatronics definition: the challenge consists in moving the legs synchronously to make the robot walk. The system strongly interacts with the environment and has to be adapted consequently to keep a satisfying behavior of the robot body.

1.1 Context and contributions

The AMRU5 robot is the fifth machine built within the *Autonomous and Mobile Robot in Unstructured environment* framework [VERL01]. It results from a collaborative work of RMA (Royal Military Academy), ULB (Université Libre de Bruxelles) and VUB (Vrije

Universiteit Brussel). It has been created in 2002, and used in a previous PhD thesis about the neuro-fuzzy control of a leg based on pantograph mechanism [HABU04].

The robot structure was already built and motorized when this work began, but the control architecture was not satisfying. Besides the modeling of the whole robot, another important part of the work was the development of a whole decentralized control structure based on a master (PC)- slaves (microcontrollers) relationship.

The motivation of the work is to illustrate the help of dynamic simulation:

- to determine with confidence the required power for moving the robot;
- to develop gait algorithms.

The first step is the modeling of the mechanical structure with the generalized (or minimal) coordinates approaches, including the gravity and the ground contact forces. Then, the actuation layer (motors + controllers) and the joint friction are inserted, to obtain a comprehensive model of the whole hexapod. The contribution of this thesis is twofold for this part:

- The dynamic model is fully validated on a tripod gait example: indeed, simulation is often used for design of legged robots, but, to the best of our knowledge, there is no validation in terms of energy expenditure between a comprehensive model and a real machine (in the case of a walking machine actuated by DC motors). Because of the embedded nature of autonomous robots, energy consumption is obviously a key point.
- A special focus is set on friction. In the case of AMRU5, the joints are not reversible: effectively the robot becomes self-supporting when power is removed. The LuGre model has been implemented in the joints and an important measurement campaign has been made to find the LuGre parameters. It is shown that LuGre can produce irreversibility, which is difficult to obtain with other friction models that do not express the friction at zero velocity. It is also demonstrated that friction is very important and cannot be neglected in the dynamic study.

Gait algorithms have also been investigated. Periodic gait with omnidirectional ability has been developed, in which the swing foot profile is continuously updated such that the middle point the next foot stroke passes through the middle of the leg workspace, maximizing in this way the mobility potential of the vehicle. Dynamic simulations have shown that the leg profile should be enhanced to avoid antagonist forces that disturb the robot motion.

A free gait algorithm converging to forward wave gait has also been developed. It has been extensively simulated with our dynamic model, to prove that the robot is able to follow complex trajectories on uneven ground. It does not involve force sensors as it is usually the case in walking machines, but rather a posture correction which automatically adapts the body to the ground profile. An algorithm based on a current sensing and a dynamic model of the leg allows the ground detection. Moreover, the free gait is developed from a navigation point of view: instead of determining local velocity of the robot body, the heading velocities are imposed in the ground plane, such that the heading velocity is respected, whatever the relief is. Simulations and experimentations have been made using a parallel (also called concurrent-engineering) approach: for this reason, it was interesting to use the same code on both real and simulated system, avoiding repetitive programming. This justifies the choice of the flexible and open-source `EasyDyn` C++ library for the modeling.

1.2 Outline of the work

A general state of the art related to six-legged machine is given in Chapter 2. It focuses on their history and presents some landmarks and important research works. It illustrates that legged locomotion is not very old and has been increasingly studied for thirty years, mainly due to the advance in computer science. This section also highlights the diversity in structure design and in actuation.

Modeling tool and its application to AMRU5 are presented in Chapter 3. The complete model results in a rigid multibody representation with generalized coordinates and multiphysics elements like motors, controllers and joint friction. The used formulation leads to a set of ordinary differential equations, which are solved in a monolithic way using the Newmark integration scheme.

Chapter 4 presents a detailed study of the leg dynamics. Some assumptions are made, which show that the control scheme can be totally decentralized in 18 PI (*Proportional-Integral*) position controllers.

Chapter 5 is dedicated to the identification of friction parameters for the 18 joints of the robot. Steady-state friction without load on the joint is first deduced, then supplementary investigations on some specific parameters are pursued, in particular the stiction of the vertical joint and the load-friction dependency of the horizontal joint. The last part of this chapter details the ground detection algorithm, based on the current sensing in the motor and on the leg dynamic model.

Chapter 6 validates the complete model according to a tripod test case. Joint tracking errors, currents and voltages of the 18 actuators are compared. It is shown that the LuGre model is more accurate than a classical dry/viscous model for the vertical joint, because it produces an irreversibility. Power consumptions and energy expenditure are also compared. In this case the LuGre and the dry/viscous model give identical results. A mechanical defect, present on the horizontal joints, is finally treated: it is responsible for the deformation of the mechanism, which increases the friction and deteriorate the efficiency of the walking machine.

Chapter 7 compares the *engineering* and the *bio-inspired* approaches in the gait control of hexapods. The choice of gait generation algorithm for AMRU5 is first justified. Nomenclature proper to walking robot and gait concepts are given. The periodic gaits are briefly described. Thereafter complexity of the algorithms is increasing, beginning with omnidirectional gait and ending with the free gait with posture control.

Chapter 8 concludes and gives indications for further researches in the area of six-legged robot modeling and control to aspire to more efficient vehicles.

CHAPTER 2

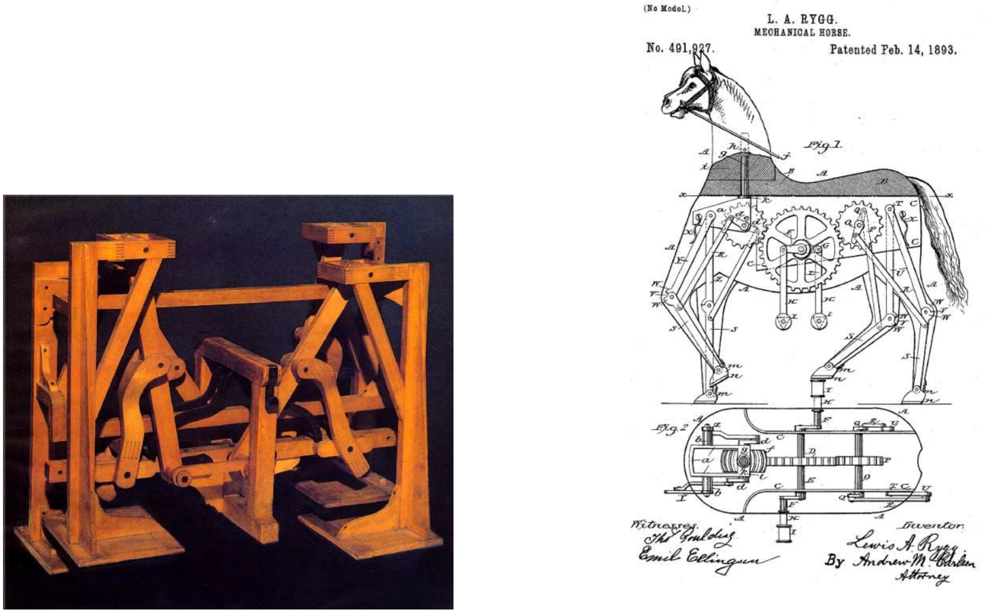
Legged robots

Legged robots have an indisputable advantage over their wheeled counterparts in terms of mobility potential. They are able to move on unstructured ground or to cross ditches or steps, where equal-sized wheeled machines could be blocked. Their legs act like an active suspension: whatever the discrete nature of the ground, the body velocity can travel smoothly, while footholds are discontinuous. Moreover, the main body is completely free in the space, allowing maneuvers by far more flexible than a standard wheeled robot.

Walking machines are first designed to operate in hostile conditions. For example, the European Atomic Energy Community financed the European project TELEMAN to respond to the needs of the nuclear industry like maintenance, decommissioning and accident management [ROBE97]. Demining is another dangerous task which could be handled by mobile robots. Wheels are widely spread in this field [DEBE05], because wheeled robots are fast and can carry large loads (demining unit), but legs reduce the footprints on the ground, and consequently the risk of activating a mine.

The second motivation for developing legged robots is to study the insect and/or animal behavior. Bio-inspired robots are developed for more than twenty years; bio-inspiration concerns the leg design as well as the control architecture. The mechanical design involves the actuation of the legs, their number, and their configuration with respect to the main body. The control algorithms implement local reflex abilities, and try to mimic at best the gait pattern observed in cockroach-like insects.

The present section begins with some history about legged machines, from the early designs to nowadays. It would be impossible to mention all the existing legged machines in the world: our choice has been to present some of the remarkable robots built in the past in Section 2.1, and then to describe robot by labs which have contributed significantly to research in legged machines in Section 2.2. The end of this chapter focuses on the mechanical design and control architecture of AMRU5 presented in Section 2.3.



(a) Chebyshev's walking machine [CYBE] (Chebyshev 1850)

(b) Rygg's mechanical horse [CYBE] (Rygg 1893)

Figure 2.1: Early legged robots

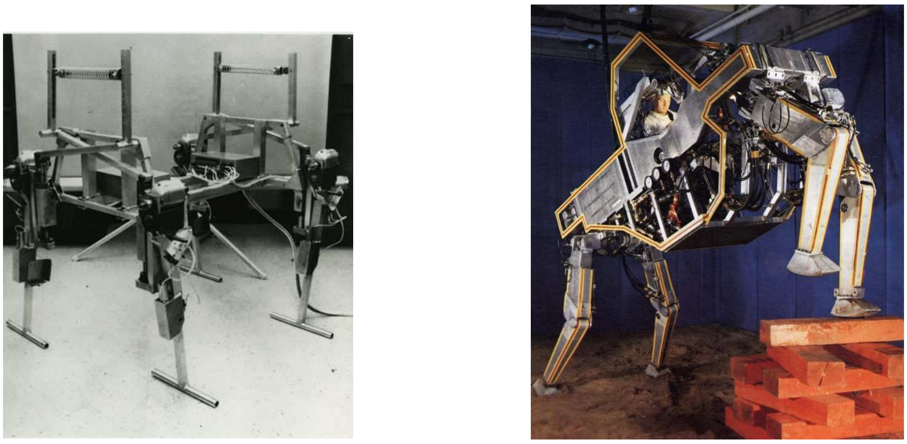
2.1 A little History

2.1.1 From 19th Century to 1970

Legged mechanisms have a long history. In 1850, Chebyshev designed a four legged machine with mechanically synchronized legs, based on straight-line linkages (Fig. 2.1(a)). In 1893, Rygg obtained a patent for the human-powered mechanical horse (Fig. 2.1(b)). About two decades later, while the automotive industry was in full expansion, Bechtolsheim had the idea of a "legged car". This concept was also patented, but success about this machine was never reported.

During more than sixty years, the concept of walking machine was focused on more and more complex linkage design, driven by only one source of power (Shigley 1960, Balndwin and Miller 1966 with the walking wheel chair for disabled people). The pure mechanical solution is nice, and reliable, but is obviously limited to flat ground. It becomes an evidence that dealing with less predictable environment requires control.

The pioneers in term of control were Franck and McGhee (1968) with *Phony Pony* (Fig. 2.2(a)). Each leg had two joints (a hip and a knee) and was actively controlled by a finite state machine. The robot was able to emulate walking patterns as crawl and trot. An impressive quadruped, called *General Electric Walking Truck* (Fig. 2.2(b)) was developed by Mosher in 1969. With its 1300 kg, it was able to accommodate a driver whose arms and legs were directly commanding the four legs of the machine. Thanks to high fidelity force feedback, the driver was able to control the robot efficiently after some hours of training. This robot is, still now, one of the most accomplished legged vehicles.



(a) Phony Pony [CYBE] (McGhee 1968)

(b) GE quadruped [CYBE] (Mosher 1969)

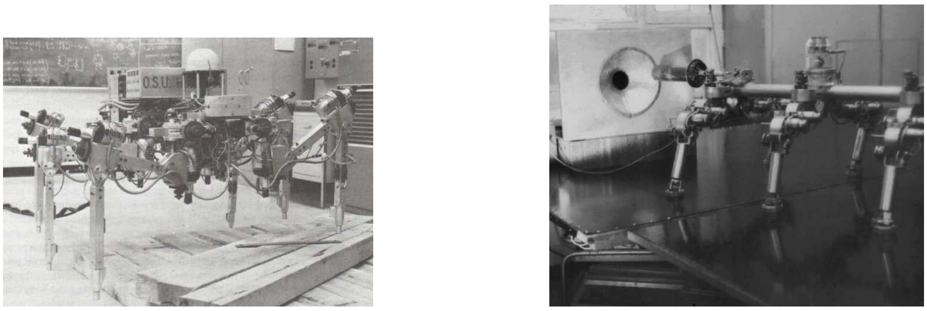
Figure 2.2: Remarkable quadrupeds in the late sixties

2.1.2 From 1970 to 1980

Progress in digital computers marks a transition in the development of legged machines. The first landmark of the seventies is due to the McGhee's group at Ohio State University [MCGH79]. They implemented successfully wave gait on the OSU hexapod (Fig. 2.3(a)). Leg motions are coordinated by computer and no more by human. This project has given rise to many of the present research topics in legged locomotion, and more generally, in robotics. First, the concept of "active compliance" was elaborated [KLEI80]: force information was used in the feedback position control loop to detect the ground. Energetic efficiency on soft surfaces (sand for example) had also been envisaged. The related studies conclude that legged machines should be more efficient than rolling ones was formulated, because rolling induces a bow-wave effect on the ground. However, tests had shown that the efficiency is really bad for legged robots, mainly because of the high reduction ratio of the actuators, meaning bad power efficiency and high friction losses. They also used Jacobian transforms to link foot and joints coordinates, which is the base of inverse kinematics widely encountered in robotics.

While the United States had the monopoly of walking machines, the Russian researchers also began to develop hexapods. Okhotsimski in 1976 developed a walker with terrain scanning; Gurfinkel, Schneider, Gorinevski et al. worked on *Masha* (Fig. 2.3(b)), and especially on force sensing at ground interface, for different types of soil.

As Japan is concerned, the development of autonomous robots was mainly directed by the team of Sigheo Hirose at the Tokyo Institute of Technology. Their first machines were climbing and snake robots, but they came to legged machines in the early eighties, by developing quadrupeds with pantograph-based legs and reflex-like control to avoid obstacles.



(a) OSU hexapod [CYBE]
(McGhee et al 1977)

(b) Masha hexapod [CYBE]
(Gurfinkel et al 1969)

Figure 2.3: Remarkable hexapods in the late seventies

2.2 Since 1980 across the world...

Since 1980 walking machines spread around the world, becoming an important research field. The following section presents the evolution of four and six-legged robots by research teams that have significantly contributed to progress in this domain.

2.2.1 United States of America

USA are unquestionably the leader in legged locomotion. They have a long tradition and many research centers, whose most famous are detailed below.

Ohio State University Song and Waldron developed the Adaptive Suspension Vehicle (ASV) to walk on rough terrain. Still nowadays, it is one of the biggest and most successful legged machines of this size. Several kinds of gait were implemented, to deal with various environments. It weights 2700 kg and can reach up to 3.6 m/s.

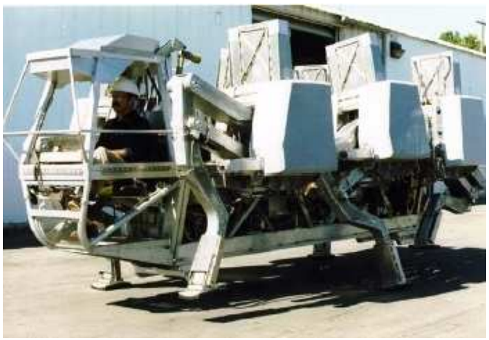
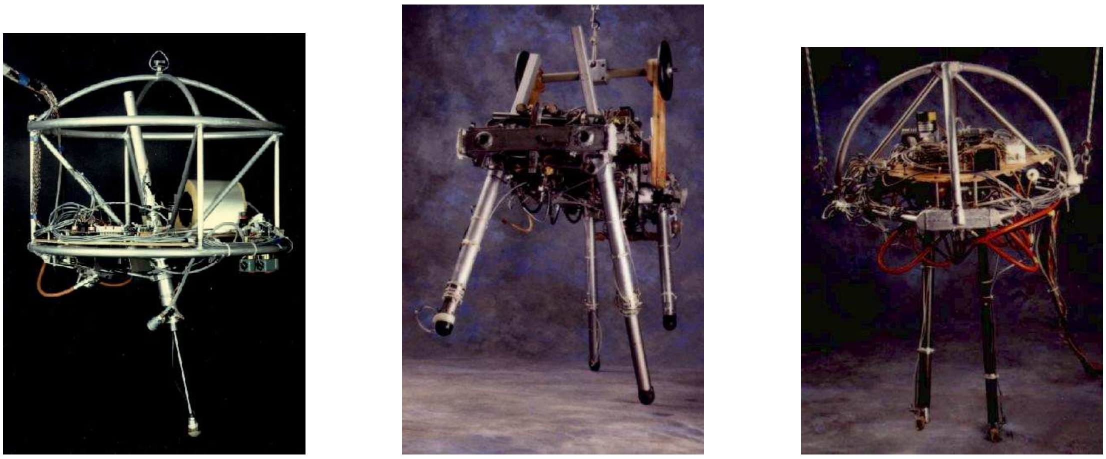


Figure 2.4: ASV [CYBE] (1985)

Massassuchet Institute of Technology Several robotics labs exists in MIT and have widely contributed to the walking machine history. M. Raibert developed the legged lab in the beginning of the 80's. His research team deeply investigated dynamics and balance of walking: the first studies were made on a hopping robot (Fig. 2.5(a)), then extended to quadruped (Fig. 2.5(b)) and biped (Fig. 2.5(c)). Note that the biped is able to make a somersault. This labs ends its activity at the end of nineties.



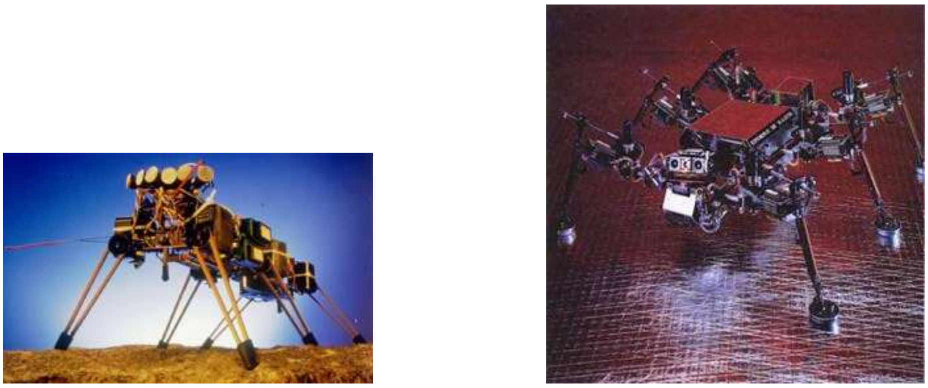
(a) 3D hopping robot [MIT] (1984)

(b) Quadruped robot [MIT] (1984)

(c) Biped robot [MIT] (1990)

Figure 2.5: Legged robots that balance

Another important contribution comes from the MIT Humanoid Robotics group. R. Brooks developed the *subsumption* architecture and implemented it on Genghis (Fig. 2.6(a)) and Hannibal (Fig. 2.6(b)). The concept of this new architecture is the following: a basic layer is based on a strong coupling between actuators and sensors, promoting a more reactive behavior than in the classical control approaches. Higher levels are based on the skills developed by lower levels to enhance their behavior.



(a) Genghis [MIT2] (1989)

(b) Hannibal [MIT2] (1990)

Figure 2.6: Robots with subsumption control architecture

Boston Dynamics M. Raibert and his colleagues founded the Boston Dynamics company in 1992. It creates innovative animal-like robots with advanced behavior. The most famous is probably *BigDog* (Fig. 2.7), able to move up to 4 mph on very hard terrains, to climb slopes up to 35 degrees and to carry a load of 150 kg. This is doubtless the most advanced legged machine nowadays.

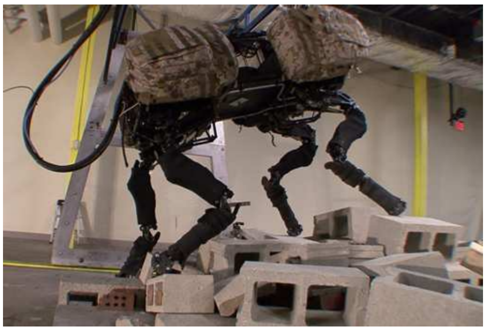


Figure 2.7: BigDog [BOST] (2008)

Case Western Reserve University Biologically Inspired Robotics Center is directed by R. Quinn since 1980. Their first prototype of legged machine was built to test a biologically-inspired neural network controller developed by R. D. Beer and H. J. Chiel for a computer simulated insect. Then, collaboration with biologists leads to new hexapod designs like *Ajax* (Fig. 2.8), whose leg kinematics is inspired from the cockroach *Blaberus discoidalis*. The lab also works on worms, flying machines and whegged (wheel + leg) robots.

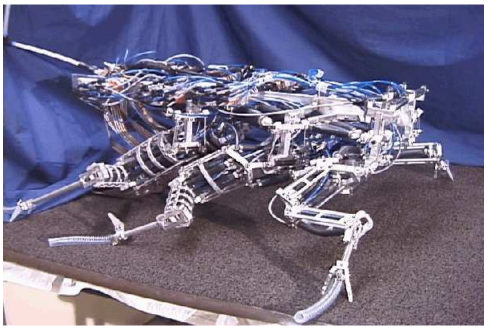
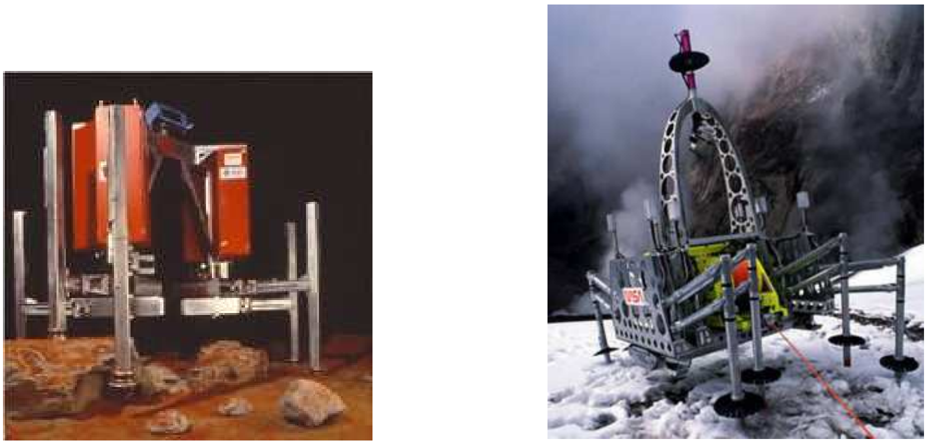


Figure 2.8: *Ajax* [CRWU] (2005)

Carnegie Mellon University W. L. Whittaker's team developed robots to walk in hard environments, like *Ambler* (Fig. 2.9(a)) with the originality of orthogonal actuator placement to facilitate the legs control, or the famous *Dante II* (Fig. 2.9(b)), to explore the Mt. Spurr volcano in Alaska. *Dante II* was able to go down along crater walls in a rappelling-like manner to gather and analyze high temperature gazes from the crater floor.

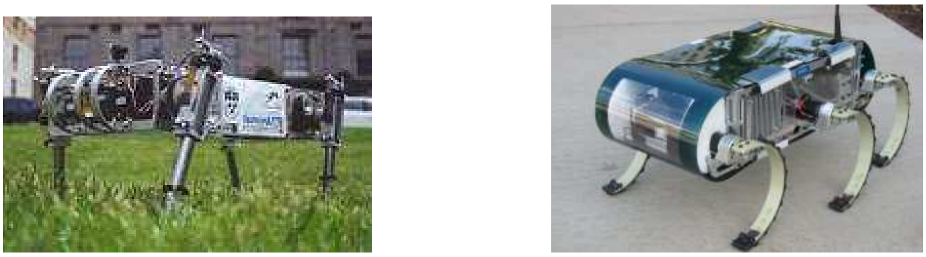
2.2.2 Canada

The Ambulatory Robotics Lab (1991-2003) of McGill University develops several robots with dynamic walking, under the direction of M. Buehler. The most famous are the Scout's series (Fig. 2.10(a)), and the *RHex* hexapod (Fig 2.10(b)). *RHex* is a quite original robot which can travel across hard terrains, with only one actuator per leg. It was a collaborative work between McGill, Michigan and Berkeley universities.



(a) *Ambler* [CMU] (1990) (b) *Dante II* [CMU] (1994)

Figure 2.9: CMU hexapods

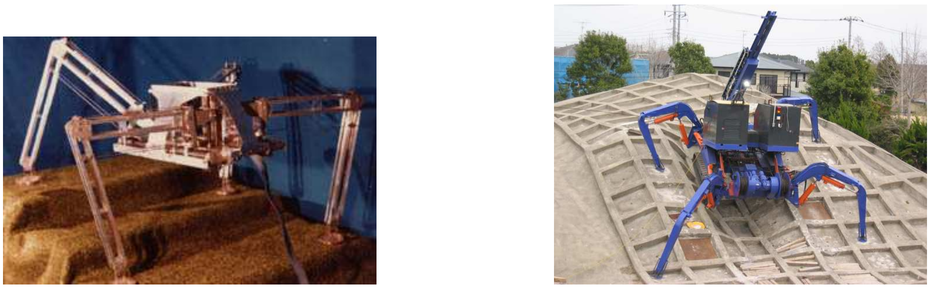


(a) *Scout II* [OMIC] (2003) (b) *RHex* [OMIC] (2003)

Figure 2.10: McGill legged robots

2.2.3 Japan

The main contributions of Japan come from The Fukushima Robotics lab of the Tokyo Institute of Technology, directed by S. Hirose. The lab research focuses on quadruped robots. One of the first built was *PVII* (Fig 2.11(a)), whose originality was the pantograph mechanism: the latter avoids a negative work during the gait cycle, and facilitates the kinematic control because of the gravitational decoupling. Legs of AMRU5 are based on this design. A long series of quadruped called *Titan* has been developed since 1980 to nowadays. The latest is called *Titan XI* (Fig. 2.11(b)). A particular attention has been paid to the leg design of each robot.



(a) *PVII* [TIT] (1978) (b) *Titan XI* [TIT] (1996)

Figure 2.11: Tokyo Institute of Technology quadrupeds

Finally, it could be impossible to end the Japanese contribution without citing the *AIBO* pet dog, developed in 1999 by Sony.

2.2.4 Europe

Germany is very active in development of four-, six- or eight-legged machines. Other European countries have a more modest contribution, but some remarkable legged machines are nonetheless cited.

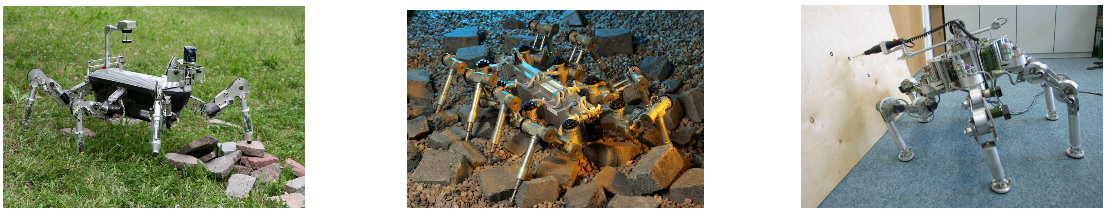
Germany B. Gassman and J. Albiez's team from FZI Karlsruhe is working on the *Lauron's* series with stick-insect like design. The fourth of the series (Fig. 2.12(a)) is equipped with omni-remote-vision system and numerous sensors; the final aim is to use it for urban rescue missions.

Biologists are also involved in the modeling of neural network controllers which mimic the insect behavior. Universities of Duisburg (M. Frik) and Bielefeld (H. Cruse) implemented successfully such controllers on relatively small (<5 kg) insect-inspired robots.

The *Scorpion* eight-legged robot of F. Kirchner from Bremen (Fig. 2.12(b)) is also inspired from biology. It uses a walking pattern inspired by the movement of scorpions coupled with reflexes that will help the robot to free a stuck leg, among other things.

Beyond the bio-inspired researches, there are legged robots controlled according to more conventional engineering approach. U. Schmucker and A. Schneider from the

Fraunhofer Institute for Factory Operation (IFF) developed force sensing and gait algorithm on *Katharina*, whose aim is to execute accurate tasks like drilling a hole in a wall (Fig. 2.12(c)).



(a) *Lauron IV* [FZI] (2004)

(b) *Scorpion* [BREM] (2000)

(c) *Katharina* [IFF] (2001)

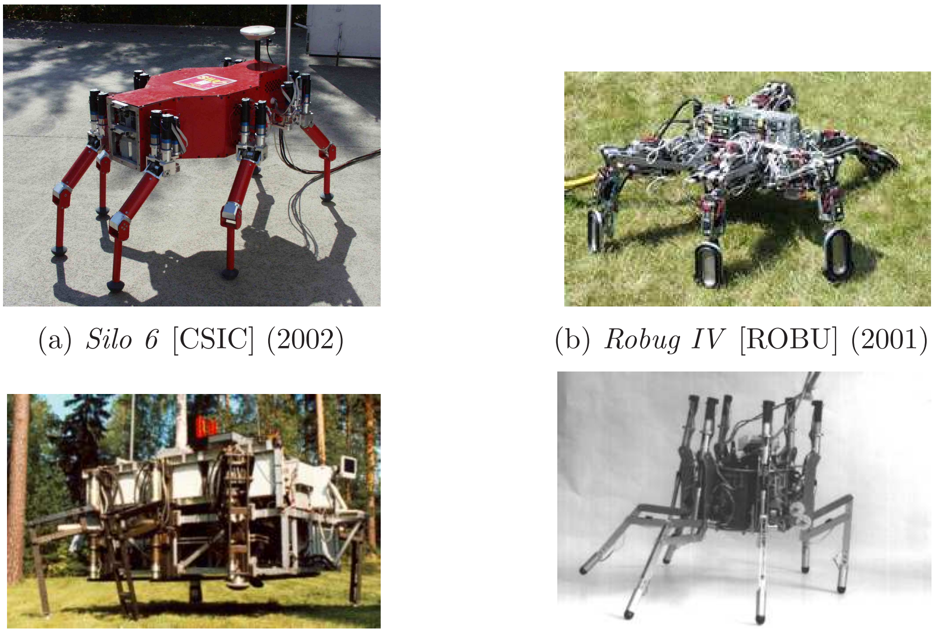
Figure 2.12: Legged machines in Germany

Spain The Department of Automatic Control at the Institute of Industrial Automation of Madrid has about twenty years experience in the field of legged locomotion. They worked on walking in hazardous environment and in humanitarian demining. Team of M. A. Jimenez and P. Gonzalez de Santos created *Rhimo*, *Silo 4* and *Silo 6* [GONZ05] (Fig. 2.13(a)).

United Kingdom *Robug IV* (Fig. 2.13(b)) is an eight-legged robot with pneumatic actuators. It was designed for rescue, inspection and maintenance tasks in hazardous environments. Each foot has a vacuum gripper. It can carry a 100 kg payload while climbing, and perform floor-to-wall, wall-to-roof, and wall-to-wall internal and external plane transitions.

Finland The Automation Technology Laboratory of Helsinki University, with A. Halme et al, was an important contributor to free gait and terrain adaptation development [HALM94, HALM96, SALM96]. They implemented their algorithms on the Mecant (Fig. 2.13(c)), a fully independent hydraulic six-legged walking machine weighting about 1100 kg.

Belgium The early research in Belgium was directed by A. Preumont at the Université Libre de Bruxelles. *Max*, a robot similar to Genghis, was developed for gait studies. Then, P. Alexandre [ALEX96, ALEX97] made further research on *Silex* (Fig. 2.13(d)) and developed an original free gait algorithm, as well as active suspension to adapt altitude and attitude of the robot. The concept of active compliance was also developed on the conceptual walking vehicle *Ioan* [PREU97]. The department of Mechanics of the Royal Military Academy supervised by Y. Baudoin also developed six legged robots, within the framework of the Autonomous Mobile Robot in Unstructured environments (AMRU) [VERL01]. Research about the neuro-fuzzy control of a leg was made by J.C. Habumuremyi [HABU04] on *AMRU5*.



(a) *Silo 6* [CSIC] (2002)

(b) *Robug IV* [ROBU] (2001)

(c) *Mecant* [TKK] (1992)

(d) *Silex* [ULB] (1996)

Figure 2.13: Legged machines from Europe

2.3 AMRU5 among these legged machines

AMRU5 (Fig. 2.14) is a six-legged robot with hexagonal architecture developped for demining purposes. It has been built by the Royal Military Academy of Belgium, in collaboration with the VUB (Vrije Universiteit Brussel) and the ULB (Université Libre de Bruxelles)

2.3.1 Mechanical overview

The robot has a weight of 34 kg and an outer diameter which can reach 1.4 m. The main body weights 11.2 kg, with a circular shape of radius 0.175 m. The maximum reachable velocity on flat ground is about 3 cm/s. The six legs are equally distributed around the robot body. The frame $O_b x_b y_b z_b$, located at the robot center of mass, gives the forward advance along $O_b x_b$ and the lateral motion along $O_b y_b$; $O_b z_b$ is upwards. Numbering of the legs is given in Fig. 2.14.

Each leg has three degrees of freedom (DOF) (Fig. 2.15(a)):

- q_r is the rotation of the leg around a vertical axis;
- q_v allows the up and down motion of the foot;
- q_h describes its back and forth (or radial) motion.

Each degree of freedom is actuated by a Maxon RE35 DC motors. It's worth to notice the pantograph mechanism which allows a gravitational decoupling, and should theoretically result in a better efficiency in the gait than classical revolute joints placed along the

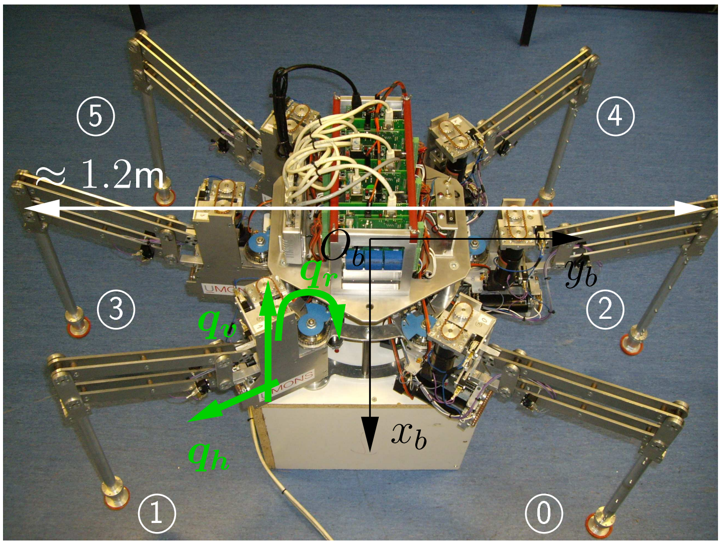
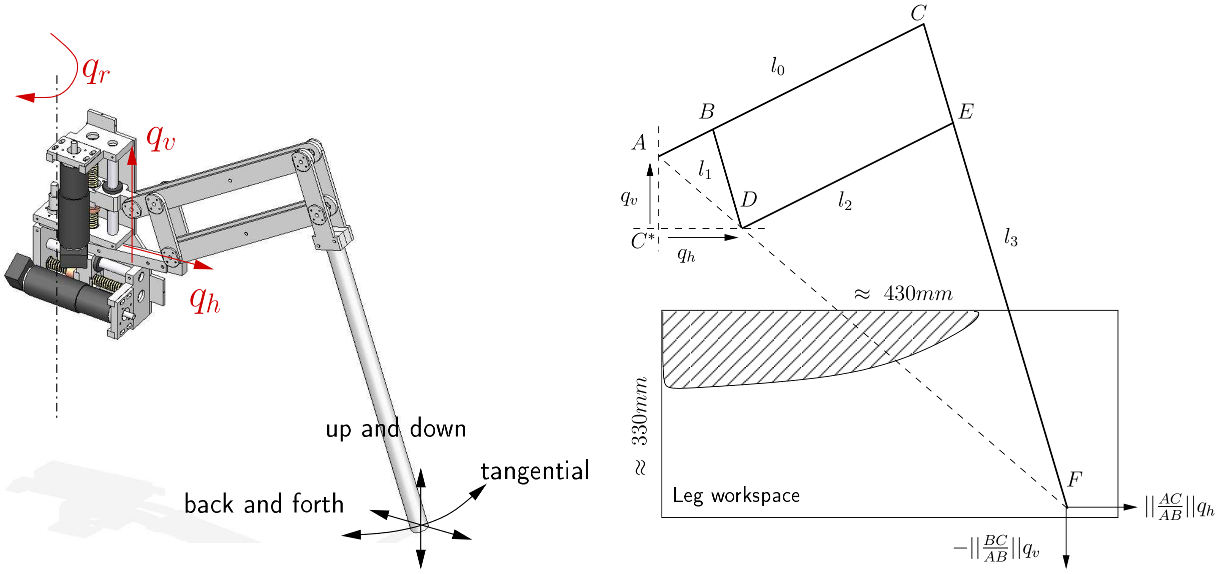


Figure 2.14: AMRU5 hexapod

leg [HIRO84]. The foot trajectory generation is also easier, because the up and down motion of the foot is directly proportional to q_v with an amplification ratio of -6 (the same rule applies for the back and forth motion and q_h with an amplification ratio of 7).

Figure 2.15(b) gives the detail of the pantograph mechanism in the frontal plane. The hatched area is unreacheable because of mechanical limitations. Some relevant characteristics of the leg are gathered in Table 2.1. For a detailed description of the robot, please consult Appendix B.



(a) CAD view of leg mechanism (without rotational actuator)

(b) Planar view of pantograph

Figure 2.15: Leg overview

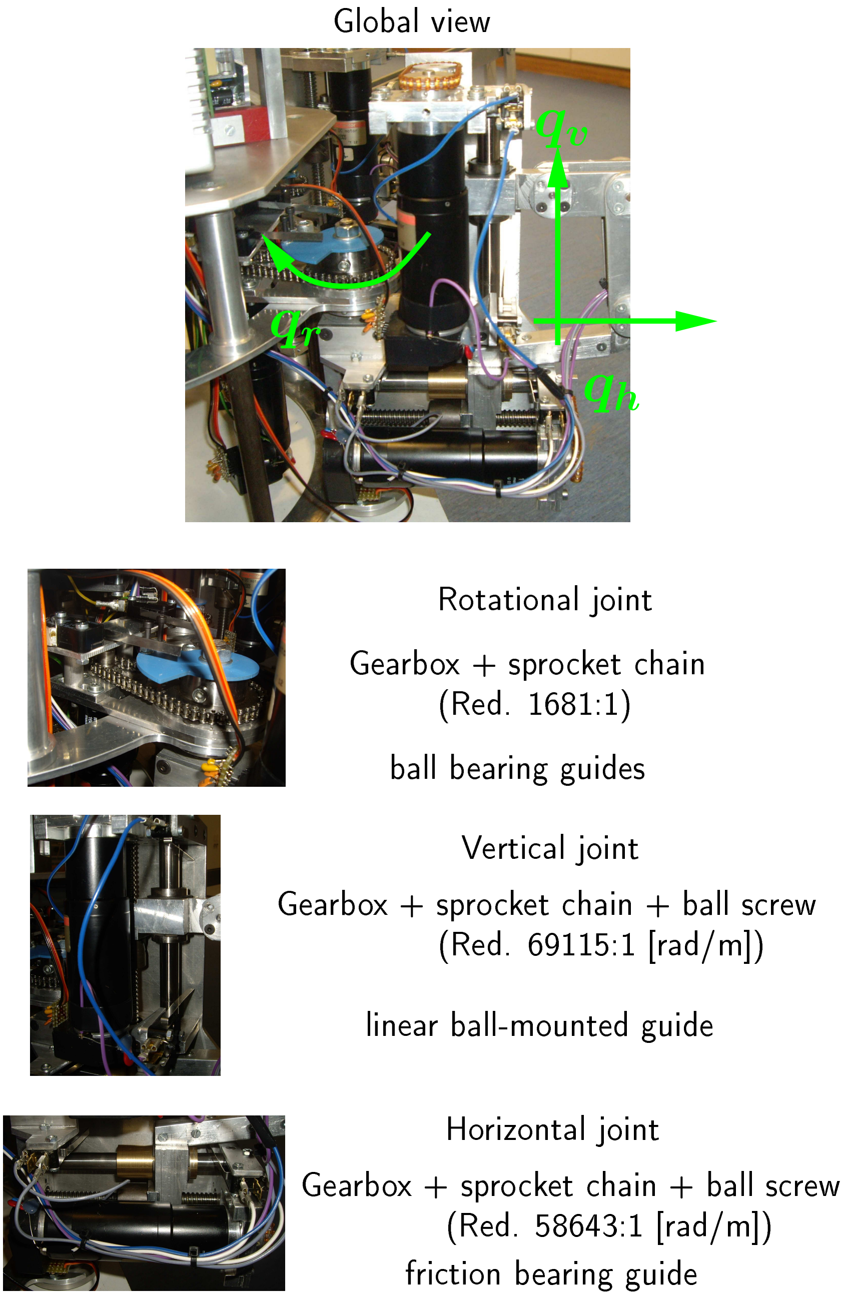
Each leg is assembled to a chassis, on which the vertical and horizontal motors are fixed.

Part	Length (m)
l_0 (AC)	0.28
l_1 (BD)	0.07
l_2 (DE=BC)	0.24
l_3 (CF)	0.492
AB	0.04

Table 2.1: Relevant geometrical characteristics of the pantograph

For both motors, the transmission between the rotor shaft is composed of a planetary gearhead, whose output is connected to a ball screw combined with a linear guide. For the vertical joint, this is a linear ball-mounted guide, and for the horizontal joint, a friction bearing. The position of the nut driven by the screw is the DOF (q_v or q_h) considered in the model of Section 3.4. Positions are given with respect to point C^*, called the *crossing point*: it results from the intersection of the driving lines of q_v and q_h. The leg chassis is fixed to the main robot body via the revolute joint q_r (see Fig. 2.16). The transmission between the rotational actuator and the leg chassis is composed of one planetary gearhead at the motor followed by a chain sprocket. The rotational DOF q_r is the position of the leg chassis with respect to the main body.

The global reduction ratios between the rotor shafts and the considered DOF, as well as the transmission elements, are shown in Fig. 2.16. Note that each joint is equipped with mechanical switches limiting their stroke. The motor power is cut off when a switch is activated.



Global view

Rotational joint

Gearbox + sprocket chain
(Red. 1681:1)

ball bearing guides

Vertical joint

Gearbox + sprocket chain + ball screw
(Red. 69115:1 [rad/m])

linear ball-mounted guide

Horizontal joint

Gearbox + sprocket chain + ball screw
(Red. 58643:1 [rad/m])
friction bearing guide

Figure 2.16: Focus on AMRU5 transmissions

2.3.2 Control architecture

The hardware architecture of legged robots usually consists of three levels [ORIN82]:

- **Level A: navigation:** the trajectory of the main robot body is programmed via setpoints, or given in real-time by a user equipped with a joystick for example.
- **Level B: motion planning:** this level is responsible for the legs motion and their synchronization. This is the "creative" level where original algorithms concerning stability of the robot, gait pattern determination, optimal repartition of ground contact forces, orientation and altitude control of the main body, ... can be developed.
- **Level C: motion execution:** local leg controllers have to follow the trajectory generated by level B. Note that biologically inspired robots tend to drain more and more tasks from the motion planning to implement them as *local reflexes* at this level. More details will be given in the state of the art of gait control in Section 7.2.

The global architecture of the AMRU5 robot is depicted in Fig. 2.17. Level B, or master, is rated 100 Hz: it prepares the 18 motor voltages $(u_r^0, ..., u_h^5)$ resulting from gait algorithm and PI controllers. Level C is composed by six slave boards. Each of them is controlling one leg and is measuring joint positions (q_r, q_v, q_h), motor currents (i_r, i_v, i_h) and voltages (u_r, u_v, u_h). Master-slaves are linked to a simple Ethernet switch inside a local network. Level A is not really implemented on the robot; there are neither localization, nor inclination or direction measurements. However, the architecture is quite open and lets the possibility to add this layer in future work.

Master Computer

The master is a PC (dual core 3GHz, 1.5 Go RAM) running real-time Linux kernel 2.6.31-11 PREEMPT RT (Ingo Molnar's patch [MOLN10]), which enhance considerably the latency, i.e. the delays in instructions.

The master routine is based on a blocking standard C function `select()`, which polls among a set of file descriptor representing the slaves. This function is "unlocked" in three cases:

- an error happens;
- data are incoming on the master;
- a timeout occurs.

The beginning of the control loop consists in sending new computed commands to the 6 slaves simultaneously by means of a *broadcast* packet (proper to UDP/IP protocol developed in Section 2.3.2). The broadcast ensures that all the slaves are synchronized. Then master waits for their responses. Once all data are received, the gait algorithm is executed and PI controllers generate the commands for the next control time. The master waits during the control step remaining time, computes thereafter the next deadline, and

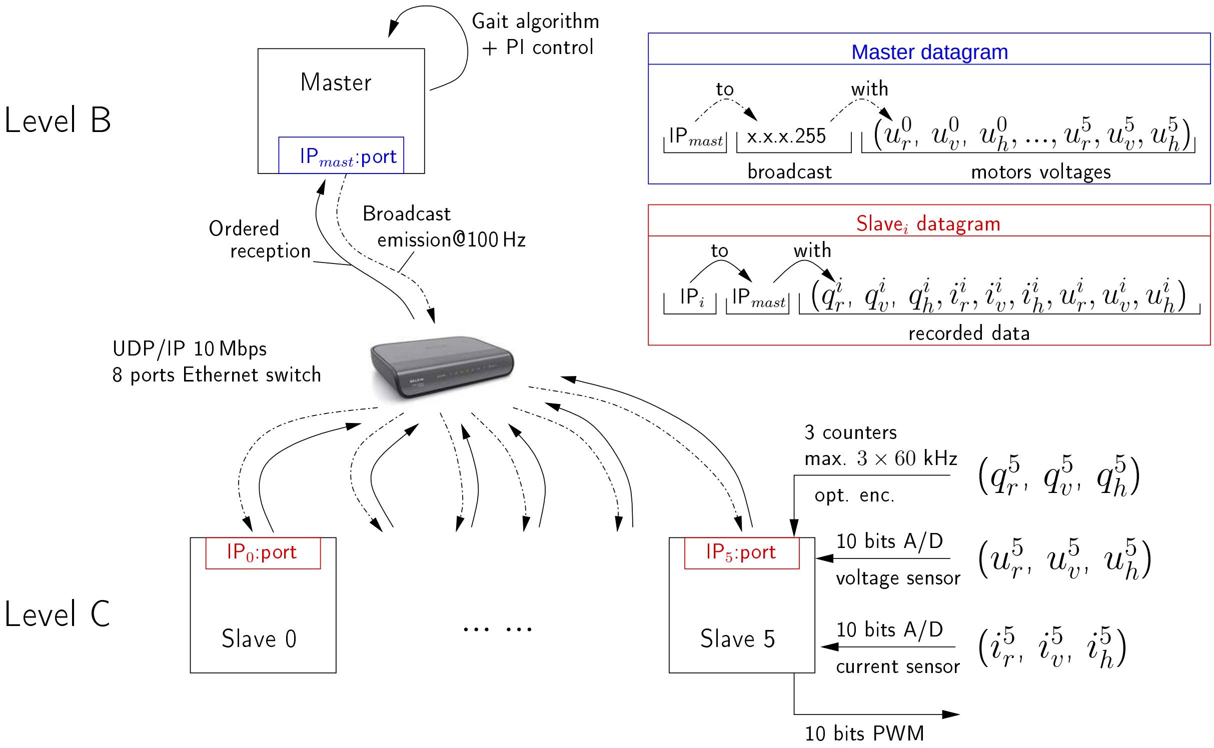


Figure 2.17: Global communication scheme

sends immediately the new motors commands by broadcast. The following pseudo-code summarizes the master execution:

```
deadline = now() + delay; // delay of 10ms

while(robot_has_to_move())
{
  if(all_data_from_slaves_received)
    command = compute_motor_command();

  switch(select(socket,deadline-now()))
  {
    case -1: // network error
      return(error);
    case 0: // timeout is over
      broadcast(targets);
      deadline = deadline + delay;
      break;
    case 1:
      data_from_slave = receive(socket);
      break;
   }
}
```

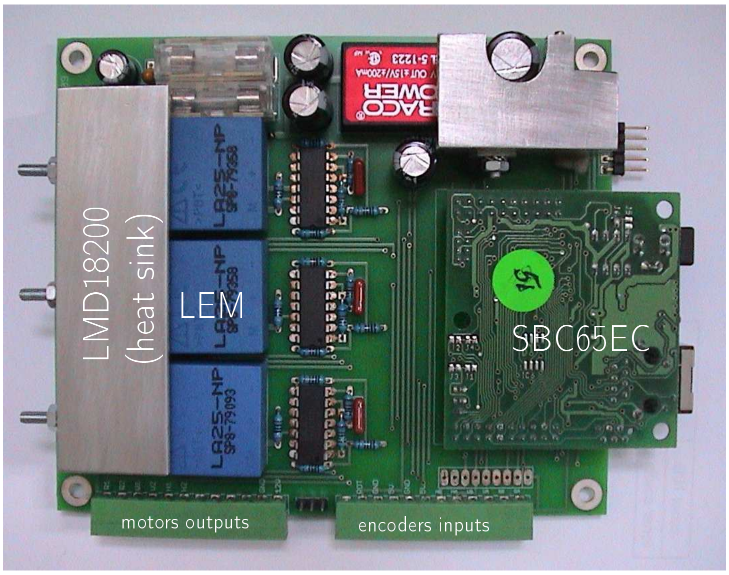


Figure 2.18: Overview of the PIC-based slave board

Slave Board

Each slave board (Fig. 2.18) works with a SBC65EC board from MODTRONIX: this small single board computer with Ethernet capabilities (10 Mbps) is based on a PIC18F6627 running at up to 40 MHz. Schematic of the board can be found in Fig. B.2 of Appendix B. Note that all the presented circuitry and programs have been developed by the Department of TMDV of the UMons.

Basically, each slave board has four tasks to complete:

1. **Manage local network communication**: this first task includes global packet reception from the master, and the forwarding of real-world informations proper to each leg.

2. **Collect continuously positions of the three motor shafts**: the reading of encoders is executed continuously thanks to a hardware interrupt mechanism and an interrupt service routine, which is triggered at each rising edge from the optical encoder. Encoders have a resolution of 500 impulses/revolution which leads to a signal frequency of about 60 kHz per encoder for full speed. Note that position used in the master control update is stored as soon as the PWM signal has been applied to the H-bridges, and that PI control is made on the stored values.

3. **Drive the three motors**: the three master commands are translated in a 10-bit PWM signals for the LMD18200 H-bridges, which deliver necessary power to the motors.

4. **Perform A/D conversions of three motor currents and the board voltage supply**: they are executed after the PWM application. Voltage is measured only one each board, because power source is identical for the three motors. The voltage of each motor is deduced from this value and from the PWM command. Even if the

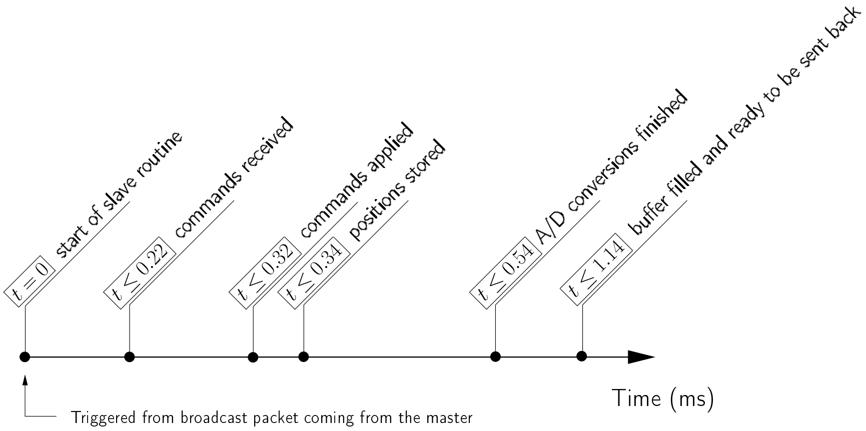

$t = 0$ start of slave routine
$t \leq 0.22$ commands received
$t \leq 0.32$ commands applied
$t \leq 0.34$ positions stored
$t \leq 0.54$ A/D conversions finished
$t \leq 1.14$ buffer filled and ready to be sent back
Time (ms)
Triggered from broadcast packet coming from the master

Figure 2.19: Timing diagram of the slave

H-bridge has a current output, the latter does not take into account the recirculating currents: consequently they are overestimated. Moreover, the high gearing of the motors requires an accurate sensing: this is why a Hall-effect current sensor (LEM LA 25-NP/SP8) has been directly inserted in the motor circuitry.

The time diagram in Fig. 2.19 summarizes precisely the necessary time for each task of the slave (time is relative to the slave). Note that these times depend on the CPU load of the microcontroller, which itself depends on the frequency of the rising edges coming from the position sensors, thus from the motors velocity. For this reason, the time at which operations are completed is not precisely determined. An upper bound is set to 1.5 ms to consider that the slave is ready to emit data. This value has been checked experimentally with full velocity of the motors.

It is worth to mention that no computation occurs on the board. This way of working could be criticized because a lost of data information due to bad communication prevent from a correct PI control. However, it has been checked experimentally that the communication protocol is very reliable and never loss data. A particular attention has been paid to develop a very generic algorithm on the slave boards: in this way, no re-programming is needed and controller implementation can takes place on the master directly, enjoying the computational efficiency of the PC instead of the poor floating point calculation on microcontrollers.

Communication

In control systems, there are usually two choices for communication interface:

- Parallel bus (ISA, PCI, Multibus, ...) is a set of parallel wires which allow high data transmission rates, but with limited to relatively short distance because of the difficulty of keeping the signal integrity. In the case of walking robots, these distances are well lower than limits, and therefore parallel bus is usable (see for example the SILO4 architecture in [SILO4]).

- Serial bus (RS232, CAN, Device Net, Profibus, Ethernet, ...) is a serial communication channel, transmitting one bit at time. Devices are organized in network.

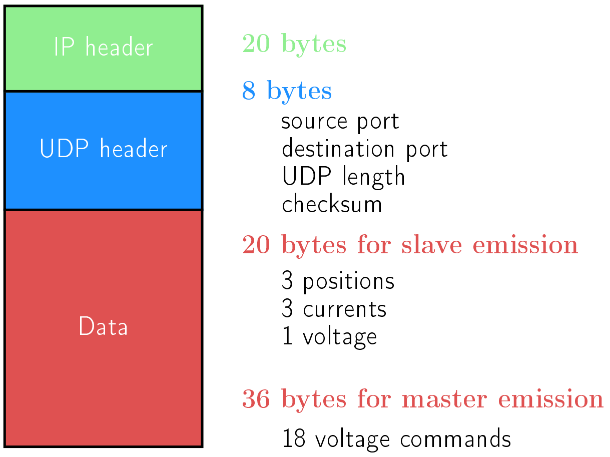


Figure 2.20: UDP data encapsulation [KURO08]

Among this set of solutions, the Ethernet communication is appealing, mainly because of its large bandwidth (up to 1 Gbps), and the capability to give an address to identify the devices on the network. Also, priority and collisions are managed in a transparent way.

Above this layer, the Internet tools have been developed in ANSI C to send data packets over the network. Standard functions allow the data conditioning, called *encapsulation*, and the creation of *sockets*, which are the access points in a program.

Two widespread protocols exit with the Internet: TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). TCP is connection-oriented, and guarantees a reliable transmission over the network. Indeed, an acknowledgement is sent by the receiver in case of success; otherwise, a request asks to the emitter to send again the data, until successful transmission. The weakness of this method is the time consumed for acknowledgement(s) and data retransmission(s). One alternative is the connectionless User Datagram Protocol UDP. Connectionless means that the target has no way to know if data has been lost during transmission, but communication is faster.

In control applications like legged machines, the network is isolated from external perturbations (local network) thus the lower layer is very reliable. Moreover, the network traffic is predictable because size of packets do not change, and time for emissions and receptions are determined. The next point is that, even if a packet is lost, the update cycle will be missed, but corrected during the next cycle. Practically, losses of packets are extremely rare. For these reasons, UDP is most of the time preferred to TCP [AUSL02]. UDP also has the ability to send *broadcast* messages intended for every listener of the network. This ability is used to synchronize the control process: the master sends in one time all the commands to all the slaves, so they can begin their tasks simultaneously.

Practically, slaves and master have two sockets each: one for reception and one for emission. Data are encapsulated as shown in Fig. 2.20: the IP header contains an address proper to the device, and the UDP header contains the source and destination ports, which are proper to the running applications.

Network Load The eighteen commands of the master are gathered in a 36 bytes buffer, while the real-world data are put into a 20 bytes (buffer per slave).

There are two rules to ensure a correct working of the communication:

1. avoid segmentation of a frame in the network: it is recommended to limit the size of UDP frames according to the following relationship:

   $$\mathtt{UDP}_{frame} \leq \mathtt{Path\ MTU} - 28\,\mathtt{bytes}.$$

   Path MTU (Maximum Transmission Unit) is 1500 bytes for UDP: this condition is fulfilled;

2. the maximum transmission rate must not be exceeded. Every 10 ms, there are 36+20+8 = 64 bytes emitted by the master and 6×(20+20+8) = 288 bytes by the slaves. So, the total transfer rate would be 100×(64+288)= 35,2 kB/s, widely inferior to the 1250 kB/s ethernet connection (10 Mbps = 1250 kB/s).

Note: while for buses, the pair of communicating devices is usually called *Master/Slave*, in the case of network terms *Client/Server* are rather used: for AMRU5, the servers (slave boards) are waiting for the client request (broadcast packet with the motor commands), then proceed to motor control and data acquisition to send back data to the client.

Data synchronization

The complete control sequence from the master point of view is given in Fig. 2.21.

The time t_{s0} at which the first slave begins its emission and the time interval Δt_s between two slave emissions are sensitive parameters in the communication scheme.

The first one is set in such a way that all the slave tasks described in Section 2.3.2 are completed: it has been experimentally checked that 1.5 ms gives a correct behavior. Accounting for the delay transmission, the slave 0 begins its emission at 2.5 ms in the control period.

The time interval Δt_s must not be too long because the gait algorithm is performed only at the end of all the slaves data reception, but it must be high enough to avoid interferences between two slaves emissions. Indeed, the slave CPU load caused by the interrupt-driven routine which counts the impulses from the motors can modify the emission instant over the ethernet. The more the motors are fast, the more the load is important, and the more the emission time is delayed. This is represented by the Δt_d in Fig. 2.17. Note that gait algorithm computation is not limiting in the communication scheme: time required for each gait algorithm will presented in Chapter 7. Generally speaking, these calculations takes less than 1 ms.

Figure 2.22 gives more detail about the reception time of the slave data in the case of a tripod gait. Index ⓘ denotes the slave identification number. During the swing phase of legs 1-2-5 (grey area), the vertical actuator is moving fast, loading the CPU, and consequently delaying the time at which the data are sent back to the master. Emissions

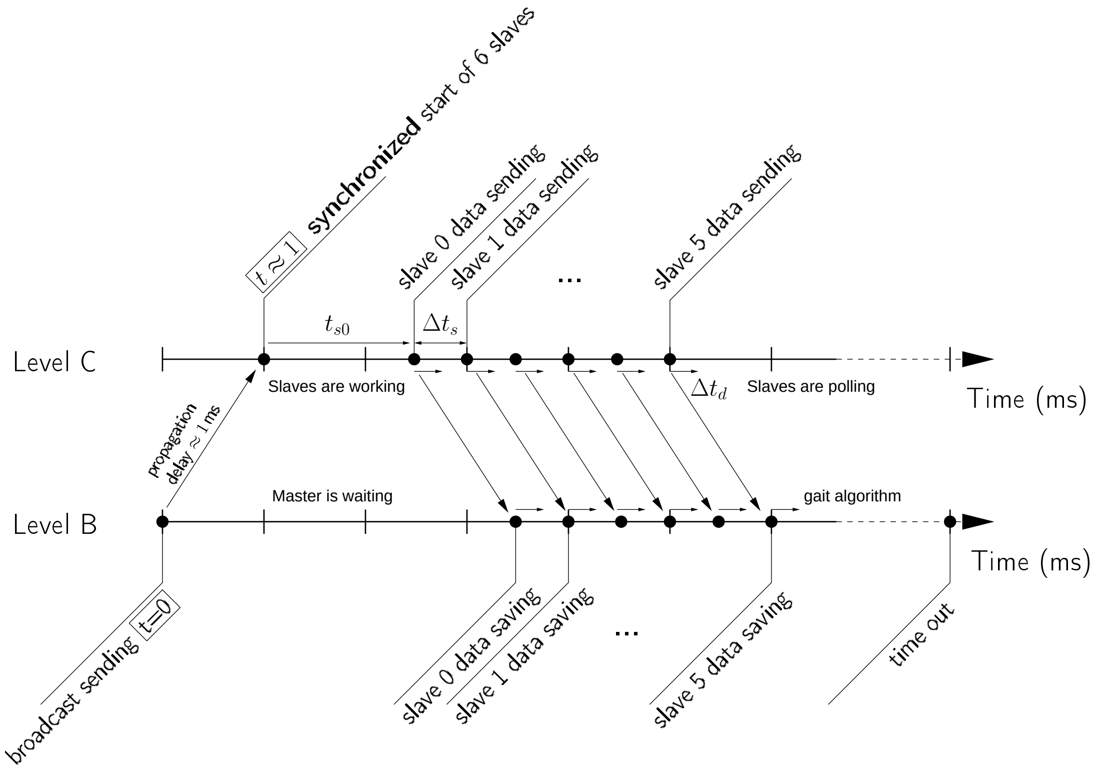


Figure 2.21: Communication between master and slaves

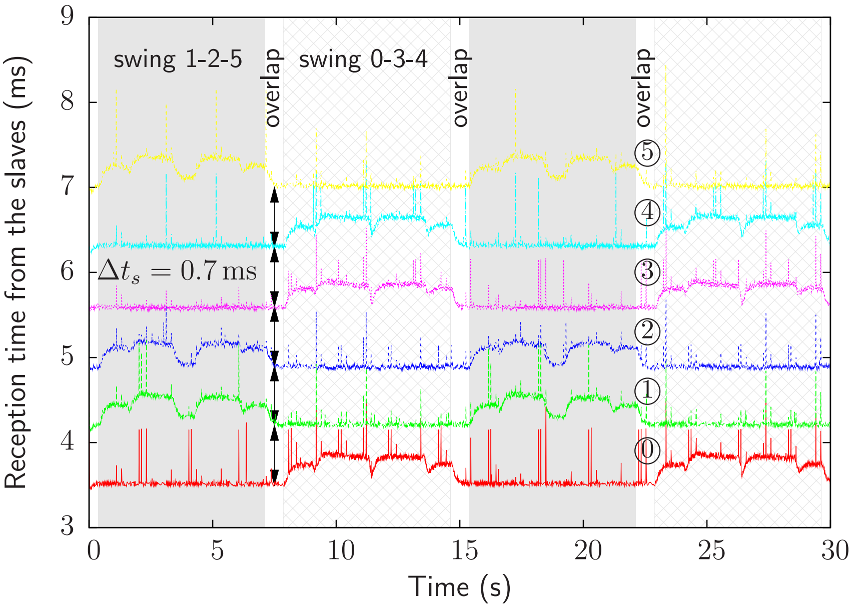


Figure 2.22: Timing for receptions of the slave data by the master

from slaves 0-3-4 (in support phase) are quite regular due to the low velocities of the actuators, meaning a weak CPU load. This phenomenom is inverted when legs 0-3-4 are swinging (hatched area). Note that during the *overlap* phases (transitions during which the six legs are on the ground - see Section 7.5.1), all the slaves undergo approximately the same CPU load. Results is that emission are even with a $\Delta t_s = 700\,\mu$s.

Measurements shown that delayed time Δt_d is less than $300\,\mu$s: therefore Δt_s has finally been fixed to $500\,\mu$s. The $700\,\mu$s of Fig. 2.22 having been imposed for the reader's convenience.

A last remark is about the noisy shape of the curves in Fig. 2.22. Because the used operating system is not "hard" real-time, other processes can take the priority, delaying the broadcast sending and the rest of the operations in the communication scheme. Most of the time, all the operations are done within 10 ms, as shown in Fig.2.22, where the latest data reception are under 9 ms. The control time slice is rarely exceeded. If necessary, the robot is stopped.

CHAPTER 3

Modeling of six-Legged robots

Apart from the dynamic model of the robot structure, some multiphysic elements are required to study walking machines. For example, walking gives rise to transitional loads on the feet, which have to be handled correctly by the actuator controllers. Actuation often consists in high-geared motors : consequently, joint friction should be modeled to have a correct estimation of power consumption.

The present chapter is devoted to the modeling method applied to AMRU5. Section 3.1 presents a state of the art about the modeling of walking machines, and outlines the contribution of the present work. Then, the modeling issues are stated and discussed in Section 3.2. The multibody framework `EasyDyn` is comprehensively presented in Section 3.3. Application to the AMRU5 hexapod is developed in Section 3.4. This chapter ends with Section 3.5 where an overview of the model specifications such as the number of configuration parameters, bodies and equations is given.

3.1 State of the art

In the case of walking robots, the simulation can help for three points:

1. the design of the structure;
2. the development of gait algorithms;
3. the gait optimization.

The designs of six-legged robots are very numerous, as evidenced in Chapter 2. A wide range of possibilities exist to create a hexapod. Simulations allow to explore original designs and their feasibility. Placement of the legs around the body can be rectangular or hexagonal, actuation can be electrical, pneumatical or hydraulical. Leg mechanisms are also quite different: some have closed- (AMRU5) or open-loop (SILO6) structure, or are biologically-inspired (AJAX). Actually, each walking structure is almost unique.

In this sense, dynamic simulation-based design could help in the development and the characterization of such machines.

Interaction of these vehicles with the environment is the second step of the design study. Dynamic simulation of legged vehicle began twenty five years ago, with Shih et al. [SHIH87]. They develop a simplified three dimensional dynamic model of the OSU hexapod, using a compliant joint model. Principle is that the parts of the robot are linked with a force element, in this case a three DOF spring-damper. The model is quite simplified, each leg is considered as one link, attached to the main body by a ball joint. The ground contact is also modeled by a compliant element. Results are shown for impact of legs on the ground, and agree quite well with the original measurements. One of the first general models dealing with realistic considerations like joint friction or soil interactions was developed for locomotion on natural terrain [MANK92]. Foot-soil interactions and joint friction, both experimentally identified, are included in the model. Simulations on flat settlement ("drop" of the robot on the ground and settlement of the ground contact forces at foot) and of one gait cycle are made to estimate the joint torques needed in the case of the *Ambler* hexapod. Establishment of other general models are found in literature [BARR98, GONZ06, GARC03, MAHA09].

Beside the global 3D modeling, some authors focus on the leg and its dynamics. Garcia et al. [GARC00, GARC03] underline that most authors neglect the leg parts in their model, without any real estimation of their contribution. Indeed, it is commonly admitted that high-geared actuators typically encountered in legged robots are very few sensitive to inertia, centrifugal and Coriolis forces. However, high-geared motors introduce flexibility, backlash and friction which should be modeled to get reliable simulations. They finally propose a method to determine the relevant dynamics influencing the joint of a leg mechanism, with an actuator model including friction losses. Silva et al. [SILV05] go further by involving a compliant behavior in the joints and the backlash occurring at teeth impact in the gearbox.

The second point concerns the development of gait algorithms. They were initially based on pure kinematic robot models [LEE88, SALM96] for classical gait approaches, or, more recently, for biologically-inspired approaches [FIGL07]. They just described the kinematics of the robot, and did not involve dynamic aspects. However, interaction with environment and actuator limitations have an important influence on robot control, and the scientific community began to involve them in the modeling. For example, capabilities in terms of mobility and agility are investigated in [BOWL06]. Bowling determines the maximum achievable accelerations of the main body, computed under non-slipping feet condition and without saturation of the actuators. In the same lineage, Silva et al. analyze the robot "locomobility" [SILV08]: this index, close to the manipulability ellipsoids given in robotic manipulators theory [SICI09], measures the capability of the body and the legs to change their position arbitrarily. Other motions, like turning, have been investigated. Tsuchiya et al. [TSUC06] use a simplified 2D dynamic model of a multi-legged robot to show the birth of a meandering motion for inadequate controller gains.

The third point relates to gait optimization for energy saving. Marhefka and Orin [MARH97] edit five rules for setting velocity, footholds, body height, duty factor and stroke to achieve a maximum energy efficiency during the gait. Although the hexapod structure and environment are no very detailed, the actuator model includes viscous friction, electrical circuits and load caused by the legs dynamics. Silva et al. [SILV08] propose, although 2D, a more detailed dynamic model of a hexapod with which they derive trends about the energy consumption by varying the gait duty cycle, the body height, the forward velocity and the maximum foot clearance. A simplified energy consumption model based on the DC motors equations is proposed by Gonzalez De Santos [GONZ09], where only the ground-contact forces during a tripod gait are computed and used to derive the actuators energy consumption. The robot is supposed to move at constant velocity, with constant stroke of the feet: the only parameter used for the energy minimization is the distance between the foot and the leg anchor point on the main body.

Following this brief review, we state that many models of legged machines exist. First conclusion is that they seem comprehensive enough to capture the relevant elements required for mechanical design. Secondly, gait algorithms can be developed with a kinematic model in a relative confidence, because hexapods are generally slow machines, and consequently inertial forces are not so important in the gait. For the third point concerning the gait optimization and energy expenditure, authors have deeply investigated the effects encountered in actuators, but results of simulations are not validated by experiments on the whole robot. In robot manipulators, comparisons between simulation and experiment exist [ARMS91, GROT02, GROT04, BONA06]. For legged machines, Guardabrazo et al. [GUAR03] show a good correspondence between model-based currents and measurements for the three joints of the *SILO4* leg performing a gait cycle on a treadmill. This work has been remarkably materialized in [GONZ06]. However, to the best of our knowledge, no author has compared the simulations results to the measurements for a complete hexapod actuated by DC motors and walking. Therefore, an important part of the present work is dedicated to the validation of a comprehensive dynamic model by comparing simulation results and experiments, in the case of a periodic gait. However, it is worth citing the work of Nair et al. [NAIR92]. They modeled comprehensively the *ASV* vehicle based on hydraulic actuators. They were able to reproduce the evolution of differential pressure in the hydraulic actuators in a very accurate way.

A huge number of tools exist to simulate robot dynamics [CORK96, SAMI03, ZLAJ08], some of them being even dedicated to legged machines [REIC00, SCHM03]. Choice has been made to use the `EasyDyn`[1] C++ open source library [VERL05b]. It has been developed for the simulation of systems represented by second-order (or first-order) differential equations and, more particularly, multibody systems. Even if the modeling is not straightforward as in commercial software, its flexibility is by far the most important asset which motivates our choice.

[1] http://mecara.fpms.ac.be/`EasyDyn`/

3.2 Modeling issues

Basically, the modeling of legged machines is composed of two parts. The first one concerns the dynamics of the robot motion: we chose a multibody approach which automatically takes into account the inertia and gravity forces. This model is completed by the relevant forces encountered by walking robots, that is to say:

- the ground contact forces, which are intermittent unilateral forces between the feet and the soil;
- the friction in the joints;
- the motor torques.

The second part of the model concerns dynamics and control of the actuators.

Before the modeling itself, a description of the `EasyDyn` framework is provided in Sect. 3.3.

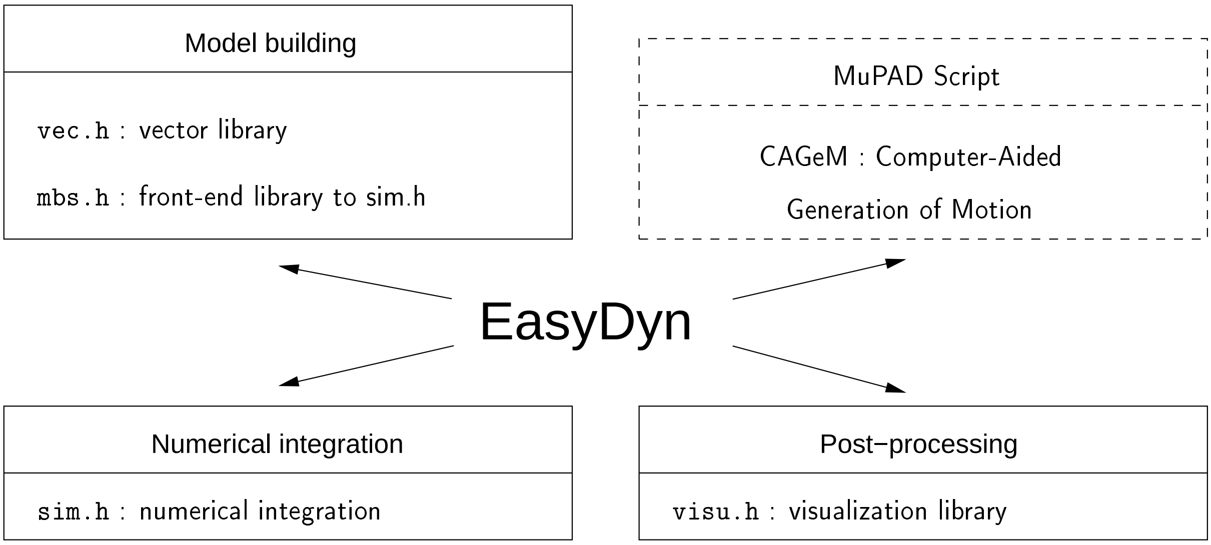

Model building
vec.h : vector library
mbs.h : front-end library to sim.h
MuPAD Script
CAGeM : Computer-Aided Generation of Motion
EasyDyn
Numerical integration
sim.h : numerical integration
Post–processing
visu.h : visualization library

Figure 3.1: `EasyDyn` framework

3.3 The `EasyDyn` framework

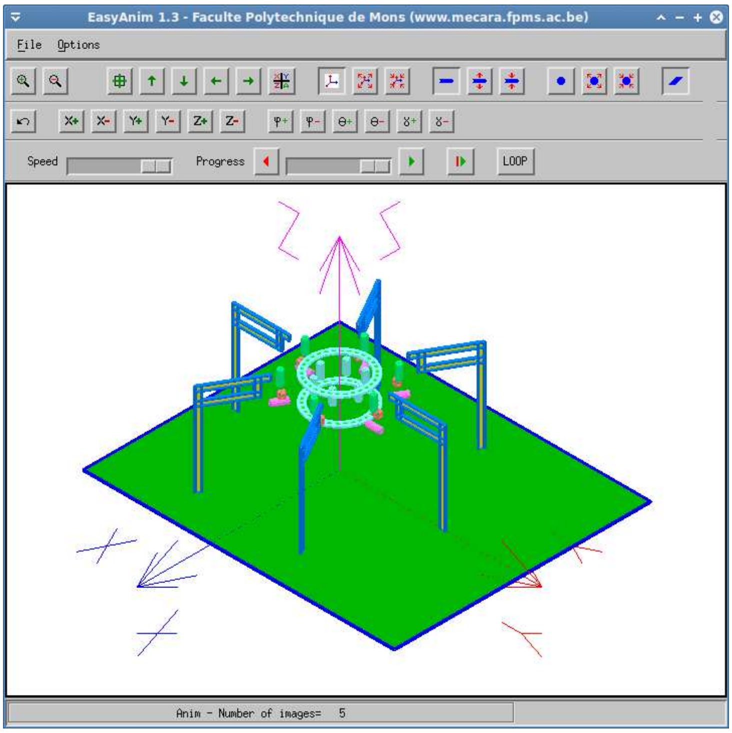
Figure 3.1 depicts the `EasyDyn` framework. It consists of four modules (`vec`, `sim`, `mbs`, `visu`) written in C++ , which allow to build and solve any multiphysic problem described by a set of first or second order differential equations.

3.3.1 General overview

The library is organized in four modules:

- `vec`: C++ classes relative to vector calculus (vector, rotation tensor, homogeneous transformation matrix, inertia tensor) and corresponding assignment methods and operators;
- `sim`: routines for integration of second-order differential equations, based on the Newmark method;
- `mbs`: front-end to `sim` building the equations of motion from kinematics and applied forces, according to a minimal coordinates approach;
- `visu`: easy creation of files describing scenes composed of moving 3D objects for further visualization.

Moreover, `EasyDyn` provides a tool, called CAGeM (Computer-Aided Generation of Motion), which automatically generates the complete kinematics (positions, velocities and accelerations) of a multibody system only from the position information. CAGeM is actually a MuPAD script, which uses the symbolic features of MuPAD to differentiate the position matrices. MuPAD is a computer algebraic system which was free for non commercial applications when `EasyDyn` was started. Unfortunately, it is no longer the case. Investigation with XCAS [PARI10] is presently in progress.


EasyAnim 1.3 - Faculte Polytechnique de Mons (www.mecara.fpms.ac.be)
File
Options
X+
X-
Y+
Y-
Z+
Z-
φ+
φ-
θ+
θ-
ȣ+
ȣ-
Speed
Progress
LOOP

Anim - Number of images= 5

Figure 3.2: `EasyAnim` window with the AMRU5 hexapod

The scenes created by the `visu` module can be read with the `EasyAnim`[2] animation software. A typical overview of the animation window is presented in Figure 3.2.

3.3.2 The `sim` module

The `sim` module solves ODE's of the form:

$$\boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, t) = \boldsymbol{0} \tag{3.1}$$

with:

- t the time;
- $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ the vectors gathering the state parameters of the system and their first and second time derivatives;
- $\boldsymbol{f}$ the vector gathering the residuals of the differential equations governing the behaviour of the system.

Note that first-order differential equations of the form $\boldsymbol{f}(\boldsymbol{y}, \dot{\boldsymbol{y}}, t) = \boldsymbol{0}$ can be solved by defining $\boldsymbol{y} = \dot{\boldsymbol{q}}$ and $\dot{\boldsymbol{y}} = \ddot{\boldsymbol{q}}$, $\boldsymbol{q}$ becoming meaningless.

Numerical integration is a step by step process consisting in determining the state a $t+h$ from the state at time t (and possibly some other previous states), h being the time step, chosen in order to get an accurate representation of the behavior of the system. If the

[2] http://mecara.fpms.ac.be/`EasyDyn`/

global system has n_{cp} parameters $\boldsymbol{q}$, then we have $3 \times n_{cp}$ unknowns $\boldsymbol{q}^{t+h}, \dot{\boldsymbol{q}}^{t+h}, \ddot{\boldsymbol{q}}^{t+h}$ and the same number of equations is required. The first n_{cp} equations are given by equations of motion under their residual form (Eq. (3.1)), noted as follows:

$$\boldsymbol{f}(\boldsymbol{q}^{t+h}, \dot{\boldsymbol{q}}^{t+h}, \ddot{\boldsymbol{q}}^{t+h}, t+h) = \boldsymbol{0} \tag{3.2}$$

The missing $2 \times n_{cp}$ equations express the dependency between configuration parameters $\boldsymbol{q}$ and their time derivatives:

$$\boldsymbol{q}^{t+h} = \boldsymbol{q}^{t} + \int_{t}^{t+h} \dot{\boldsymbol{q}}\ dt, \tag{3.3}$$

$$\dot{\boldsymbol{q}}^{t+h} = \dot{\boldsymbol{q}}^{t} + \int_{t}^{t+h} \ddot{\boldsymbol{q}}\ dt. \tag{3.4}$$

These last $2 \times n_{cp}$ equations are unusable as is and are replaced by so-called *integration formulas* which estimate the integral from an assumed evolution of the configuration during the time interval. They take the following general form

$$\dot{\boldsymbol{q}}^{t+h} = \check{\Lambda}(\dot{\boldsymbol{q}}^{\leq t}, \ddot{\boldsymbol{q}}^{\leq t}, \ddot{\boldsymbol{q}}^{t+h}) \tag{3.5}$$

$$\boldsymbol{q}^{t+h} = \Lambda(\boldsymbol{q}^{\leq t}, \dot{\boldsymbol{q}}^{\leq t}, \ddot{\boldsymbol{q}}^{\leq t}, \ddot{\boldsymbol{q}}^{t+h}). \tag{3.6}$$

Equations (3.5) and (3.6) indicate an implicit formulation if terms $\ddot{\boldsymbol{q}}^{t+h}$ are present, and explicit otherwise.

The sim module implements the Newmark integration formulas:

$$\boldsymbol{q}^{t+h} = \boldsymbol{q}^{t} + h\dot{\boldsymbol{q}}^{t} + (0.5-\beta)h^2\ddot{\boldsymbol{q}}^{t} + \beta h^2 \ddot{\boldsymbol{q}}^{t+h} \tag{3.7}$$

$$\dot{\boldsymbol{q}}^{t+h} = \dot{\boldsymbol{q}}^{t} + (1-\gamma)h\ddot{\boldsymbol{q}}^{t} + \gamma h \ddot{\boldsymbol{q}}^{t+h} \tag{3.8}$$

where h is the integration time step, and β, γ the method parameters. Equations (3.7) and (3.8) are a nice compromise between implementation issue, stability ($\gamma \leq 2\beta$ and $\gamma \geq 0.5$ ensure unconditional stability for linear systems) and accuracy (second order). Equations (3.7) and (3.8) are introduced in Eq. (3.2), resulting in a set of n_{cp} nonlinear equations in the unknowns $\ddot{\boldsymbol{q}}^{t+h}$ expressed as

$$\boldsymbol{F}(\ddot{\boldsymbol{q}}^{t+h}) = \boldsymbol{0}. \tag{3.9}$$

Resolution is performed with a Newton-Raphson algorithm, where acceleration at time $t+h$ is computed after several iterations. For iteration n, we have:

$$\ddot{\boldsymbol{q}}^{t+h,n} = \ddot{\boldsymbol{q}}^{t+h,n-1} - \boldsymbol{J}^{-1} \cdot \boldsymbol{f}(\boldsymbol{q}^{t+h,n-1}, \dot{\boldsymbol{q}}^{t+h,n-1}, \ddot{\boldsymbol{q}}^{t+h,n-1}, t+h) \tag{3.10}$$

where $\boldsymbol{J}$ is the jacobian matrix [HAIR91] (also called *iteration matrix*) of the equations $\boldsymbol{F}$ with respect to the unknowns $\ddot{\boldsymbol{q}}^{t+h}$. Term of line i and column j of this matrix is written:

$$\boldsymbol{J}_{ij} = \frac{\partial \boldsymbol{F}_i}{\partial \ddot{\boldsymbol{q}}_j^{t+h}} = \frac{\partial \boldsymbol{f}_i}{\partial \ddot{\boldsymbol{q}}_j} + \frac{\partial \boldsymbol{f}_i}{\partial \dot{\boldsymbol{q}}_j} \cdot \frac{\partial \check{\Lambda}}{\partial \ddot{q}^{t+h}} + \frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{q}_j} \cdot \frac{\partial \Lambda}{\partial \ddot{q}^{t+h}} \tag{3.11}$$

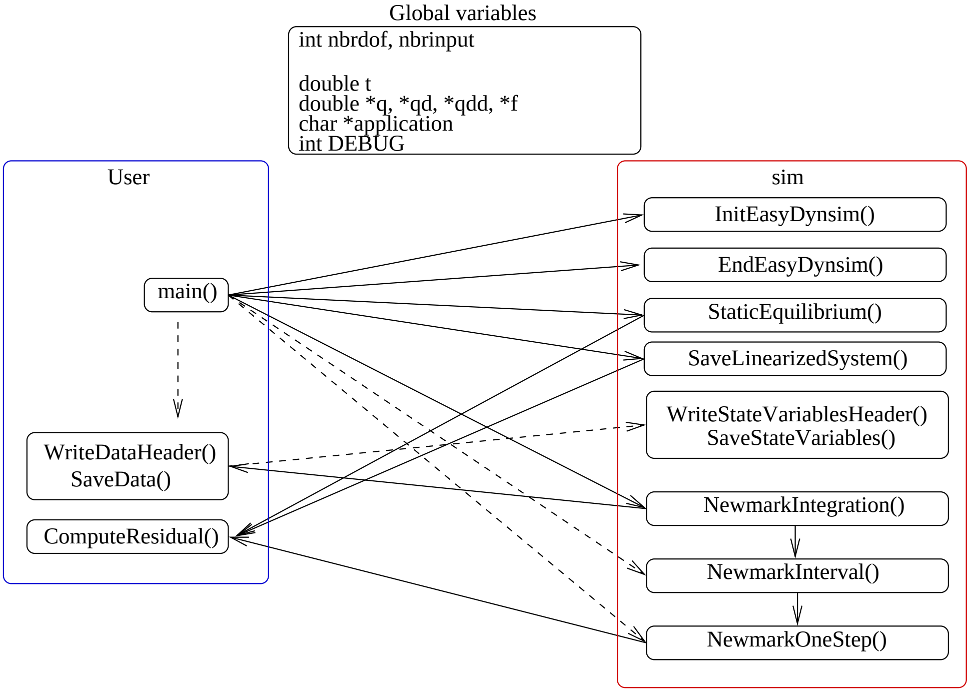


Figure 3.3: Structure of the `sim` module

If we introduce the tangent stiffness matrix $\boldsymbol{KT}$ and the tangent damping matrix $\boldsymbol{CT}$, defined by

$$\boldsymbol{KT}_{ij} = \frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{q}_j} \qquad \boldsymbol{CT}_{ij} = \frac{\partial \boldsymbol{f}_i}{\partial \dot{\boldsymbol{q}}_j} \tag{3.12}$$

the iteration matrix $\boldsymbol{J}$ can be rewritten as

$$\boldsymbol{J}_{ij} = \boldsymbol{M}_{ij} + \boldsymbol{CT}_{ij} \cdot \frac{\partial \check{\Lambda}}{\partial \ddot{q}^{t+h}} + \boldsymbol{KT}_{ij} \cdot \frac{\partial \Lambda}{\partial \ddot{q}^{t+h}} \tag{3.13}$$

In the case of the Newmark scheme, the iteration matrix is given by

$$\boldsymbol{J} = \boldsymbol{M} + \boldsymbol{CT} \cdot \gamma h + \boldsymbol{KT} \cdot \beta h^2 \tag{3.14}$$

Implementation

An overview of the `sim` module is shown in Fig. 3.3. Practically, the user just has to give:

- The equations of the system under residual form (Eq. (3.1)) in the `ComputeResidual()` routine

- The saving routine, `SaveData()`, and the routine `WriteDataHeader()` writing the header of the result file which contains the names of the saved variables. The module also comes with ready-to-use routines `WriteStateVariablesHeader()` and `SaveStateVariables()` which are configured to save time and state variables $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ and possibly the system inputs.

- The main() function, which typically contains the following:
  1. The number of configuration parameters nbrdof and possibly of inputs nbrinput.
  2. The memory allocation function InitEasyDynsim().
  3. The call to NewmarkIntegration() routine, which performs the integration up to final time, and save results at regular time intervals after which the states are saved. These intervals are specified by the user, as well as the maximum integration time step. Practically, each time interval is performed by calling NewmarkInterval() which itself calls NewmarkOneStep() (resolution of Eq. 3.10) as many times as necessary, adjusting the time step to keep the integration error below a given tolerance:

$$\epsilon = \frac{h^2}{12}\sqrt{\frac{\sum_{i=1}^{n_{cp}}(\ddot{q_i}^{t+h} - \ddot{q_i}^{t})^2}{n_{cp}}} \tag{3.15}$$

  4. The memory deallocation routine EndEasyDynsim().

The Jacobian matrix of Eq. (3.13) is estimated column by column with a numerical derivation. The method consists in computing the difference between residuals with current values $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$, and the residuals computed by these values incremented by α, $\alpha\gamma h$ and $\alpha\beta h^2$ respectively, where α is a parameter fixed in EasyDyn to obtain a satisfying convergence.

Finally, some extra tools for system analysis are provided. The first one concerns the research of static equilibrium with StaticEquilibrium(). The implemented method is the finite displacement approach: in other words, the system is abandoned to go to its natural equilibrium position. The resolution follows the Newmark integration scheme in which velocities are fixed to zero.

The second tool is used for system linearization. If only small perturbations are considered with respect to a given configuration, the equations of motion can be linearized. Such a linearization will be used in Chapter 4 for the design of the joint controllers. For a reference state of the system denoted 0, residual equations (3.1) are linearized according to:

$$\left.\frac{\partial \boldsymbol{f}}{\partial \ddot{\boldsymbol{q}}}\right|_0 \cdot \Delta\ddot{\boldsymbol{q}} + \left.\frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{q}}}\right|_0 \cdot \Delta\dot{\boldsymbol{q}} + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}\right|_0 \cdot \Delta\boldsymbol{q} = -\left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_0 \cdot \boldsymbol{u} \tag{3.16}$$

or

$$\boldsymbol{M} \cdot \Delta\ddot{\boldsymbol{q}} + \boldsymbol{CT} \cdot \Delta\dot{\boldsymbol{q}} + \boldsymbol{KT} \cdot \Delta\boldsymbol{q} = \boldsymbol{F} \cdot \boldsymbol{u} \tag{3.17}$$

where $\boldsymbol{F}$ is called the influence matrix. A numerical derivation is performed separately on $\boldsymbol{q}, \dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ and input $\boldsymbol{u}$ to deduce tangent stiffness and damping matrices, the mass matrix and the input contribution matrix respectively.

3.3.3 The `mbs` module

For multibody systems, the construction of residual equations (3.1) is very tedious to do by hand, and must be computer-aided. The aim of the `mbs` module is to supply the residual equations of motion of the multibody system to the `sim` module so that Newmark integration can be performed. The minimal coordinates method is implemented in `mbs`: in this case, there are as many equations as configuration parameters n_{cp}, because kinematic constraints are hidden in the kinematic description [ANAN91]. The drawback is that all the kinematic analysis must be performed expressly for each system and can lead to very complex expressions of bodies velocities and accelerations. In the following, conventions for kinematics and forces are detailed before the formulation of the equations of motion, and the implementation of the `mbs` module in `EasyDyn`.

Kinematics

A convenient way to describe the spatial position and orientation of body j with respect to body i is throught the homogeneous transformation matrix (HTM) formalism [PAUL81]:

$$\boldsymbol{T}_{i,j} = \begin{pmatrix} \boldsymbol{R}_{i,j} & \{\boldsymbol{p}_{G_j/i}\}_i \\ 0\ 0\ 0 & 1 \end{pmatrix} \tag{3.18}$$

where

- $\{\boldsymbol{p}_{G_j/i}\}_i$ is the coordinate vector of point G_j with respect to frame i, projected in frame i;
- $\boldsymbol{R}_{i,j}$ is the rotation tensor describing the orientation of frame j with respect to frame i.

From this, the coordinate vector of point P with respect to frame j, $\boldsymbol{p}_{P/j}$, is given in frame i by the homogeneous transformation:

$$\begin{pmatrix} \{\boldsymbol{p}_{P/i}\}_i \\ 1 \end{pmatrix} = \boldsymbol{T}_{i,j} \cdot \begin{pmatrix} \{\boldsymbol{p}_{P/j}\}_j \\ 1 \end{pmatrix} \tag{3.19}$$

which is equivalent to:

$$\{\boldsymbol{p}_{P/i}\}_i = \{\boldsymbol{p}_{G_j/i}\}_i + \boldsymbol{R}_{i,j} \cdot \{\boldsymbol{p}_{P/j}\}_j \tag{3.20}$$

For the sake of simplicity, we will introduce the $\circ$ operator such that

$$\{\boldsymbol{p}_{P/i}\}_i = \boldsymbol{T}_{i,j} \circ \{\boldsymbol{p}_{P/j}\}_j \tag{3.21}$$

One interesting property is that bodies description can be done recursively, relatively to previously defined bodies, and not systematically in the global frame. Formally, HTM between bodies k and i can be expressed through an intermediary body j:

$$\boldsymbol{T}_{i,k} = \boldsymbol{T}_{i,j} \cdot \boldsymbol{T}_{j,k} \tag{3.22}$$

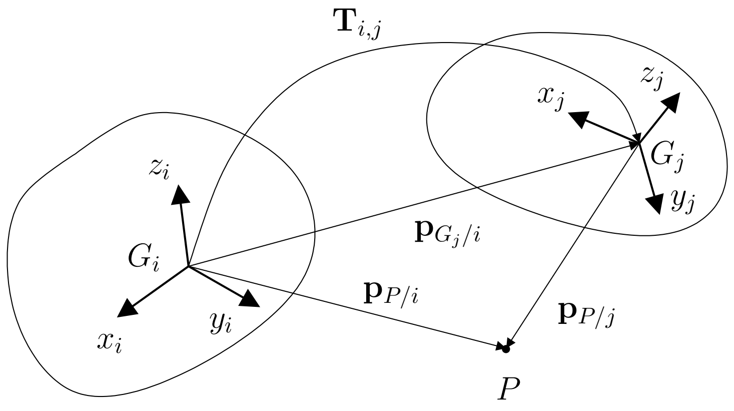


Figure 3.4: Homogeneous transformation matrices between two frames

A second nice property is that the absolute velocities and accelerations of body i in the global reference frame, $(\{\boldsymbol{v}_{i/0}\}_0, \{\boldsymbol{a}_{i/0}\}_0)$ and $(\{\boldsymbol{\omega}_{i/0}\}_0, \{\dot{\boldsymbol{\omega}}_{i/0}\}_0)$ respectively, can be obtained from differentiation of the body position and orientation. The following paragraphs describe the computation of linear and rotational velocities and acceleration.

For the reader's convenience, note that:

- the vector coordinate of body j center of gravity $\boldsymbol{p}_{G_j/i}$ will be noted $\boldsymbol{p}_{j/i}$: the point will be specified only if it differs from the center of gravity;
- the observer frame will be specified only if it is different from zero: $\boldsymbol{p}_{i/0}$ is thus simplified as $\boldsymbol{p}_i$.

Linear velocity and acceleration In a minimal coordinates approach, the body i HTM relatively to the global reference frame is expressed in terms of the configuration parameters:

$$\boldsymbol{T}_{0,i}(\boldsymbol{q}) = \begin{pmatrix} \boldsymbol{R}_{0,i}(\boldsymbol{q}) & \{\boldsymbol{p}_i(\boldsymbol{q})\}_0 \\ 0\ 0\ 0 & 1 \end{pmatrix} : \tag{3.23}$$

so that its linear velocity $\boldsymbol{v}_i$ (3×1) is expressed:

$$\boldsymbol{v}_i = \frac{d\boldsymbol{p}_i}{dt} = \sum_{j=1}^{n_{cp}} \frac{\partial \boldsymbol{p}_i}{\partial q_j} \cdot \dot{q}_j = \sum_{j=1}^{n_{cp}} \boldsymbol{d}_{ij} \cdot \dot{q}_j. \tag{3.24}$$

In Eq. (3.24), $\boldsymbol{d}_{ij}$ represents the partial contribution of parameter q_j in the translational velocity of frame i. The latter are gathered in the so-called velocity matrix $\boldsymbol{J}_{Si}$ $(3 \times n_{cp})$

$$[\boldsymbol{J}_{Si}]_0 = \left(\{\boldsymbol{d}_{i1}\}_0 \quad \{\boldsymbol{d}_{i2}\}_0 \quad \dots \quad \{\boldsymbol{d}_{in_{cp}}\}_0\right). \tag{3.25}$$

Body i velocity in the global frame is thus computed by:

$$\{\boldsymbol{v}_i\}_0 = [\boldsymbol{J}_{Si}]_0 \cdot \dot{\boldsymbol{q}}. \tag{3.26}$$

One further derivation leads to the linear acceleration:

$$\{\boldsymbol{a}_i\}_0 = \frac{d\{\boldsymbol{v}_i\}_0}{dt} = [\boldsymbol{J}_{Si}]_0 \cdot \ddot{\boldsymbol{q}} + \left[\dot{\boldsymbol{J}}_{Si}\right]_0 \cdot \dot{\boldsymbol{q}}. \tag{3.27}$$

Rotational velocity and acceleration It is well known that the elements of the rotational velocity vector of body i $\boldsymbol{\omega}_i$ are given, in the absolute reference frame, by the following relationship:

$$[\tilde{\boldsymbol{\omega}}_i]_0 = \dot{\boldsymbol{R}}_{0,i} \cdot \boldsymbol{R}_{0,i}^T \tag{3.28}$$

where operator $\tilde{\boldsymbol{a}}$ is defined by:

$$\tilde{\boldsymbol{a}} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix} \tag{3.29}$$

Further developments of this equation lead to:

$$[\tilde{\boldsymbol{\omega}}_i]_0 = \dot{\boldsymbol{R}}_{0,i} \cdot \boldsymbol{R}_{0,i}^T = \sum_{j=1}^{n_{cp}} \frac{\partial \boldsymbol{R}_{0,i}}{\partial q_j} \cdot \boldsymbol{R}_{0,i}^T \cdot \dot{q}_j = \sum_{j=1}^{n_{cp}} \tilde{\boldsymbol{\delta}}_{ij} \cdot \dot{q}_j \tag{3.30}$$

$$\boldsymbol{\omega}_i = \sum_{j=1}^{n_{cp}} \boldsymbol{\delta}_{ij} \cdot \dot{q}_j \tag{3.31}$$

where $\boldsymbol{\delta}_{ij}$ is the partial contribution of parameter q_j in the rotational velocity of frame i. They are gathered in the so-called velocity transformation matrix $\boldsymbol{J}_{\omega i}$ for a rotational motion

$$[\boldsymbol{J}_{\omega i}]_0 = \left(\{\boldsymbol{\delta}_{i1}\}_0 \quad \{\boldsymbol{\delta}_{i2}\}_0 \quad \ldots \quad \{\boldsymbol{\delta}_{in_{cp}}\}_0 \right). \tag{3.32}$$

Body i angular velocity in the global frame is thus computed by:

$$\{\boldsymbol{\omega}_i\}_0 = [\boldsymbol{J}_{\omega i}]_0 \cdot \dot{\boldsymbol{q}}. \tag{3.33}$$

One further derivation leads to the angular acceleration:

$$\{\dot{\boldsymbol{\omega}}_i\}_0 = \frac{d\{\boldsymbol{\omega}_i\}_0}{dt} = [\boldsymbol{J}_{\omega i}]_0 \cdot \ddot{\boldsymbol{q}} + \left[\dot{\boldsymbol{J}}_{\omega i}\right]_0 \cdot \dot{\boldsymbol{q}}. \tag{3.34}$$

Applied forces

The minimal coordinates approach implies that only the applied forces have to be expressed, because joints forces don't produce any work and would disappear from the equations of motion.

We will consider here the forces from their resultant wrench at the principal frame (center of gravity) of the body they act on. If n_f forces and n_m moments exert on body i, the resulting force $\boldsymbol{R}_i$ is given by:

$$\boldsymbol{R}_i = \sum_{j=1}^{n_f} \boldsymbol{F}_j \tag{3.35}$$

and the resulting moment $\boldsymbol{\mathcal{M}}_i$:

$$\boldsymbol{\mathcal{M}}_i = \sum_{k=1}^{n_m} \boldsymbol{\mathcal{M}}_k + \sum_{j=1}^{n_f} \boldsymbol{p}_{P_j/i} \times \boldsymbol{F}_j \tag{3.36}$$

The dynamic analysis of the system needs the expression of the resultant wrench of all forces acting on each body, in the axes of the global reference frame, in terms of the configuration parameters, their first time derivative, and time, that is to say, for each body i, $\{\boldsymbol{R}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)\}_0$ and $\{\boldsymbol{\mathcal{M}}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)\}_0$. In the case of walking robots, gravity, friction, ground contact forces and torques applied by the actuators will be considered.

Formulation of the equations of motion

Equations of motion are written in minimal coordinates, following the virtual powers principle, which states that *the power developed by all forces acting on a system, including applied, joint and inertia forces must vanish, for any virtual motion of the system.* Moreover, if a kinematically admissible virtual motion is considered, the joint forces do not develop any power and only the inertia (including Coriolis and centrifugal forces) and applied forces will have to be taken into account. Formally, for a system with n_b bodies and n_{cp} configuration parameters gathered in array $\boldsymbol{q}$ ($n_{cp} \times 1$), the n_{cp} equations can be built as follows:

$$\sum_{i=1}^{n_b} [\boldsymbol{d}_{ij} \cdot (\boldsymbol{R}_i - m_i \boldsymbol{a}_i) + \boldsymbol{\delta}_{ij} \cdot (\boldsymbol{\mathcal{M}}_i - \boldsymbol{\Phi}_{G_i} \dot{\boldsymbol{\omega}}_i - \boldsymbol{\omega}_i \times \boldsymbol{\Phi}_{G_i} \boldsymbol{\omega}_i)] = 0 \quad j = 1, ..., n_{cp} \tag{3.37}$$

with

1. m_i and $\boldsymbol{\Phi}_{G_i}$ the mass and the central inertia tensor of body i;
2. $\boldsymbol{R}_i$ and $\boldsymbol{\mathcal{M}}_i$ the resultant force and moment, at the center of gravity, of all the applied forces exerted on body i;
3. $\boldsymbol{a}_i$, $\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$ the translational acceleration, the rotational velocity and the rotational acceleration of body i, respectively;
4. $\boldsymbol{d}_{ij}$ and $\boldsymbol{\delta}_{ij}$ the partial velocity contributions.

Projection of Eq. (3.37) in the absolute reference frame leads to the classical form of the n_{cp} equations of motion:

$$\boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{g}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) \tag{3.38}$$

with

- $\boldsymbol{M}(\boldsymbol{q})$ ($n_{cp} \times n_{cp}$) the mass matrix defined by

$$\boldsymbol{M}(\boldsymbol{q}) = \textstyle\sum_{i=1}^{n_B} [m_i \, [\boldsymbol{J}_{Si}]_0^T \cdot [\boldsymbol{J}_{Si}]_0 \quad + [\boldsymbol{J}_{\omega i}]_0^T \cdot \{\boldsymbol{\Phi}_{G_i}\}_0 \cdot [\boldsymbol{J}_{\omega i}]_0]; \tag{3.39}$$

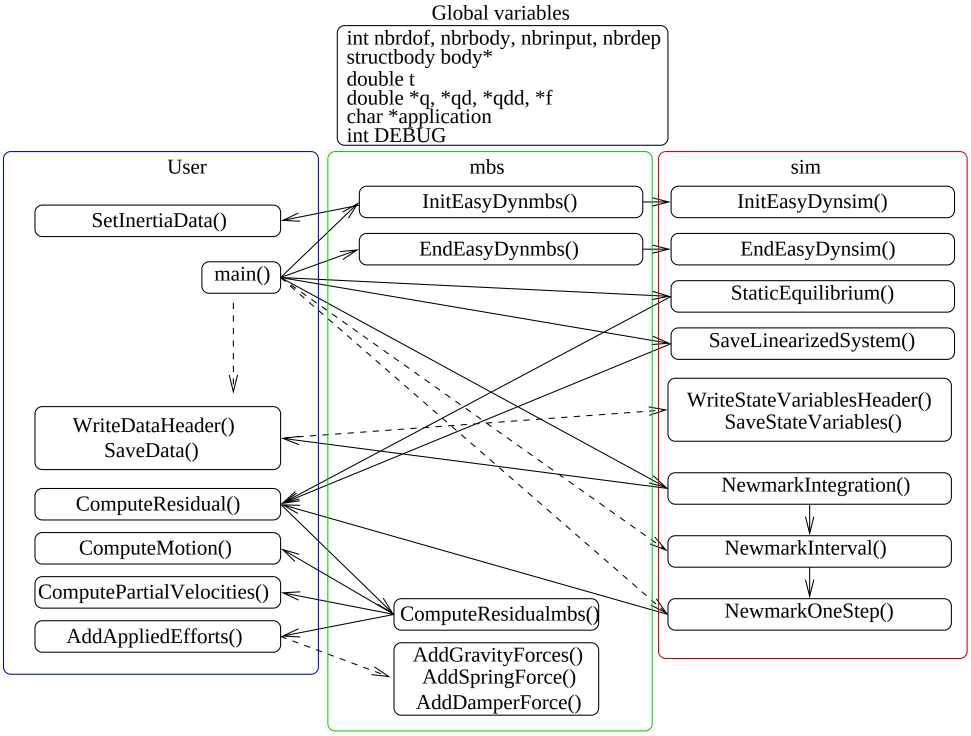


Figure 3.5: Structure of the `mbs` module and interactions with `sim`

- $\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}})$ $(n_{cp}\times 1)$ the contribution of centrifugal and Coriolis forces given by

$$\begin{aligned}\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \ & \textstyle\sum_{i=1}^{n_B}[(m_i\,[\boldsymbol{J}_{Si}]_0^T\cdot\left[\dot{\boldsymbol{J}}_{Si}\right]_0+[\boldsymbol{J}_{\omega i}]_0^T\cdot\{\boldsymbol{\Phi}_{G_i}\}_0\cdot\left[\dot{\boldsymbol{J}}_{\omega i}\right]_0)\cdot\dot{\boldsymbol{q}}\\ & +[\boldsymbol{J}_{\omega i}]_0^T\cdot\{\boldsymbol{\omega_i}\times\boldsymbol{\Phi}_{G_i}\boldsymbol{\omega}_i\}_0];\end{aligned} \tag{3.40}$$

- $\boldsymbol{g}(\boldsymbol{q},\dot{\boldsymbol{q}},t)$ $(n_{cp}\times 1)$ the contribution of the applied forces described by

$$\boldsymbol{g}(\boldsymbol{q},\dot{\boldsymbol{q}},t)=\textstyle\sum_{i=1}^{n_B}[[\boldsymbol{J}_{Si}]_0^T\cdot\{\boldsymbol{R}_i\}_0+[\boldsymbol{J}_{\omega i}]_0^T\cdot\{\boldsymbol{\mathcal{M}}_i\}_0]. \tag{3.41}$$

Implementation

An overview of the `mbs` module and its interaction with `sim` is shown in Fig. 3.5. The user has to give the same parameters as for `sim`, plus:

- The number of bodies `nbrbody`.
- The number of dependent parameters `nbrdep`.
- The routine `SetInertiaData()` which defines masses and inertia tensors of the bodies.
- The routine `ComputeMotion()` which includes the kinematics of the whole system expressed in terms of configuration parameters. Practically, the HTM of each body $\boldsymbol{T}_{0,i}$ with respect to the global frame and velocities and accelerations projected in the global reference frame, (that is to say $(\{\boldsymbol{v}_i\}_0,\{\boldsymbol{a}_i\}_0)$ and $(\{\boldsymbol{\omega}_i\}_0,\{\dot{\boldsymbol{\omega}}_i\}_0)$) are expressed inside this function.

- The routine ComputePartialVelocities() (optional) inside which the partial velocities of Eq. (3.37) are described in the global frame; if it is not specified by the user, the partial velocities are estimated by a numerical derivation;

- The routine AddAppliedEfforts() where the resulting forces and moments acting on each body (Eqs. (3.35) and (3.36)) must be given. To help the user, some routines are provided for classical force elements like springs, dampers, tyres, ...

The main routine must call the functions for memory allocation and deallocation (InitEasyDynmbs() and EndEasyDynmbs()), and as previously typically calls NewmarkIntegration() for time integration.

The ComputeResidual() routine of sim calls ComputeResidualmbs(), which builds the equations of motion (3.37) from the results provided by ComputeMotion() and AddAppliedEfforts().

It is interesting to notice that the dimension of the system solved by the sim module can be equal or greater than the number of equations of motion of the multibody system. In other words, any other physical system whose dynamics is described by first- or second-order differential equations can be inserted in addition to ComputeResidualmbs() inside the ComputeResidual() routine. This has been done for the robot, by adding controllers, actuators and friction models (see Sect. 3.4).

3.3.4 Practical use

Practically, the user has a computer-aided tool called CAGeM to generate its C source code. He has to fulfill an ASCII file with:

- the inertia properties;

- the number of configuration parameters;

- the number of dependent parameters;

- the number of bodies;

- the number of inputs;

- the kinematics of the whole system, in terms of configuration and dependent parameters; the kinematic description is made at position level only, by means of elementary homogeneous transformation matrices (see Appendix A);

- the expression of the prospective dependent parameters;

- the initial conditions.

This file is read by the CAGeM script under MuPAD, which generates a ready to compile C++ file, including the complete kinematics of the system (positions, velocities and accelerations of all the bodies) obtained by symbolic differentiation of the homogeneous transformation matrices. This file is then completed by the user as desired.

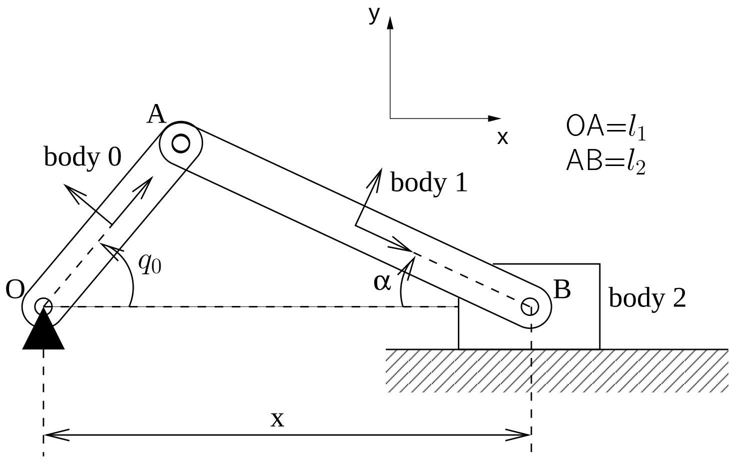


Figure 3.6: Slider-scrank mechanism

Enhancement of the kinematic calculation time

The modeling in minimal coordinates with `EasyDyn` initially required long calculation times for kinematics, especially for three points:

1. the partial velocities of Eq. (3.37) were estimated by a numerical derivative on the global kinematic function `ComputeMotion()`. Many terms of the partial velocities can be null, but are anyway computed;

2. the description of the bodies with respect to the global reference frame becomes quite long for systems with a large number of bodies, resulting in an increasing time of velocities and accelerations calculations by Eqs. (3.26), (3.27), (3.33) and (3.34);

3. the kinematic configuration can be a complex expression of configuration parameters, involving trigonometric functions, long to derive and compute.

For point 1), the CAGeM module has been adapted so that the direct expressions of partial velocities are given in the C++ source code, avoiding the numerical derivation. For point 2), relative motion has been implemented in `EasyDyn`, in the same idea as multibody modeling with relative coordinates. Velocities and accelerations are computed with respect to a reference body instead of the global coordinate system, and then they are recomposed from velocities and accelerations of this reference body. For point 3), the complex combination of configuration parameters is defined through *dependent parameters*, which are computed only once at each time step of the simulation, instead of reevaluating it many times in the kinematic function.

To illustrate the use of dependent parameters, consider the slider-crank mechanism of Fig. 3.6, whose configuration parameter is q_0, the angle of the crank. If we define intermediary variables α and x as:

$$\alpha = \arcsin\left(\frac{l_1 \sin(q_0)}{l_2}\right) \tag{3.42}$$

$$x = l_1 \cos(q_0) + l_2 \cos(q_1), \tag{3.43}$$

the kinematic description in terms of HTM becomes:

- for the crank :
$$\boldsymbol{T}_{0,crank} = \boldsymbol{T}^{R_z}(q_0) \cdot \boldsymbol{T}^{d}(l_1/2, 0, 0); \tag{3.44}$$
- for the connecting rod :
$$\boldsymbol{T}_{0,connrod} = \boldsymbol{T}^{d}(x, 0, 0) \cdot \boldsymbol{T}^{R_z}(-\alpha) \cdot \boldsymbol{T}^{d}(-l_2/2, 0, 0); \tag{3.45}$$
- for the slider :
$$\boldsymbol{T}_{0,slider} = \boldsymbol{T}^{d}(x, 0, 0), \tag{3.46}$$

where $\boldsymbol{T}^{d}(\cdot)$ represents a 3D translational displacement, and $\boldsymbol{T}^{R_i}(\cdot)$ a rotation about i axis.

Without dependent parameters, α and x are replaced everywhere by their expression in the symbolic engine (MuPAD/XCAS) leading, for example, to the following expression for the velocity of center of gravity of body 1:

$$\{\boldsymbol{v}_{G_1}\}_0 = \begin{pmatrix} \dfrac{\frac{-l_1^2 \dot{q}_0 \sin(2q_0)}{2} - 2l_1 l_2 \dot{q}_0 \sin(q_0) \cdot \sqrt{\frac{l_2^2 - \frac{l_1^2}{2} + \frac{l_1^2 \cos(2q_0)}{2}}{l_2^2}}}{2l_2 \cdot \sqrt{\frac{l_2^2 - \frac{l_1^2}{2} + \frac{l_1^2 \cos(2q_0)}{2}}{l_2^2}}} \\ \frac{l_1 \dot{q}_0 \cos(q_0)}{2} \\ 0 \end{pmatrix}. \tag{3.47}$$

If dependent parameters are specified such as $p_0 = \alpha$ and $p_1 = x$, the velocity becomes simpler:

$$\{\boldsymbol{v}_{G_1}\}_0 = \begin{pmatrix} \dot{p}_1 + \frac{l_2 \dot{p}_0 \sin(p_0)}{2} \\ \frac{l_2 \dot{p}_0 \cos(p_0)}{2} \\ 0 \end{pmatrix}. \tag{3.48}$$

Evaluation of Eq. (3.48) is by far faster than that of Eq. (3.47).

The first time derivative of dependent parameters are:

$$\dot{p}_0 = \frac{l_1 \dot{q}_0 \cos(q_0)}{l_2 \cdot \sqrt{\frac{l_2^2 - \frac{l_1^2}{2} + \frac{l_1^2 \cos(2q_0)}{2}}{l_2^2}}} \tag{3.49}$$

$$\dot{p}_1 = \frac{-\frac{l_1^2 \dot{q}_0 \sin(2q_0)}{2} - l_1 l_2 \dot{q}_0 \sin(q_0) \cdot \sqrt{\frac{l_2^2 - \frac{l_1^2}{2} + \frac{l_1^2 \cos(2q_0)}{2}}{l_2^2}}}{\sqrt{\frac{l_2^2 - \frac{l_1^2}{2} + \frac{l_1^2 \cos(2q_0)}{2}}{l_2^2}}} \tag{3.50}$$

Even if Eqs. (3.49) and (3.50) are also long, they are computed only once during the kinematic calculation, while without them α and x are always completely developed. The effect is still more impressive on acceleration. For example, the expression of $a_{G_1,x}$ is 1710 character long, without the use of dependent parameters, and only 78 otherwise.

Reduction type	q_r	q_v	q_h
Gearbox	531:1	33:1	28:1
Sprocket chain	38:12	1:1	1:1
Ball-screw	-	2094:1 (rad/m)	2094:1 (rad/m)
Global reduction n	1682:1	69115:1 (rad/m)	58643:1 (rad/m)

Table 3.1: Reduction ratios of transmission elements

3.4 Application to AMRU5

The AMRU5 model consists of 49 bodies and 24 degrees of freedom (DOF):

- main robot body: 1 free body with 6 DOF;
- leg chassis: 1 body per leg, linked to the main body by a revolute joint, and whose position depends on the rotational DOF q_r;
- motors shafts: 3 bodies per leg; they have a revolution motion directly related to the rotational, vertical and horizontal DOF through the reduction ratios given in Table 3.1;
- pantograph mechanism: 4 bodies per leg, linked by revolute joints, and forming a closed-loop structure depending on q_v and q_h (Fig. 3.8).

3.4.1 Kinematic description

Degrees of freedom are gathered in the vector $\boldsymbol{q}$ according to

$$\boldsymbol{q} = \begin{pmatrix} \boldsymbol{q}^b \\ \boldsymbol{q}^0 \\ \dots \\ \boldsymbol{q}^5 \end{pmatrix} = \begin{pmatrix} (X \ Y \ Z \ \Phi \ \Theta \ \Psi)^T \\ (q_r^0 \ q_v^0 \ q_h^0)^T \\ \dots \\ (q_r^5 \ q_v^5 \ q_h^5)^T \end{pmatrix} \tag{3.51}$$

where superscript b and 1...5 denote the main body configuration parameters, and the leg identification number respectively. Figure 3.7 illustrates the modeling of the main body and a leg. The pantograph mechanism is included in the π plane, and the center of gravity of the leg chassis is in π'.

Position of the main robot body in the global reference frame is expressed by successive elementary operations of translation and rotation. The resulting HTM of the body $\boldsymbol{T}_b$ is written

$$\boldsymbol{T}_{0,b} = \begin{pmatrix} \boldsymbol{R}_{0,b} & \{\boldsymbol{p}_b\}_0 \\ 0\ 0\ 0 & 1 \end{pmatrix} = \boldsymbol{T}^d(X,Y,Z) \cdot \boldsymbol{T}^{R_z}(\Phi) \cdot \boldsymbol{T}^{R_y}(\Theta) \cdot \boldsymbol{T}^{R_x}(\Psi) \tag{3.52}$$

The form of elementary HTM is given in Appendix A. The orientations are given according to the yaw (Φ), pitch (Θ) and roll (Ψ) angles. To make the description of the pantograph

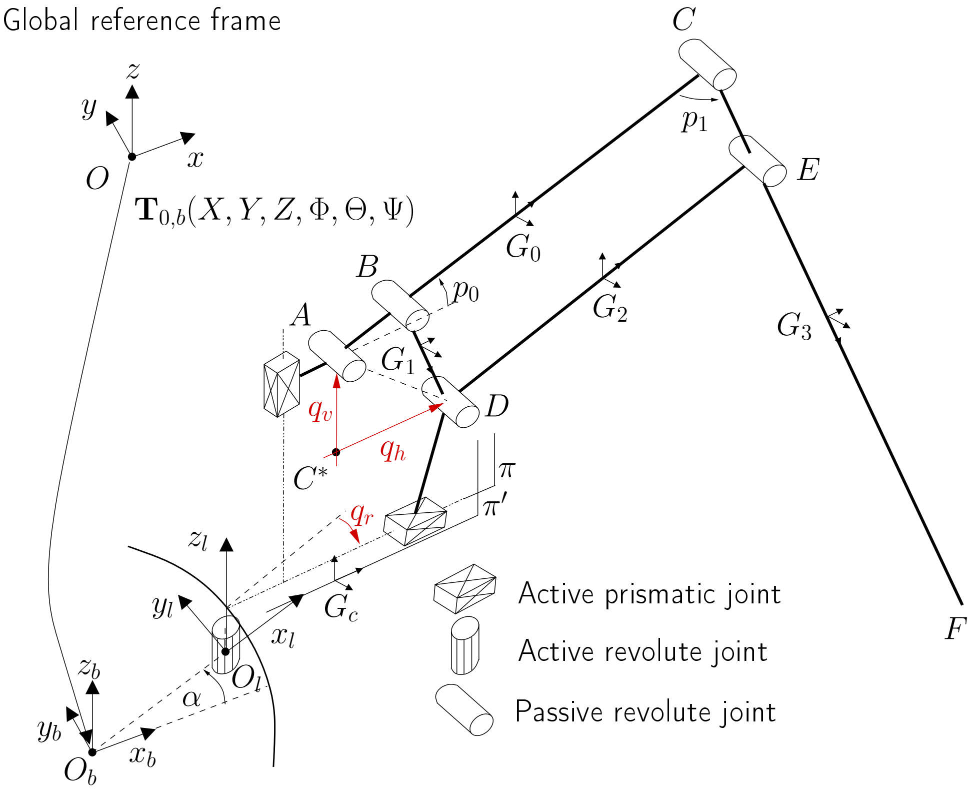


Figure 3.7: Kinematic description of a leg

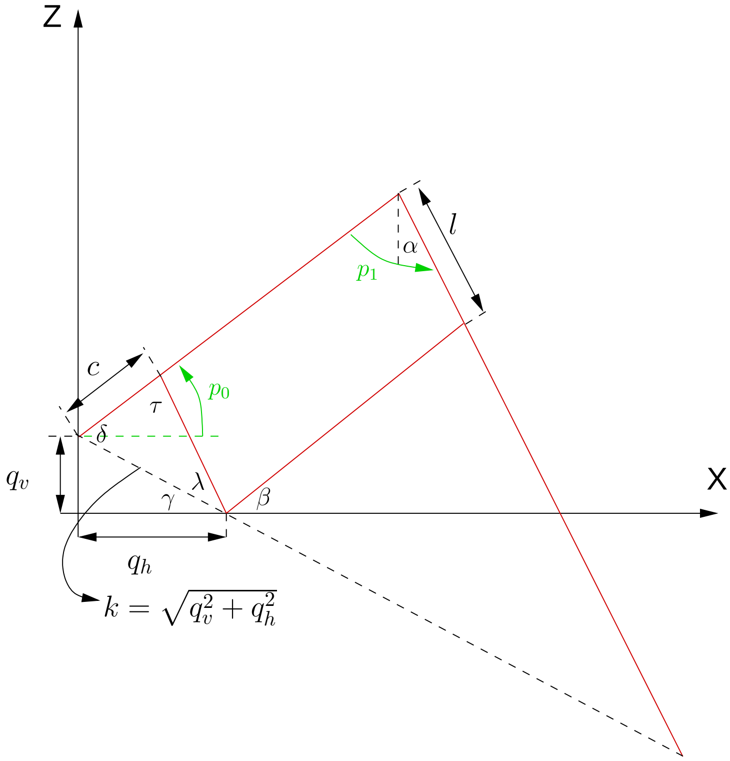


Figure 3.8: Kinematic study of the pantograph

easier, two dependent parameters p_0 and p_1 are expressed in terms of q_v and q_h. Consider the pantograph of Fig. 3.8: we have:

$$\begin{aligned} p_0 &= \delta - \gamma && (3.53)\\ p_1 &= \tau && (3.54)\\ & && (3.55) \end{aligned}$$

The angle γ is expressed by:

$$\gamma = \arcsin\left(\frac{q_v}{\sqrt{q_v^2 + q_h^2}}\right), \qquad (3.56)$$

For angles δ and τ, the Carnot's rule can be written in triangle klc:

$$\begin{aligned} l^2 &= k^2 + c^2 - 2kc\cos\delta && (3.57)\\ k^2 &= l^2 + c^2 - 2kc\cos\tau, && (3.58) \end{aligned}$$

from which δ and τ are expressed:

$$\begin{aligned} \delta &= \arccos\left(\frac{k^2 + c^2 - l^2}{2kc}\right) && (3.59)\\ \tau &= \arccos\left(\frac{c^2 + l^2 - k^2}{2lc}\right) && (3.60)\\ & && (3.61) \end{aligned}$$

Finally we come to:

$$\begin{aligned} p_0 &= \arccos\left(\frac{q_v^2 + q_h^2 + c^2 - l^2}{2c\sqrt{q_v^2 + q_h^2}}\right) - \arcsin\left(\frac{q_v}{\sqrt{q_v^2 + q_h^2}}\right) && (3.62)\\ p_1 &= \arccos\left(\frac{c^2 + l^2 - q_v^2 - q_h^2}{2lc}\right) && (3.63) \end{aligned}$$

The leg bodies are therefore characterized by the following HTM:

- leg chassis (the center of mass G_c belongs to π'):

$$\boldsymbol{T}_{0,c} = \boldsymbol{T}_{0,b} \cdot \boldsymbol{T}^{R_z}(\alpha) \cdot \boldsymbol{T}^d(O_bO_l) \cdot \boldsymbol{T}^{R_z}(-q_r) \cdot \boldsymbol{T}^d(O_lG_c); \qquad (3.64)$$

- the pantograph link 0 (center of mass G_0):

$$\boldsymbol{T}_{0,l0} = \boldsymbol{T}_{0,c} \cdot \boldsymbol{T}^d(G_cC^*) \cdot \boldsymbol{T}^d(0,0,q_v) \cdot \boldsymbol{T}^{R_y}(-p_0) \cdot \boldsymbol{T}^d(AG_0); \qquad (3.65)$$

- the pantograph link 1 (center of mass G_1):

$$\boldsymbol{T}_{0,l1} = \boldsymbol{T}_{0,l0} \cdot \boldsymbol{T}^d(G_0B) \cdot \boldsymbol{T}^{R_y}(\pi - p_1) \cdot \boldsymbol{T}^d(BG_1); \qquad (3.66)$$

- the pantograph link 2 (center of mass G_2):

$$\boldsymbol{T}_{0,l2} = \boldsymbol{T}_{0,c} \cdot \boldsymbol{T}^d(G_c C^*) \cdot \boldsymbol{T}^d(q_h, 0, 0) \cdot \cdot \boldsymbol{T}^{R_y}(-p_0) \cdot \boldsymbol{T}^d(DG_2) \qquad (3.67)$$

- the pantograph link 3 (center of mass G_3):

$$\boldsymbol{T}_{0,l3} = \boldsymbol{T}_{0,l0} \cdot \boldsymbol{T}^d(G_0 C) \cdot \boldsymbol{T}^{R_y}(\pi - p_1) \cdot \boldsymbol{T}^d(DG_3); \qquad (3.68)$$

- the rotors (subscript r) are related to the leg degrees of freedom q_r, q_v, q_h through the reduction ratio n_r, n_v, n_h:

$$\begin{aligned} \boldsymbol{T}_{r,r} &= \boldsymbol{T}_{0,c} \cdot \boldsymbol{T}^d(...) \cdot \boldsymbol{T}^{R_z}(n_r \cdot q_r) & (3.69) \\ \boldsymbol{T}_{r,v} &= \boldsymbol{T}_{0,c} \cdot \boldsymbol{T}^d(...) \cdot \boldsymbol{T}^{R_z}(n_v \cdot q_v) & (3.70) \\ \boldsymbol{T}_{r,h} &= \boldsymbol{T}_{0,c} \cdot \boldsymbol{T}^d(...) \cdot \boldsymbol{T}^{R_x}(n_h \cdot q_h). & (3.71) \end{aligned}$$

The rotors location has not been detailed explicitely for the sake of clarity of Fig. 3.7.

Contribution of kinematic improvements

Even if the robot model is not completely developed yet, it is interesting to attract the reader's attention to the kinematic computation improvements developed in Sect. 3.3.

Four simulations of a gait during 75 s have been performed on a 3GHz Dual Core PC (64 bits) with 2Go RAM :

- the first one does not use neither dependent parameters nor direct calculation of partial velocities;
- the second one uses the dependent parameters but still estimates the partial velocities with a numerical derivation;
- the third one uses direct computation of partial velocities but no dependent parameters;
- the last one enjoys dependent parameters and direct computation of partial velocities.

All the other modeled elements are identical: simulations are made for a tripod gait of the robot, with a leg cycle of 15 s, a main body forward velocity of 2 cm/s. It includes gravity, ground contact forces, actuators and controllers models, and LuGre friction in the joints.

Table 3.2 shows that both contributions clearly improve the simulation time, with a factor ≈ 2-3 for dependent parameters and ≈ 8-13 for direct computation of partial velocities.

Remark that the generation of the C++ file without the use of relative motion has not been successful with MuPAD, because expressions of the complete model are too long and require too much memory.

Computation of partial velocities	Dependent parameters	Simulation time
numerical derivation	no	93 min 53 s
numerical derivation	yes	32 min 59 s
direct method	no	7 min 11 s
direct method	yes	4 min 20 s

Table 3.2: Simulation times for different kinematic improvements

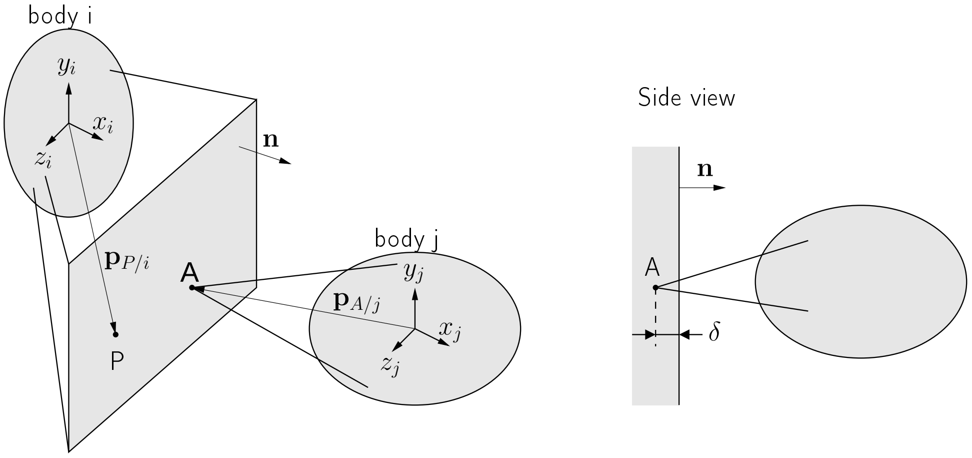


Figure 3.9: Penetration between two bodies [VERL07]

3.4.2 Ground contact forces

Modeling of ground contact forces has been widely discussed by the scientific community. In the case of walking robots, two approaches emerge from the literature: either contact with ground is modeled through constraint equations [BOWL06, KRAU97, PFEI00]; or by a force element (3-DOF spring-damper) inserted between the leg tip and the ground [SHIH87, FREE91, MANK92, SILV05, GONZ06, SHEN08]. With this method, the foot is assumed to not slip at ground interface. If sticking and slipping are considered, it is necessary to describe the friction at ground with a method similar to those of Ouezdou et al. [OUEZ98].

We assume the case of a contact between a point and a plane with some compliant behavior, which depends on the penetration δ represented in Fig. 3.9. The magnitude of the normal contact force (following the vector $\boldsymbol{n}$ direction) is

$$F_n = K_{gnd}\delta^{p_K} + D_{gnd}\delta^{p_D}\frac{d\delta}{dt}, \tag{3.72}$$

where K_{gnd} and D_{gnd} are the contact stiffness and damping respectively, and p_K, p_D fitting exponential coefficients. The tangential friction force is defined by a Coulomb friction model with a linear approximation at zero neighborhood:

$$\boldsymbol{F}_t \;=\; \begin{cases} -\mu \cdot F_n \cdot \frac{\boldsymbol{v}_g}{v_{glim}} & \text{if } ||\boldsymbol{v}_g|| < v_{glim}, \\ -\mu \cdot F_n \cdot \frac{\boldsymbol{v}_g}{||\boldsymbol{v}_g||} & \text{if } ||\boldsymbol{v}_g|| \geq v_{glim} \end{cases} \tag{3.73}$$

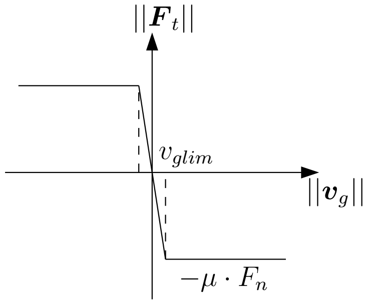


Figure 3.10: Friction model for the ground contact

Parameters	Symbol	Value
Penetration exponent	p_K	1.5
Penetration rate exponent	p_D	0.5
Soil stiffness	K_{gnd} (N/m^{p_K})	1e6
Soil damping	D_{gnd} (Ns/m$^{p_D+1}$)	5e4
Soil static friction coefficient	μ	0.3
Limit slip velocity	v_{glim} (m/s)	1e-4

Table 3.3: Parameters of ground contact interaction

where μ is the dynamic friction coefficient of the contact, $\boldsymbol{v}_g$ is the slip velocity of the contact point with respect to the ground and v_{lim} is the threshold preventing from discontinuity at zero velocity. A schematic representation of the model is given in Fig. 3.10.

The fact that there always exists a small slipping at leg-ground interface is not critical: by setting the threshold v_{lim} low enough, the slip velocity becomes negligible and the effects of friction can be caught.

AMRU5 is supposed to move on a rigid ground. The penetration exponent has been chosen to verify the Hertz's theory for a contact between a sphere and an elastic half plane:

$$F_{Hertz} = \frac{4}{3} E^* \sqrt{R} \cdot \delta^{3/2} \tag{3.74}$$

where F_{Hertz} is the contact force, R is the sphere radius, δ is the soil deflection, and E^* is an equivalent elastic modulus given by:

$$\frac{1}{E^*} = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2} \tag{3.75}$$

with $E1, E2$ the elastic moduli and ν_1, ν_2 the Poisson's ratios of the bodies in contact. In this theory, the soil stiffness parameters K_{gnd} is equivalent to $\frac{4}{3}E^*\sqrt{R}$. The soil stiffness has been fixed to 1e6 N/m^p_K such that the deflection is about 2.3 mm when the robot is standing on three legs. Parameters of Eq. (3.72) and (3.73) for feet/ground contact are gathered in Table 3.3. Soil damping is set to ensure good convergence of the numerical simulations.

3.4.3 Joint friction

Friction is a complex phenomenon which has a decisive influence on the actuation energy expenditure. Several models are briefly introduced in this section: the identification for AMRU5 is detailed in Chapter 5.

Friction at steady-state velocities is known for a while. Da Vinci already states that friction force is proportional to load, opposed to the direction of motion and independent of the contact area. Coulomb (1785), Morin (1833) and Reynolds (1866) followed him and contributed to many discovers. But friction is also a dynamic problem. Phenomena as stick-slip, hysteresis at low displacements and frictional memory [ARMS94b] confer copious challenges to engineers who desire to control servomechanisms with precision. All these mechanisms exist when surfaces are in contact, but they occur for different conditions (velocity, lubrication, load...) [RABI58]. An impressive state of the art about friction phenomena is given in [ARMS94].

Friction at steady-state velocities can be generally described by the *General Kinetic Friction* model (or *Stribeck* curve). For the case of a simple mass pulled by a force F_a (Fig. 3.11(a)), the friction force is expressed by the following equation:

$$F_f = \begin{cases} F_d + (F_s - F_d)\exp\left(-|v/v_{st}|^{\gamma}\right) + f_v v & \text{if } v \neq 0 \\ F_a & \text{if } v = 0 \text{ and } |F_a| < F_s \\ F_s sgn(F_a) & \text{if } |F_a| \geq F_s\text{, with } v = 0 \text{ but } \dot{v} \neq 0 \text{ (breakaway)} \end{cases} \tag{3.76}$$

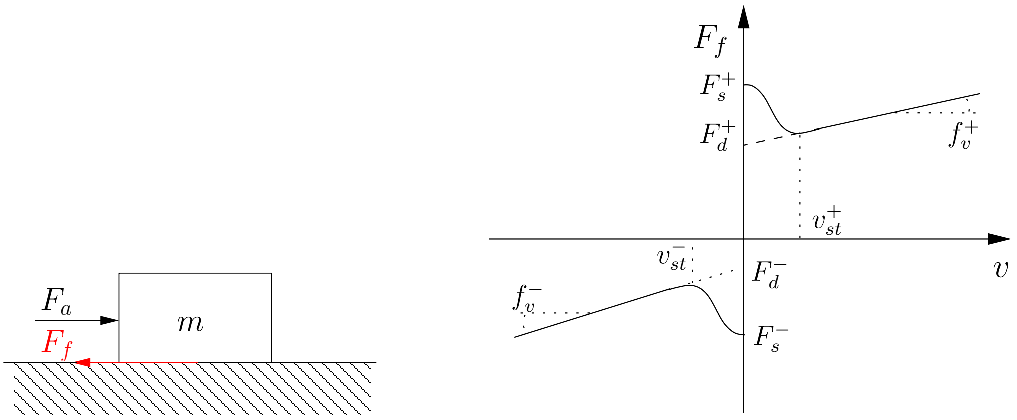
where :

- F_a, F_d and F_s are the force applied to the mass, the dry (or Coulomb) friction and the static friction (or *breakaway*) forces respectively;
- f_v is the viscous friction coefficient;
- v is the relative velocity between the sliding surfaces;
- v_{st} and γ are parameters used to fit the curve with experimental data; the Stribeck's velocity v_{st} corresponds to the transition zone between the partial and the full fluid lubrication [SPIN97].

The evolution of F_f with the velocity v can be asymmetric, as depicted in Fig. 3.11(b).

At high velocity, friction is easy to model, because the main effect is the lubricant viscosity. The Stribeck's effect happening at low velocity is unstable: when v increases from 0, friction force decreases, which rises the acceleration, so the velocity, of the mass. In [SPIN97], a detailed analysis of a sliding mass driven by a spring whose extremity is moved at constant velocity is performed. The spring physically represents the compliance of the transmission mechanism. Solving the equation of motion of this system leads to three conclusions, depending on the evolution of the friction at low velocity:

- constant friction force: a succession of stop and slide motions (called *stick-slip*) arises for a constant dynamic friction coefficient



(a) Sliding mass with friction

(b) General Kinetic Friction model

Figure 3.11: General Kinetic Friction model

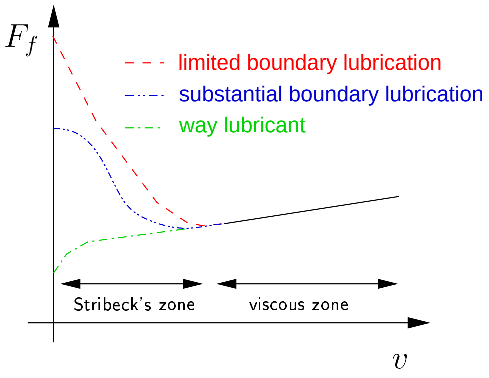


Figure 3.12: Steady-state characteristic of friction for different lubrications (adapted from [ARMS94])

- decreasing friction force with the velocity: the system is unstable as the system damping is negative (the system oscillates with an increasing amplitude);
- increasing friction force with the velocity: the system is stable and without oscillations.

Figure 3.12 emphasizes the importance of the lubrication in the system: for way lubricant[3], static friction is below the Coulomb's friction, consequently the stick-slip should not occur. But for limited or substantial boundary lubrication, commanding servomechanisms with precision at low velocity is quite difficult (see for example the identification and control made in [PAPA04]).

In addition to the stick-slip, other dynamic behaviors have also been denoted for very low velocity (i.e. below the Stribeck velocity):

- frictional memory [OLSS98, ARMS94] : phase lag between the change in velocity and the change in friction force in the Stribeck's zone;

[3] Way lubricant is a fluid designed for lubrication of slideways of metalworking machine tools. Most of way lubricants are mineral oil based, however synthetic oil based way lubricants are also used in some applications.

- varying break-away force [RABI58, CANU95] : break-away force F_s decreases as force rate increases;

- presliding displacement : sliding is a direct effect of the asperities break. Before "cracking", it is common to model asperities as a whole of springs. So presliding phase concerns the force-displacement relationship, and finds its application in high-precision servomechanisms. A typical plot is given in Fig. 3.13. The curve describes an hysteresis, thus a loss of energy;

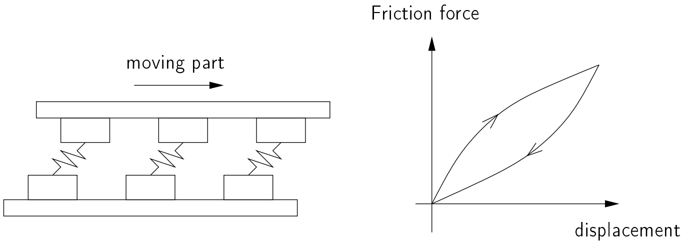


Figure 3.13: Presliding regime

- hysteresis with nonlocal memory : precise experiments in machine tools [ALTP99, SWEV00] have emphasized that hysteresis had a nonlocal memory. A hysteresis with nonlocal memory is defined as *an input-output relationship for which the output at any instant not only depends on the output at some time instant in the past and the input since then, but also on past extremum values of the input or output as well* [SWEV00];

- transition curves in presliding regime [ALTP99, SWEV00] : stiffness at contact surfaces is playing an important role in presliding regime, and is highly non-linear.

Many different dynamic models exist to include these phenomena. They involve internal state variables to model the presliding displacement. They are:

- the *Dahl's* model, developed by Dahl in the sixties [ARMS94];

- the *LuGre* model, developed by Canudas de Wit et al. [CANU95];

- the *Single State Elastoplastic* model, of Dupont et al. [DUPO02];

- the *Generalized Maxwell-Slip* model from KULeuven in [LAMP02, LAMP03, LAMP04, AL-BE04, AL-BE05].

Another method to include friction in MBS is by the means of *nonsmooth mechanics*. The approach is not treated in this work: details can be found in [PFEI04, ACAR08].

Choice of the LuGre Model for Hexapod Modeling Most of the authors working on legged robots, or in robotics in general, model the friction as a simple dry/viscous velocity dependence [BRUN96, PFEI00, GROT04], or with a mathematic formulation which best fits their measurements [GROT02, BONA06]. Problem of the latter formulation is that discontinuity at zero velocity generates numerical instabilities. Two solutions commonly used are the dead-zone (Karnopp, 1985, described in [ARMS94]) or the linear approximation around zero velocity (Eq. (3.73)).

Nevertheless, these assumptions are not adapted to model friction in the case of a constant position (null velocity), which is the case for walking machines. For example, an AMRU5 leg keeps its vertical position constant throughout its support phase. The related actuator should then counteract the ground reaction force, while in reality, friction "helps" in some way to keep this setpoint without power consumption due to the transmission irreversibility.

Thus, behavior at zero velocity is necessary. Given the fact that a walking machine has several actuators and that a too simple friction model would give bad results in terms of energy expenditure, the choice of the LuGre [CANU95] model seems to be a good tradeoff between simplicity and realism.

The physical interpretation of the LuGre model is that sliding surfaces have *bristles* undergoing a visco-elastic deformation as shown in Fig. 3.14.

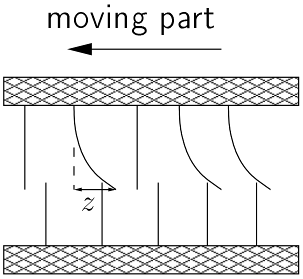


Figure 3.14: Bristles deflection

The mean bristles deflection is represented by the internal state variable z, whose dynamic behavior corresponds to

$$\dot{z} = v - \frac{|v|}{g(v)} z, \tag{3.77}$$

where v is the relative velocity between surfaces, and $g(v)$ the steady-state characteristic curve. The function $g(v)$ is commonly chosen to model a steady-state characteristic of friction including the Stribeck's effect:

$$g(v) = \frac{F_d + (F_s - F_d) \exp -(v/v_{st})^2}{\sigma_0} \tag{3.78}$$

where σ_0 is the stiffness of the bristles. This function must be positive [OLSS96].

The friction force is computed from the bristles stiffness and viscous contributions:

$$F_f = \sigma_0 z + \sigma_1 \dot{z} + f_v v \tag{3.79}$$

with σ_1 a damping coefficient set to obtain an adequate behavior of the simulation (see dissipativity issue below). The nice property of the model is that, for steady-state velocity, the dynamic behavior of the bristles (Eq.3.77) and the friction force (Eq. (3.79)) comes down to

$$\begin{aligned} z_{ss} &= \frac{v}{|v|}g(v) = g(v)sgn(v) && (3.80)\\ F_{f,ss} &= \sigma_0 z_{ss} + f_s v && (3.81)\\ &= F_d + (F_s - F_d)\exp -(v/v_{st})^2 + f_v v && (3.82) \end{aligned}$$

where Eq.(3.82) is equivalent to the general kinetic friction model (Eq.3.76) with $\gamma = 2$. Advantages of the LuGre model are:

- previous use in robotics [BONA05];
- well conditioned for numerical integration [OLSS96];
- deals with asymmetric steady-state characteristics [OLSS96];
- few parameters to identify.

However, it presents three limitations :

1. Dissipativity issues have been raised because of the risk of energy storage in the model. It has been shown [OLSS96] that the damping factor σ_1 should be velocity-dependent and must satisfy the following condition:

   $$0 \leq \sigma_1(v) \leq 4\frac{\sigma_0 g(v)}{|v|}. \qquad (3.83)$$

   Equation (3.83) is easily verified when $|v|$ decreases. For higher velocities, Olsson suggests to use the following form to have a vanishing damping coefficient for increasing velocities:

   $$\sigma_1(v) = \sigma_1 \exp\left(-v/v_d\right)^{\delta_d} \qquad (3.84)$$

   where v_d and δ_d are fitting parameters. He suggests to choose $\delta_d = 2$ and v_d in relation to the Stribeck's velocity [OLSS96].

2. Hysteresis with nonlocal memory can not be simulated confidently. Several authors [OLSS98, ALTP99, LAMP04] show both by experiments and simulations this limitation. Dupont in [DUPO02] speaks about *drift*, and explains that when a load with friction similar to the case of Fig. 3.11(a) is submitted to a force $F_a = b + asin(\omega t)$, with $b > a$, simulation with LuGre model gives a global displacement of the mass, while experimentally the mass oscillates around a constant equilibrium position. This phenomenon is again emphasized in [ALTP99], where Altpeter shows the impossibility for the LuGre model to reach the exact velocity reversal point many times (point at which direction of motion changes for back and forth motion), while experimentally this point is always reached, no matter how many reversals happened previously.

3. Variations of stiffness in the presliding regime can not be modeled, because of the limited number of parameters (see [OLSS98]).

Both last limitations concern very accurate servomechanisms (μm or nm precision), and do not imply walking robots. The first one has to be carefully handled to avoid non physical results.

In the AMRU5 modeling, the joint friction model links the leg joint velocity to a friction torque acting on the corresponding motor shaft. The state equation of the bristle (3.77) is put under the second order residual form for consistent integration:

$$\ddot{q}_z - \dot{q} + \frac{|\dot{q}|}{g(\dot{q})} \cdot \dot{q}_z = 0, \tag{3.85}$$

where $\ddot{q}_z = \dot{z}$ and $\dot{q}_z = z$, and where the steady-state characteristic $g(\dot{q})$ is rewritten in terms of friction torque at joint [4]:

$$g(\dot{q}) = \frac{\tau_{f,d} + (\tau_{f,s} - \tau_{f,d}) \exp\left(-(\dot{q}/\dot{q}_{st})^2\right)}{\sigma_0}, \tag{3.86}$$

where $\dot{q}_{st}$ is the Stribeck's velocity of the joint, and $\tau_{f,d}$, $\tau_{f,s}$ the dry friction and the breakaway torques respectively.

Therefore, the friction torque τ_f acting on the considered joint is

$$\tau_f = \sigma_0 \dot{q}_z + \sigma_1 \ddot{q}_z + f_v \dot{q}, \tag{3.87}$$

Further explanations about the identification of friction are given in Chapter 5.

[4] In the following, the *generalized forces* applied to the joints are called *joint torques* as it is commonly the case in robotics [SICI09]. For a rotational joint, the torque is effectively in Nm; however, for a translational joint, the related joint torque is in N.

3.4.4 Actuation

In legged machines, a classical actuation system is composed of a DC motor with gearing (planetary gear, ball screw, ...) and its controller.

DC motor The behavior of a DC motor with permanent magnet is described by the following set of equations [SICI09]

$$
\begin{aligned}
u_a(t) &= Ri(t) + L\frac{di(t)}{dt} + u_e(t) && (3.88)\\
u_e(t) &= k_v\dot{\theta}_m(t) && (3.89)\\
c_m(t) &= k_t i(t) && (3.90)
\end{aligned}
$$

with

- u_a, u_e and i the armature voltage, the back electromotive force, and the current;
- k_v and k_t the voltage and the torque contants (they are identical [MAYE00]);
- $\dot{\theta}_m$ the velocity at the rotor shaft;
- c_m the motor torque.

The mechanical torque c_m is directly proportional to the current flowing through the coil armature of the motor. At this stage, it is convenient to differentiate the motor torques $\boldsymbol{c}_m$ from the joint torques $\boldsymbol{\tau}_m = (\tau_r, \tau_v, \tau_h)$. In the model, the motor torques are applied to bodies defining the rotor shafts. The joint torques are related to the joint space defined by (q_r, q_v, q_h).

Practically, the electrical equation (3.89) of a motor is put under its residual form in `EasyDyn`:

$$f = R\dot{q}_i + L\ddot{q}_i + k_v\dot{\theta}_m - u_a \qquad (3.91)$$

where u_a has been replaced by u, di/dt by $\ddot{q}_i$ and i by $\dot{q}_i$. The velocity of the rotor is expressed as a function of the joint velocity and the reduction ratio:

$$\dot{\theta}_m = n\dot{q}. \qquad (3.92)$$

The coupling between the motor and the multibody model is depicted in Fig. 3.15. The input of the actuator is the voltage u_a. The electrical model computes the resulting current with residual equation (3.91), and then a torque is applied to the rotor shaft following Eq. (3.90). The multibody model reacts consequently, and provides the velocity of the joint, transformed in rotor velocity by Eq. (3.92).

The eighteen motors are identical. Their characteristics are given in Table 3.4.

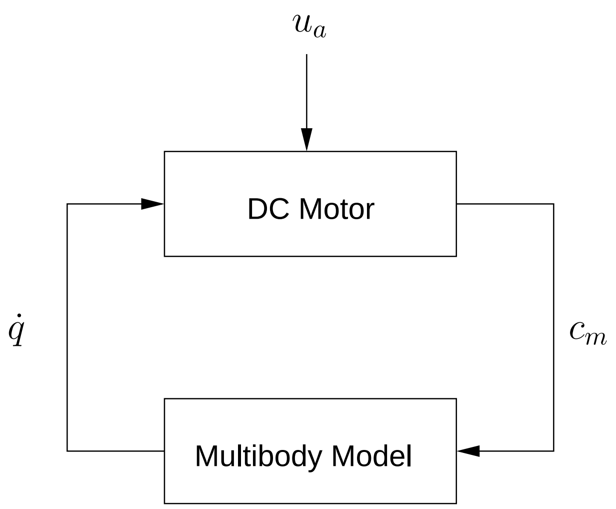


Figure 3.15: Coupling between electrical part of the motor and multibody model

Parameters	Symbol	Value
Nominal voltage (Volt)	u_{sat}	15
Coil resistance (Ω)	R	0.334
Coil inductance (mH)	L	0.09
Torque constant (Nm/A)	k_t	19.4e-3
Rotor inertia (gcm^2)	J_r	65.5
Weight (kg)	m_{motor}	0.34
Speed without load (tr/min)	(for information)	7070

Table 3.4: Motors parameters

Controllers The role of the controller is to determine the voltage command to apply to the motor. Most of the controllers encountered in walking machines are position PI(D) or velocity P(D) controllers. The general Laplace form of a PID position controller with filtered derivative is considered [ASTR06]:

$$C(s) = \frac{U(s)}{E(s)} = K\left(1 + \frac{1}{sT_I} + \frac{sT_D}{1 + sT_D/N}\right) \tag{3.93}$$

where K, T_I and T_D are the gain controller, the integral and the derivative time constants respectively, $U(s)$ is the Laplace transform of the controller output signal and $E(s)$ the one of the position error signal $\epsilon(t) = q_d(t) - q(t)$, with q_d the desired joint position. Coefficient N limits to N/T_d the high frequencies gain caused by the derivative action (high frequencies *roll-off*).

Implementation of such a system must appear in a residual form (3.1) for `EasyDyn`: consider first the derivative form of Eq. (3.93):

$$u + \frac{T_D}{N}\dot{u} = K(1 + \frac{T_D}{NT_I})\epsilon + KT_D(1 + \frac{1}{N})\dot{\epsilon} + \frac{K}{T_I}\int_0^t \epsilon(\tau)d\tau. \tag{3.94}$$

Transposition of such an equation is not straightforward in `EasyDyn`: for each controlled joint, two extra configuration parameters are required:

- The first one is due to the voltage derivative $\dot{u}$: for consistent integration, we define $\ddot{q}_u = \dot{u}$ and $\dot{q}_u = u$, q_u being meaningless.
- The second one is used for the computation of the integral of the error term, which is not directly available. Trick consists in adding an extra differential equation of the form:

  $$\ddot{q}_I + \omega_0(\dot{q}_I - \epsilon) = 0 \tag{3.95}$$

  in which ω_0 is quite superior to the bandwidth of the closed-loop system. Consequently, $\ddot{q}_I \simeq \dot{\epsilon}$, $\dot{q}_I \simeq \epsilon$ and $q_I \simeq \int_0^t \epsilon(\tau)d\tau$.

From this, the differential equations implemented in `EasyDyn` for one PID controller are:

$$\begin{aligned} 0 &= \dot{q}_u + \frac{T_D}{N}\ddot{q}_u - K(1 + \frac{T_D}{NT_I})\dot{q}_I - KT_D(1 + \frac{1}{N})\ddot{q}_I - \frac{K}{T_I}q_I && (3.96)\\ 0 &= \ddot{q}_I + \omega_0(\dot{q}_I - \epsilon) && (3.97) \end{aligned}$$

For a digital implementation, the additional differential equations are not required, because the derivative and the integral error terms of Eq. (3.94) are estimated by the following algebraic equations:

$$\begin{aligned} \dot{u}(t) &\simeq \frac{u(kT_s) - u((k-1)T_s)}{T_s} = \frac{u(t) - u(t - T_s)}{T_s} && (3.98)\\ \dot{\epsilon}(t) &\simeq \frac{\epsilon(kT_s) - \epsilon((k-1)T_s)}{T_s} = \frac{\epsilon(t) - \epsilon(t - T_s)}{T_s} && (3.99)\\ \int_0^t \epsilon(\tau)d\tau &\simeq \sum_{j=0}^{k} \epsilon(jT_s) \cdot T_s && (3.100) \end{aligned}$$

with T_s the control sampling time and k the present time index. Note that the evolution of the trajectory and these digital terms are updated every nth call of the `SaveData()` procedure of `EasyDyn` ($n \geq 1$).

Parameters of the controllers are found in Chapter 4, where their design is performed.

	MBS	LuGre friction	DC motors	PID controllers	Total
Number of bodies	49	-	-	-	49
Number of DOFs	24	18	18	36	96

Table 3.5: Summary of the modeling in terms of bodies and degrees of freedom

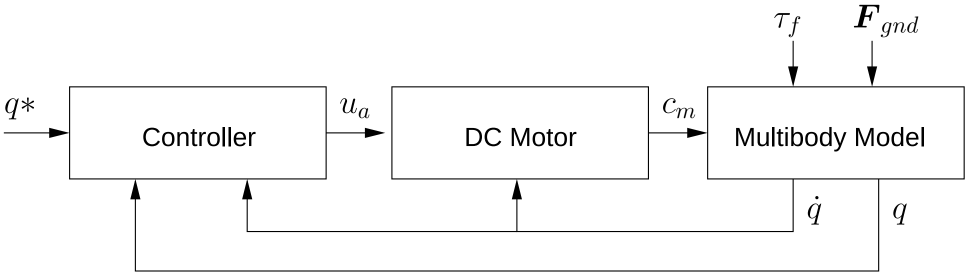


Figure 3.16: Global overview of the modeling for one DOF

3.5 Summary of the modeling

A multiphysic model of the AMRU5 robot has been presented in this section. It is based on a multibody model developed with the minimal coordinates approach, whose steps are:

1. choose a set of independent configuration parameters $\boldsymbol{q}$;
2. describe the kinematics of each body in terms of the chosen configuration parameters;
3. describe the applied forces acting on each body;
4. build the equations of motion;
5. integrate numerically the equations of *motion* + the *electrical* and *friction* equations to obtain the time history of $\boldsymbol{q}, \dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$.

The EasyDyn C++ library coupled with the script CAGeM running on the MuPAD computer algebra system helps us in the description of the mechanical system. Management of inertia forces and gravity is automatic when building the equations of motion.

On top of this, several multiphysic elements are included in the model:

- the ground contact forces;
- the friction in the joints;
- the actuation by DC motors with their controllers.

Table 3.5 summarizes the modeling in terms of bodies and DOFs, while Fig. 3.16 shows the coupling interaction between the different elements of the model.

CHAPTER 4

Control design

It is interesting to focus on a single leg model to design the controllers. In this chapter we justify the choice of the controllers developed on AMRU5. Section 4.1 presents the multibody model of one leg, without ground contact and friction forces. Simplification of the actuator model is described in Section 4.2. The coupling interactions between joints are considered in Section 4.3. The controller synthesis is made in Section 4.4. The main conclusion of this chapter is that each of the DOF of the leg can be seen as a Single Input Single Output system, controlled with a simple PI position loop. Section 4.5 ends the chapter and details the simplification of the whole robot dynamic model.

4.1 Leg model

The complete leg+motors model (dimension 6×6) can be put under the form:

$$\begin{pmatrix} \boldsymbol{M}(\boldsymbol{q}) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{L} \end{pmatrix} \cdot \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \dot{\boldsymbol{i}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) & -\boldsymbol{K} \\ \boldsymbol{K} & \boldsymbol{R} \end{pmatrix} \cdot \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{i} \end{pmatrix} + \begin{pmatrix} \boldsymbol{g}(\boldsymbol{q}) \\ \boldsymbol{0} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{pmatrix} \cdot \boldsymbol{u} \quad (4.1)$$

where

- $\boldsymbol{L}$ and $\boldsymbol{R}$ are diagonal 3×3 matrices with inductance and resistance of the motors respectively (see Table 3.4);
- $\boldsymbol{K}$ is a diagonal 3×3 matrix, whose element K_{ii} is the product of the motor torque constant k_t and the reduction ratio n_i of the corresponding joint;
- $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ a tangent damping matrix[1]; the Coriolis and centrifugal terms of equation i are rigorously expressed as:

$$\sum_k \sum_l C_{ikl}\dot{q}_k\dot{q}_l; \quad (4.2)$$

[1] It is not usual to describe this matrix by $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ because it could let think that terms are different from zero with null velocities, while $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is null if $\dot{\boldsymbol{q}} = 0$

- $\boldsymbol{g}(\boldsymbol{q})$ is the 3×1 contributions of the gravity forces;
- $\boldsymbol{u}$ is a 3×1 array with motors voltages.

Note that the motor torques applied to the mechanical system appear in the terms $-\boldsymbol{Ki}$ in the upper part of the system (4.1), and that current is computed through the dynamics of the electrical circuit.

4.2 Inductance simplification

First observation is that the inductance of the motors is very low and can be neglected: indeed, the electrical time constant is given by

$$\tau_e = \frac{L}{R} = \frac{0.09e-3[\text{H}]}{0.334[\Omega]} = 0.27\,\text{ms} \tag{4.3}$$

while the mechanical time constant is about 6 ms (see Section 4.4). Therefore, dynamics of the electrical part is much faster than the mechanical one: consequently, at mechanical time scale, the degrees of freedom regarding the current can be forgotten. In other words, Eq. (3.89) simply becomes:

$$u = Ri + k_t n\dot{q} \tag{4.4}$$

The torque applied to the joints τ is proportional to the motor torque, and subsequently to the current:

$$\begin{aligned} \tau &= n \cdot c_m && (4.5)\\ \tau &= n \cdot k_t \cdot i && (4.6) \end{aligned}$$

Substituting Eq. (4.4) in (4.6) leads to:

$$\tau = n \cdot k_t \cdot \left(\frac{u - nk_t\dot{q}}{R}\right). \tag{4.7}$$

These joint torques can be inserted in the mechanical model of the leg (dimension (3×3)), such as:

$$\boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \cdot \dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{4.8}$$

By substituting Eq. (4.7) in system (4.8) we finally come to the following 3×3 system:

$$\boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + (\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{C}_{emf}) \cdot \dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{F}_{elec} \cdot \boldsymbol{u} \tag{4.9}$$

where $\boldsymbol{C}_{emf}$ contains the extra damping from back electromotive force

$$\boldsymbol{C}_{emf} = \begin{pmatrix} (n_r k_t)^2/R & 0 & 0 \\ 0 & (n_v k_t)^2/R & 0 \\ 0 & 0 & (n_h k_t)^2/R \end{pmatrix}, \tag{4.10}$$

and $\boldsymbol{F}_{elec}$ the influence matrix

$$\boldsymbol{F}_{elec} = \begin{pmatrix} n_r k_t/R & 0 & 0 \\ 0 & n_v k_t/R & 0 \\ 0 & 0 & n_h k_t/R \end{pmatrix}. \tag{4.11}$$

4.3 Coupling interactions

Up to now, the system has 3 inputs and 3 outputs, but the coupling between joints has not been investigated yet. Even if the pantograph mechanism indicates that coupling should be small, it is nevertheless interesting to pursue this study. Mass, tangent damping and stiffness, and input contribution matrices have been computed with the `EasyDyn` input-output linearization procedure, for 24 configurations of the pantograph mechanism:

- q_r has been set to 0;
- q_v varies from 0.04 to 0.09 by step of 0.01 m;
- q_h varies from 0.02 to 0.05 by step of 0.01 m.

4.3.1 Mass matrix

We begin the investigation by examining the linearized mass matrix:

$$\boldsymbol{M}|_0 = \begin{pmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{pmatrix} \tag{4.12}$$

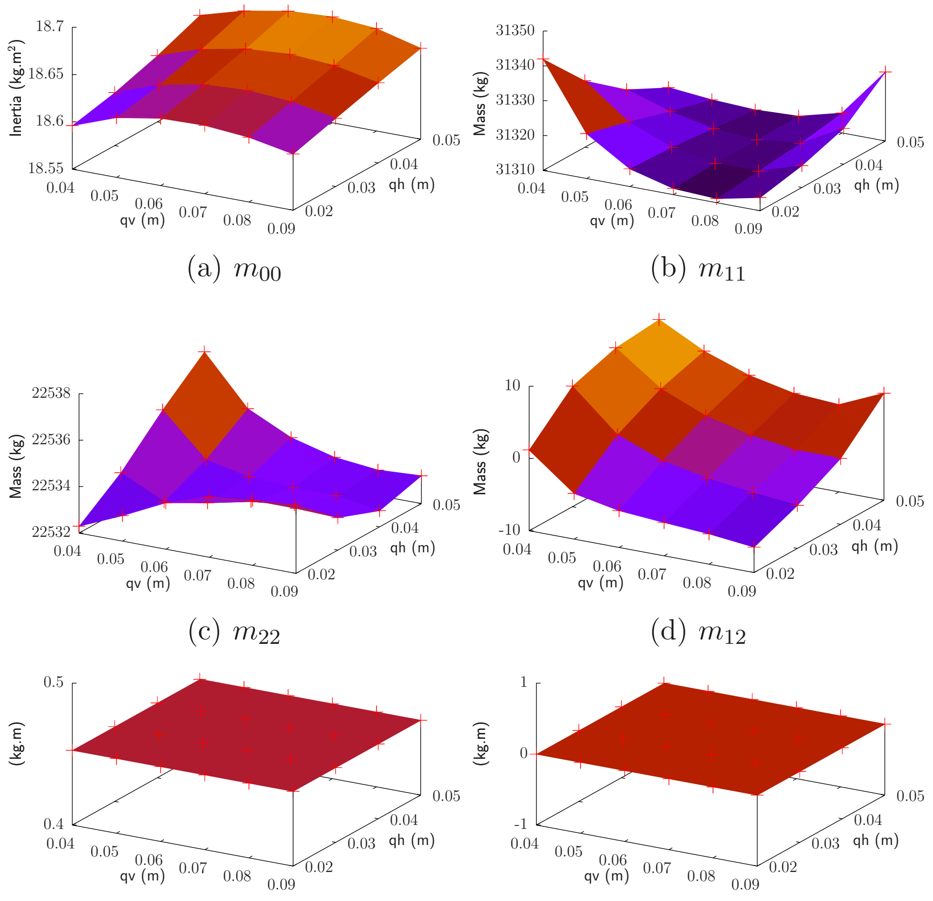
Figure 4.1 gives an overview of the mass matrix elements relative to one leg. Diagonal term m_{00} relative to the rotation of the leg increases with q_h, but in a very limited way so that this term is considered as constant for the controller design. The terms m_{11} and m_{22} (relative to q_v and q_h respectively) are also quasi-constant, whatever the pantograph configuration. Reason is the high gearing of the motors which tend to linearize the dynamics. The coupling terms are very low with respect to diagonal ones. As an example m_{12} is represented in Fig. 4.1(d): it can be neglected in the following. Terms m_{20} is null and m_{10} worth less than 3% of m_{00}.

4.3.2 Damping matrix

Tangent damping matrix has been computed with maximum velocity of the three joints, so as to maximize the matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. Diagonal contributions of $\boldsymbol{C}_{emf}$ are (in N.s/m):

$$\boldsymbol{C}_{emf} = \begin{pmatrix} 3388 & 0 & 0 \\ 0 & 5.72e6 & 0 \\ 0 & 0 & 4.12e6 \end{pmatrix}, \tag{4.13}$$

The diagonal terms of the complete damping matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{C}_{emf}$ are given in Fig. 4.2. As expected, the main source of damping comes from the back electromotive force: values are very close to those of $\boldsymbol{C}_{emf}$ alone (variation due to centrifugal force $\leq 0.01\%$). Other terms are close to zero: the maximum value, encountered by c_{12}, is about 40 N.s/m.

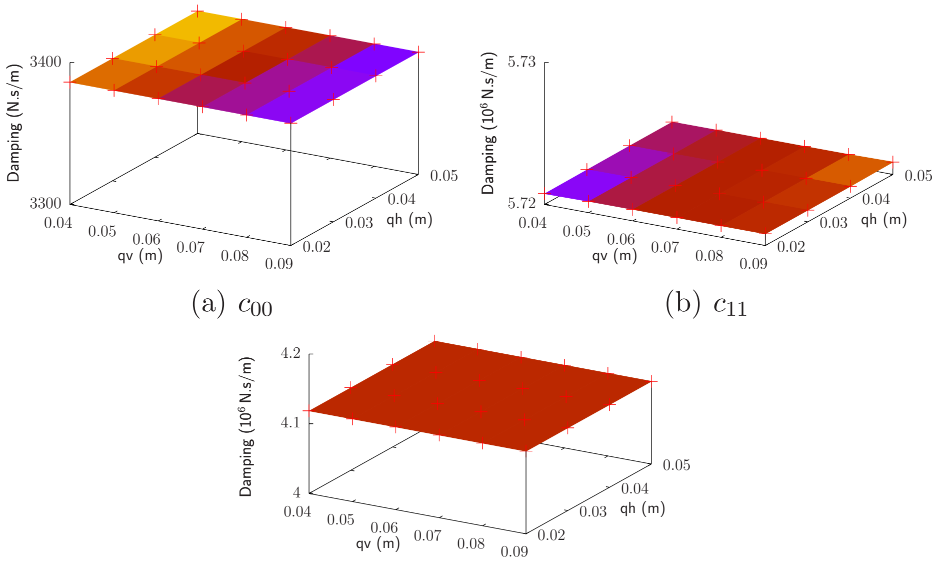


(a) m_{00} (b) m_{11}

(c) m_{22} (d) m_{12}

(e) m_{10} (f) m_{20}

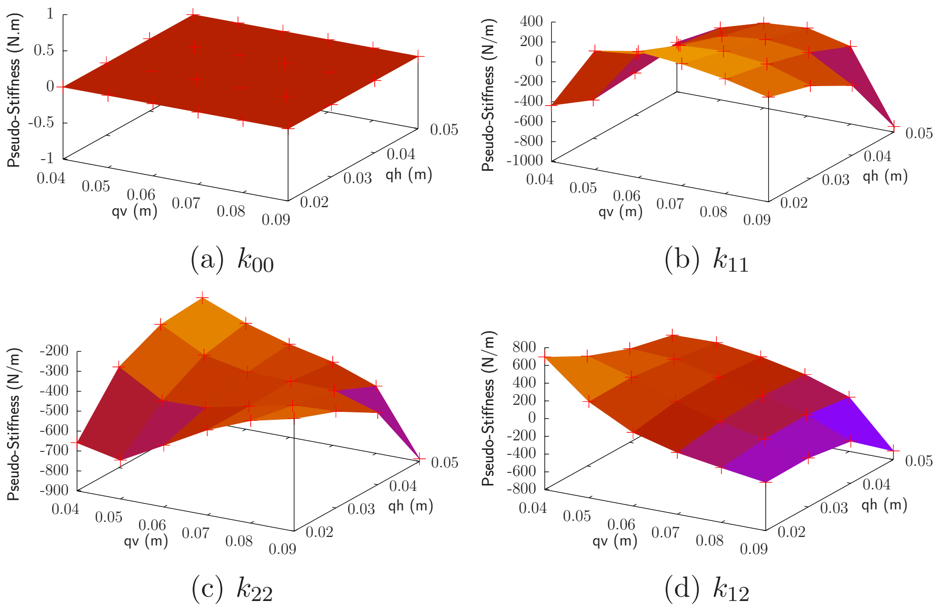
Figure 4.1: Mass matrix contributions with respect to q_v and q_h



(a) c_{00} (b) c_{11}

(c) c_{22}

Figure 4.2: Damping matrix contributions with respect to q_v and q_h



(a) k_{00}

(b) k_{11}

(c) k_{22}

(d) k_{12}

Figure 4.3: Tangent stiffness matrix contributions with respect to q_v and q_h

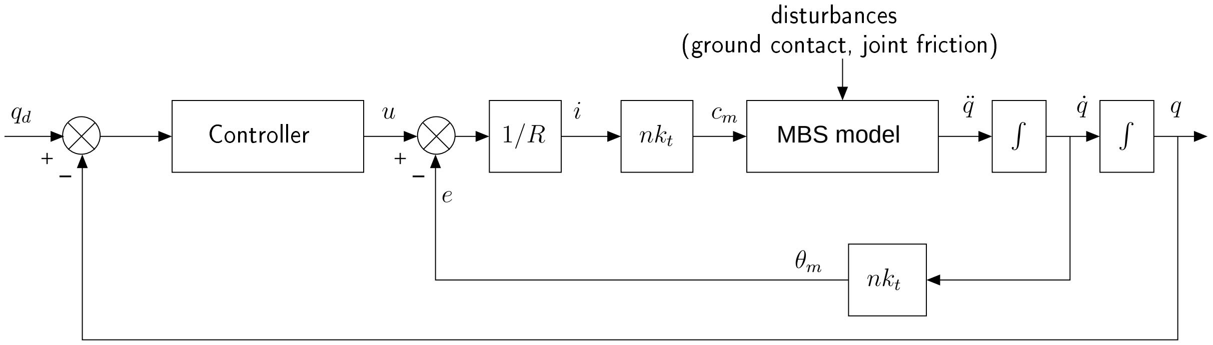


Figure 4.4: Position control loop for one joint

4.3.3 Stiffness matrix

Figure 4.3 shows tangent stiffness matrix when gravity effects are linearized. All the terms relative to the rotation, i.e. k_{00}, k_{01}, k_{02} and their symmetric are null, because gravity does not influence the leg rotation. Variations of k_{11}, k_{22} and k_{12} are quite important. Also, the coupling terms k_{12} is not negligible with respect to k_{11} and k_{22}. However, these terms are omitted in the controller design, because gravity is a disturbance from the control point of view, which will be corrected by the integral term of the controller (see Section 4.4).

4.3.4 Decoupling in 3 SISO loops

Previous section has demonstrated the low coupling between each joint and the linearity of the dynamics. Consequently, each motor will be controlled as a SISO system with a position controller, according to Fig. 4.4. Only position is used in this particular case, because motors are equipped with incremental encoders which directly give position of the rotors. Note that the voltage supply is limited to 15 V; therefore, a saturation appears in the loop scheme. This saturation effect is discussed in Section 4.4.

4.4 Controller synthesis

4.4.1 Open-loop transfer functions

The open-loop transfer function between the motor voltage $U(s)$ and the joint velocity $V(s)$ of Fig. 4.4 is written:

$$\frac{V(s)}{U(s)} = \frac{\frac{nk_t}{R} \cdot \frac{1}{Js}}{1 + \frac{(nk_t)^2}{R} \cdot \frac{1}{Js}} = \frac{1/(nk_t)}{1 + s \cdot \frac{RJ}{(nk_t)^2}} = \frac{K}{1 + s\tau_m} \tag{4.14}$$

where J is given by the diagonal element of the linearized mass matrix (for a neutral position of the leg $q_r = 0$, $q_v = 0.06$ and $q_h = 0.04$), τ_m is the mechanical time constant and K the system gain. From this, we can deduce the resulting open-loop transfer function between joint positions (q_r, q_v, q_h) and motor commands (u_r, u_v, u_h) for the three joints:

$$\frac{Q_r(s)}{U_r(s)} = \frac{1}{s} \cdot \frac{0.031}{1 + 0.0059s} \tag{4.15}$$

$$\frac{Q_v(s)}{U_v(s)} = \frac{1}{s} \cdot \frac{0.00075}{1 + 0.0058s} \tag{4.16}$$

$$\frac{Q_h(s)}{U_h(s)} = \frac{1}{s} \cdot \frac{0.00088}{1 + 0.0058s} \tag{4.17}$$

It is interesting to note that the mechanical time constants of the three systems are close to the 6 ms mentioned in the vendor specification: in fact, it is slightly inferior because we do not assume any viscous friction in the motor, while the vendor does. The second thing is that inertia of pantograph elements does not increase τ_m with respect to specifications, because of the high gearing of the motors.

4.4.2 The choice of PI controllers

The choice of PI controllers has been made at the beginning of this thesis: in earlier development, the controllers were implemented directly on the microcontrollers embedded on the robot, but time for the computation of the derivative term was prohibitive. Therefore we simplify the PID in PI controller. Then the controllers moved on the master PC, which would be able to easily compute the PID. But initial design showed good behavior so we kept the PI controllers. Finally one further argument is that the derivative action theoretically increases the performances of the system, but makes it less robust, for example with respect to a change in friction coefficient, which has been the case in this work when ball-screw gearings of horizontal joints have been replaced by friction bearings (see Section 5).

4.4.3 Saturation as bandwidth limitation

Saturation is an important limitation in our system; in fact it is the most important. This non-linear element acts like a variable gain which depends on the amplitude of the

Joint	K (V/rad or V/m)	T_i (s)	Bandwidth (Hz)	Phase margin (°)	Overshoot (%)
q_r	483	0.5	2.94	77.3	9.5
q_v	19863	0.5	2.92	77.4	9.6
q_h	16854	0.5	2.91	77.4	9.6

Table 4.1: Controller gains and closed-loop performances

controller output, and is independent of the frequency [MULL09]. Therefore, it is useless to design a controller to obtain a too high bandwidth in closed-loop: it would risk to saturate the actuator. Several PI controllers have been designed and tested by simulation in closed loop, with a position ramp reference, corresponding to the maximum reachable velocity of each joint (based on the vendor specification – see speed without load in Table 3.4). In fact, this maximum velocity has been multiplied by a factor 0.8, because the supply voltage on board is not 15 V but slightly superior to 12 V, due to the serial resistance of the cable linking the robot to the power supply.

4.4.4 Controller gains

Simulations with Simulink reveal that a closed-loop bandwidth of approximately 3 Hz keeps the controller output below the saturation level for the reference position ramp. We have paid attention to have a sufficiently fast tracking: after 0.5 s, the joint position error is about 0.1 mm.

Table 4.1 gives the controller parameters with respect to joint type, and their closed-loop performances (bandwidth, phase margin and overshoot).

4.4.5 Digital implementation and antiwindup

Because control of the robot is digital, one must be careful to have a closed-loop bandwidth below the sampling time of the system. A rule of thumbs in digital control is that a sampling frequency 30 times higher than the closed-loop bandwidth allows to assimilate the system as a continuous system: this is the case for AMRU5 because sampling is at 100 Hz.

Digital implementation of these controllers is a particular case of Eq.(3.93). For a sampling time Δt, the output k of the controller is written:

$$u_k = K\epsilon_k + I_{k-1} + \frac{K}{T_i}\epsilon_k \Delta t \tag{4.18}$$

$$u_k = P_k + I_k \tag{4.19}$$

where I_k is the integral and P_k the proportional term respectively. The integral term at time k is the sum of I_{k-1} and $\frac{K}{T_i}\epsilon_k\Delta t$.

However, saturation remains possible. In this case, it is recommended to implement an anti-windup scheme to prevent an overflow of the integral term. Practically, if saturation is reached, the integral term I_k is recomputed from the difference between the saturation and the proportional term P_k:

$$I_k = u_k - P_k, \tag{4.20}$$

ant then it is saved and used as I_{k-1} during the next control step. This is done all along the saturation time.

4.5 Simplification of the modeling

In Section 3.5, it was mentioned that the complete robot model requires 96 degrees of freedom. Eighteen of them are from the derivative action of PID controllers: the implementation of the PI controllers allows to drop them. Secondly, the controllers are digital, and do not require the extra DOFs that are used to obtain the integral of the error term. Thirdly, the motor inductance has been neglected. The final model thus includes 96-3×18 = 42 configuration parameters: 24 for the robot kinematics, and 18 for the internal state variables of the LuGre model.

CHAPTER 5

Friction identification

The choice of the LuGre model has been justified in Section 3.4.3. This section presents the methods used to determine friction parameters, and, more precisely, friction parameters that are required for validation of simulation in Chapter 6. For example, friction at low velocity is only important for the vertical actuator, because during the support phase they keep a constant position, thus a null velocity. But it would be unnecessary to identify accurately the breakaway forces for horizontal or rotational actuators (at least for the test case presented in Section 6.3), because they are always moving in the considered cases.

Some assumptions are first made in Section 5.1. Steady-state friction curve is identified in Section 5.2 for middle and high velocities. This range of velocities corresponds to those encountered during gait. Low velocity behavior concerning the vertical actuator is more deeply investigated in Section 5.3. The influence of the leg load on friction parameters for the horizontal joints is studied in Section 5.4. Remarks on some previous work are made in Section 5.5.

The second purpose of this section, in addition to the identification of friction parameters, is to implement a ground contact detection via the current measurement in the vertical motor. This part represents an originality of this thesis. The principle is quite basic: the difference between the real joint torque and its estimation, based on a comprehensive leg model, should vanish excepted when there is a contact of the foot with an obstacle. This theoretical principle is applied in Section 5.6 for the ground detection. Section 5.7 ends the chapter with a brief summary.

5.1 Preliminary assumptions

Assumptions are the following:

- Friction has been identified in each joint separately.
- Gearboxes, ball screws, friction bearings and revolute joints of the pantograph all

include friction. The identification and the modeling are made for the whole system: the aim is to find an equivalent friction torque at joint τ_f depending on the joint velocity $\dot{q}$ and the bristle state z:

$$\begin{aligned} \dot{z} &= \dot{q} - \frac{|\dot{q}|}{g(\dot{q})} z, & (5.1) \\ \tau_f &= \sigma_0 z + \sigma_1 \dot{z} + f_v \dot{q} & (5.2) \end{aligned}$$

- The steady-state function $g(\dot{q})$ takes the following general form:

$$g(\dot{q}) = \frac{\tau_{f,d} + (\tau_{f,s} - \tau_{f,d}) \exp\left(-(\dot{q}/\dot{q}_{st})^2\right)}{\sigma_0} \qquad (5.3)$$

For the three kinds of joint, dry and viscous parameters $\tau_{f,d}$ and f_v will be identified for unloaded leg, with the method described in Section 5.2. However, depending on the joint nature, the operating conditions during walking are quite different. Consequently, the identification procedure with unloaded leg will be completed by the following:

– Because the position of vertical joint remains constant during the support phase, provided that the robot keeps its altitude and attitude unchanged (see Section 7.1 for definitions of altitude and attitude), friction behavior at zero velocity is important. Identification of breakaway friction torque $\tau_{f,s}$ under severe load condition (there is approximately 10 kg at foot when tripod gait is considered) must be investigated. This point is described in Section 5.3.

– The horizontal and rotational joints are always moving, so that stiction is less important. Consequently Eq. (5.3) is simplified into:

$$g(\dot{q}) = \frac{\tau_{f,d}}{\sigma_0} \qquad (5.4)$$

But, during support phase, the leg load is transferred to the horizontal joint, increasing friction parameters. Equations (5.2) and (5.4) are rewritten as:

$$\begin{aligned} \tau_f &= \sigma_0 z + \sigma_1 \dot{z} + \tilde{f}_v \dot{q} & (5.5) \\ g(\dot{q}) &= \frac{\tilde{\tau}_{f,d}}{\sigma_0} & (5.6) \end{aligned}$$

where $\tilde{f}_v$ and $\tilde{\tau}_{f,d}$ are the viscous and dry friction **under load**. Study of this phenomenom is detailed in Section 5.4.

Some parameters of the model will not be identified:

- For the vertical joint, expression of Eq. (5.3) requires the Stribeck's velocity $\dot{q}_{st}$. However, the repeatability of this measurement has been reported very bad [GROT02]. Moreover, its influence on the dynamic behavior of the robot could be neglected, because the vertical joint is either at rest, or moving quite fast. We assume that joint velocities are high enough with respect to the Stribeck's zone.

	q_r	q_v	q_h
$\tau_{f,s}$	×	yes	×
$\tau_{f,d}$	yes	yes	yes
$\tilde{\tau}_{f,d}$	×	×	yes
f_v	yes	yes	yes
$\tilde{f}_v$	×	×	yes

Table 5.1: Summary of the friction parameters identification

- The bristle stiffness and damping require very accurate material to be identified. For example, 22-bit resolution encoders are used in [FERR03] to get satisfying results . We choose values based on the reasoning of Olsson [OLSS96]. He suggests to take the bristle stiffness σ_0 high enough to not disturb the friction behavior at steady-state, and to have a damping respecting the passivity condition (Eq. (3.83)).

These parameters are thus fixed to obtain satisfying simulation results in Chapter 6. Table 5.1 gives a summary of the parameters to identify for each leg. Even if a load at foot should have an influence on the rotational joint, the latter is not identified. This joint is mechanically better designed than the horizontal one, which makes it less sensitive to the load during the walk. It will be confirmed by plots of current in Chapter 6. Moreover, this identification would require an experimental setup similar to a treadmill, where the load remains parallel to the rotation axis while it is moving. Friction is not necessarily symmetric: consequently, the list of parameters of Table 5.1 should be identified for the two senses of each joint. This leads to a total number of 96 parameters to identify.

5.2 Steady-state curve identification for leg without load

The main purpose of this section is to identify dry and viscous friction parameters $\tau_{f,d}$ and f_v respectively, in the case of an unloaded leg.

5.2.1 Method

A classical way to identify this steady-state friction curve is to impose to the joint a constant velocity profile such that inertia effects vanish [BONA06, GROT02]. Hence, the balance of torques applied on the joint is rigorously expressed as:

$$\tau_m = \tau_f + \tau_g + \tau_{cor} \tag{5.7}$$

where

- τ_m is the motor torque;
- τ_g is the gravity contribution of the leg parts;

- τ_{cor} are the Coriolis and Centrifugal effects (negligible – see Section 4.3);
- τ_f is the friction torque, which includes in the real-world:
  - the meshing friction[1] which causes a cyclic position-dependent friction torque: this phenomenon arises with gears and is widely discussed in [GARC02] for gearboxes, but also exists for sprocket chain transmission;
  - any other mechanical defects: parallelism default between the sprocket wheels or between the screw and the linear guide, bad centering of screw or/and linear guide, flexibilities of the transmissions and pantograph which deform the leg mechanism and increases friction.

The friction identification does not include the meshing friction and does not take into account the mechanical defects.

The gravity contribution τ_g is position dependent. It is computed in real-time during the identification, by using Eq. (3.37) applied to a single leg model. Already note that the gravity contribution (in terms of motor current) is null for the rotational DOF q_r, and quite low for q_v and q_h ($< 5\%$). But, for the sake of a general methodology, they are nevertheless computed.

The motor torque τ_m is computed by Eq. (3.90), where i is measured with a current transducer inserted in the motor circuit (see Section 2.3.2).

Constant velocity profiles that have been imposed to the joints are such that they travel completely their operating ranges. Table 5.2 gathers the maximum and minimum joint positions, as well as the reference velocities investigated for friction identification. Joints positions are given according to the leg description of Fig. 3.7.

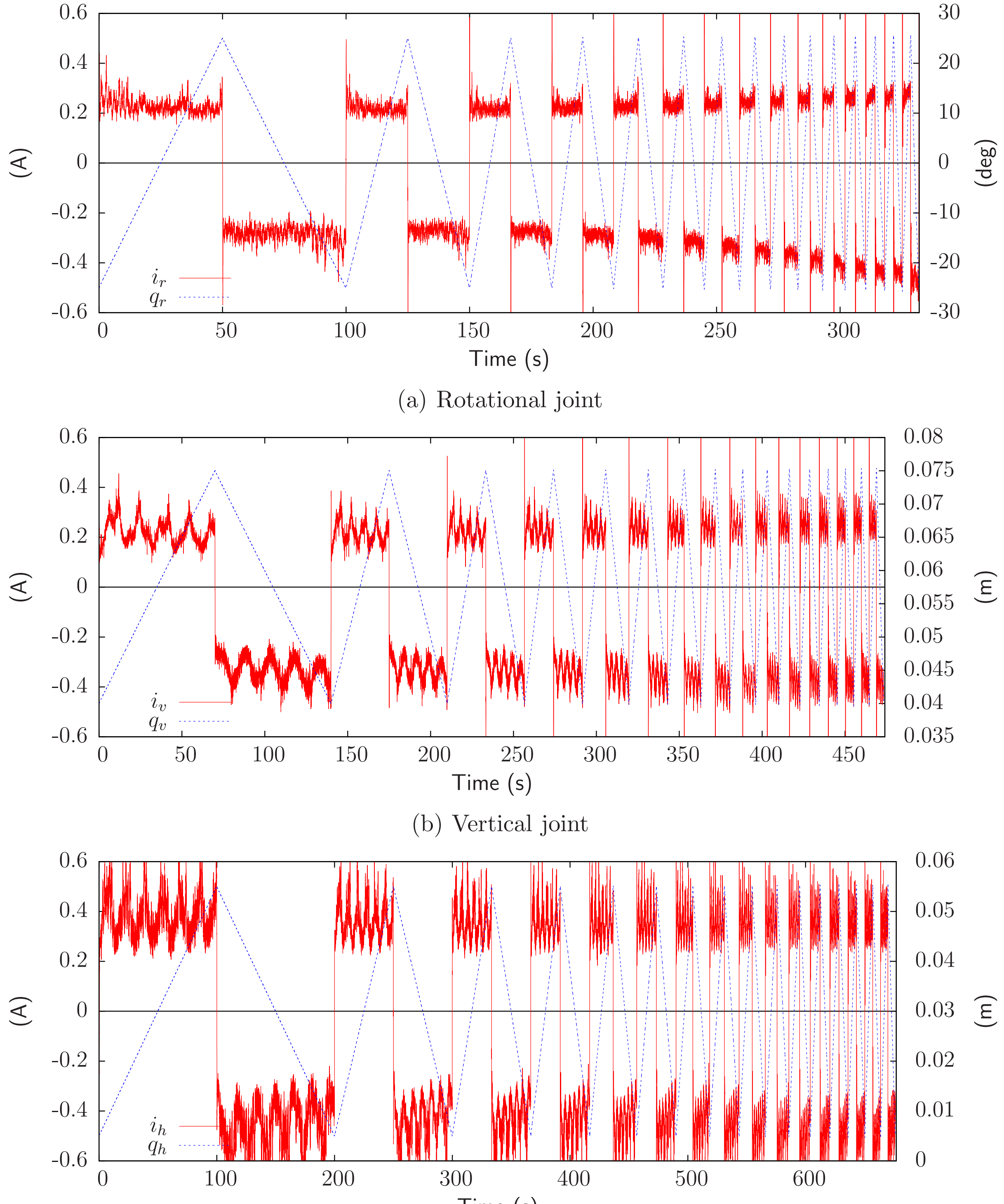
	q_r (deg)	q_v (m)	q_h (m)
Min	-25	0.04	0.005
Max	25	0.075	0.055
	$\dot{q}_r$ (deg/s)	$\dot{q}_v$ (mm/s)	$\dot{q}_h$ (mm/s)
Min	1	0.5	0.5
Max	16	8	8
Increment	1	0.5	0.5

Table 5.2: Velocities setup for steady-state curve identification

5.2.2 Measurements

Figures 5.1 and 5.2 show the measurements of current in the three actuators for legs 0 and 2, in the case of sawtooth profiles given in Table 5.2. The evolution of joint position is also represented on the graph. A complete set of measurements for the six legs are given in Appendix C.

[1] The term “meshing friction” refers to the rate-dependent friction occurring in meshing teeth of the transmissions [GARC02]

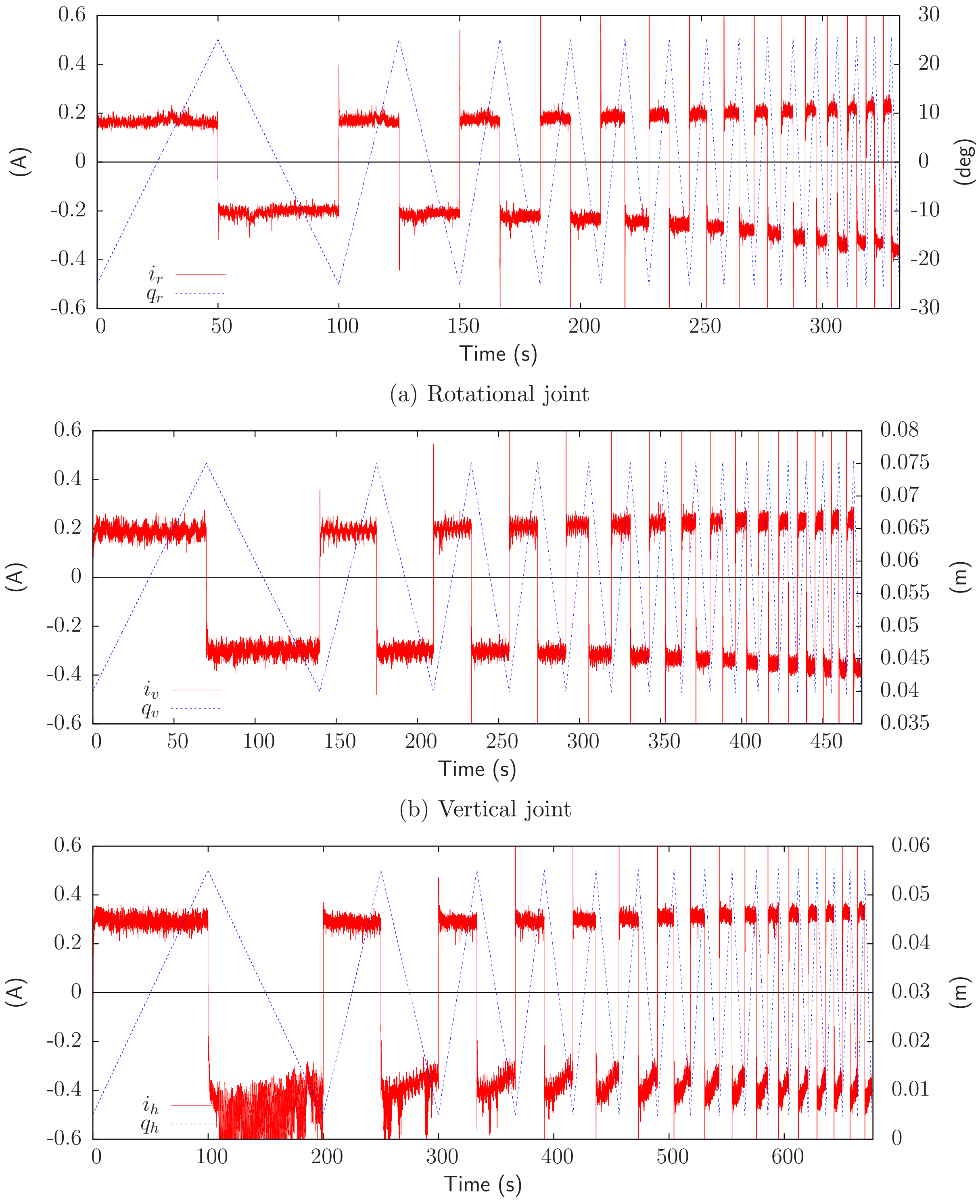


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure 5.1: Current measurements for leg 0



(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure 5.2: Current measurements for leg 2

As expected, the mean value of the current is approximately constant on a section, whatever the actuator and the leg. Also, at each velocity reversal, a current peak occurs. It may exceed the range of the Hall's effect sensor which measures current (2 A).

Examination of Fig. 5.1(a) and 5.2(a) reveals that currents for rotation are quite similar from level and noise point of view. The current global level is increasing faster for negative velocities than for positive ones. This trend is most of the time the same for all the actuators, and confirms that friction is an asymmetric phenomenon.

Vertical actuators presented in Figs. 5.1(b) and 5.2(b) are quite different. Oscillations appear on leg 0 while current on leg 2 is constant on each section. Moreover, the number of oscillations depends on the traveled distance of the joint, and not on the velocity. Comparison of measurements on the six legs in Appendix C shows that the number of cycles is different from one leg to another. Therefore, the source of these oscillations is quite difficult to identify, and should probably be a combination of the meshing of the closure chain link in the sprocket wheels and parallelism error between the two sprocket wheels of the transmission.

The measurements for horizontal joints of Figs. 5.1(c) and 5.2(c) are qualitatively identical to those of the vertical joints, which could be expected because transmission elements of these joints are similar. The only difference lies in the linear guide, which is ensured by ball-screws for the vertical joints and by friction bearings for the horizontal ones. Consequently, stick-slip arises for low velocities as represented in Fig. 5.2(c) between $t = 100$ and $t = 200$ s. The noise emitted by the joint leaves no doubt about it. It is also noted that current slightly decreases on each section for negative velocities, while it remains constant for positive velocities. This time, combination of parallelism error between the linear guide and the screw, and pantograph flexibility could explain this phenomenon. But, as before, each leg presents a different behavior, which makes difficult the interpretation of measurements.

5.2.3 Friction/velocities curves

To obtain the friction/velocity characteristics for unloaded leg, the following method is used:

- withdraw the contribution of pantograph weight on the current measurements;
- average current in each section by omitting the beginning where there is a transient in the velocity tracking due to the controllers – inertia effects are thus different from zero;
- compute the friction torques relative to the current measurements;
- plot the friction torque/velocity curves;
- perform a linear fitting of the experimental values to obtain the dry friction torque and the viscous coefficient.

Friction/velocity characteristic curves are plotted in Fig. 5.3, 5.4 and 5.5 for the six legs, and for one measurements campaign. For rotation, Fig. 5.3(a) shows that viscous effect is present on most of the legs. Figure 5.3(b) shows that the viscous contribution is quite low, and that friction for leg 3 is almost twice higher than on the other legs. Also, the slope of legs 3 and 5 is slightly negative in Fig. 5.3(b).

The dispersion of measurements is more important for friction in the vertical joints, due to the transmission defects described in the previous section. The general trend is again the one of a dry-viscous model, but the curves related to the negative velocities (Fig. 5.4(a)) emphasize a slightly more important friction level than the positive ones (Fig. 5.4(b)).

Because of the nature of friction bearing, the friction model for the horizontal joints could be approximated by a pure dry friction. However, some legs exhibit a friction level which depends on the velocity, but not according to a viscous effect. (Fig. 5.5(a)). The simple dry/viscous model is consequently limited to fit these measurements, and often gives a non physical viscous coefficient.

All the described procedure has been completely automated: regular sawtooth motions and current measurements are performed simultaneously on six joints of the same nature. Then post-processing and fitting is executed within MATLAB. Thanks to the automation, a complete measurement campaign takes only about half an hour, while for identification on each joint individually it took about half a day.

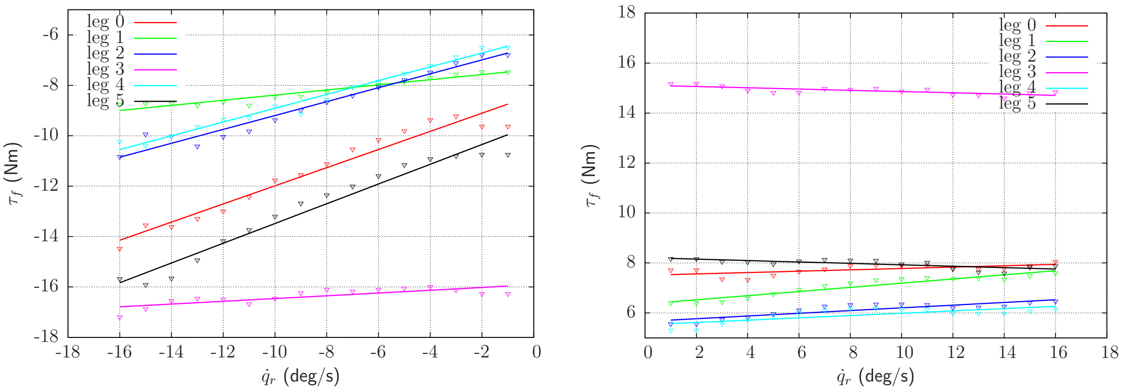
Table 5.3 gathers the parameters resulting from the steady-state friction identification for unloaded legs. The averaged value μ and the standard deviation σ have been computed for 5 measurements campaigns. The mean is given by:

$$\mu = \frac{1}{n}\sum_{i}^{n} x_i \tag{5.8}$$

where n is the number of measurements and x_i the measures. The standard deviation is computed according to:

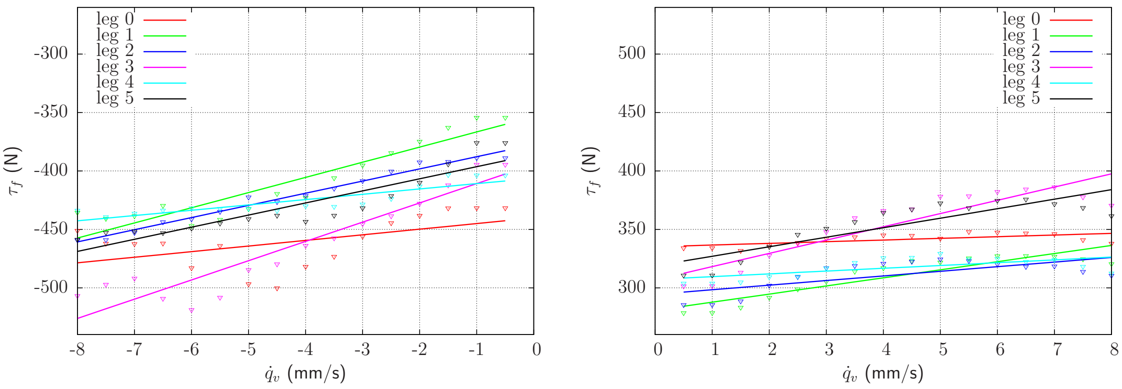
$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i}^{n}(x_i - \mu)^2} \tag{5.9}$$

The identification of the dry friction level is quite repeatable, as stated by the standard deviations which remain small with respect to the mean values. On the other hand, the identification of the viscous coefficient is less repeatable, especially for the horizontal joint.



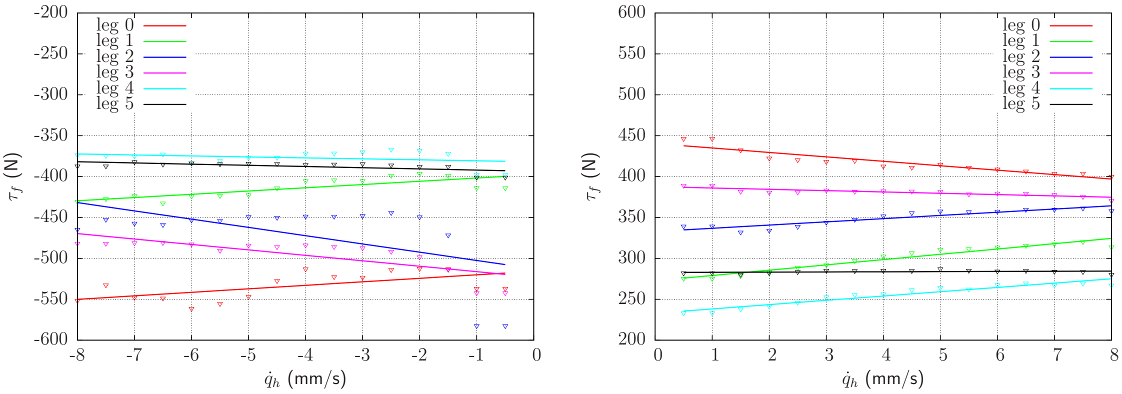
(a) Negative velocity (b) Positive velocity

Figure 5.3: Friction model for rotational joints (unloaded legs)



(a) Negative velocity (b) Positive velocity

Figure 5.4: Friction model for vertical joints (unloaded legs)



(a) Negative velocity (b) Positive velocity

Figure 5.5: Friction model for horizontal joints (unloaded legs)

q_r	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,d}^{+}$ (Nm)	7.7	0.2	6.3	0.1	5.6	0.1	14.9	0.3	5.3	0.2	8.3	0.1
f_v^{+} (Nms)	3.4	3.3	6.6	3.0	5.2	3.0	1.3	3.3	4.4	3.1	-0.3	3.3
$\tau_{f,d}^{-}$ (Nm)	-8.8	0.5	-7.3	0.3	-6.3	0.1	-15.6	0.2	-6.0	0.2	-9.9	0.3
f_v^{-} (Nms)	22.3	4.8	8.3	4.3	18.5	3.9	5.4	2.5	18.7	6.1	23.9	5.8
q_v	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,d}^{+}$ (N)	335.1	7.3	279.4	9.8	294.2	14.7	322.9	15.8	313.2	5.7	323.3	10.4
f_v^{+} (Ns/m)	3763.0	3715.6	8785.5	2450.5	6523.6	5114.6	12984.7	2840.1	5077.8	3892.9	10561.0	3546.6
$\tau_{f,d}^{-}$ (N)	-423.6	14.5	-351.6	3.5	-353.2	28.9	-381.8	10.9	-398.1	10.9	-383.8	2.1
f_v^{-} (Ns/m)	8005.8	5247.6	12820.8	192.1	18270.9	13036.5	20914.3	4523.2	6709.7	2775.0	14921.5	4291.2
q_h	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,d}^{+}$ (N)	465.1	22.2	279.9	8.3	329.1	5.3	381.3	10.6	231.0	2.2	284.8	2.7
f_v^{+} (Ns/m)	-4112.7	1364.8	7424.8	1152.6	3745.2	251.2	220.8	1827.2	6644.3	1716.8	1912.7	2199.3
$\tau_{f,d}^{-}$ (N)	-563.1	41	-399.3	3.6	-481.0	29.5	-508.4	18.0	-374.1	9.3	-391.9	4.5
f_v^{-} (Ns/m)	247.5	4909.7	7018.9	2883.5	-4556.3	6500	-5028.6	2108.0	405.4	1598.1	149.8	1764.6

Table 5.3: Dry and viscous parameters for the 18 joints (unloaded leg)

5.3 Vertical joints

AMRU5 becomes self-supporting when power is removed: consequently, it could be interesting to model the friction at zero velocity to obtain this behavior in simulation.

Experiments have been made by increasing slowly the voltage of **vertical actuators** with a rate of 0.05 V/s. Stiction has been identified on the six legs, for both senses of motion, and with/without a load of 10 kg placed at the foot: this load plays the role of the ground reaction force when the robot is walking. For a tripod gait, load on supporting legs is $\approx$ 11 kg each. Figure 5.6 illustrates the loading of a leg.

Assuming that τ_a is the motor current just before beginning of the motion (it is computed directly through a current measurement with $\tau_a = k_t i_v$), the equivalent breakaway torque $\tau_{f,s}$ can be found with the force balance represented in Fig. 5.6(b):

- for a positive voltage rate:

$$\tau_a = \tau_{f,s}^{+} - \tau_{leg}\ (+\ \tau_{load}) \tag{5.10}$$

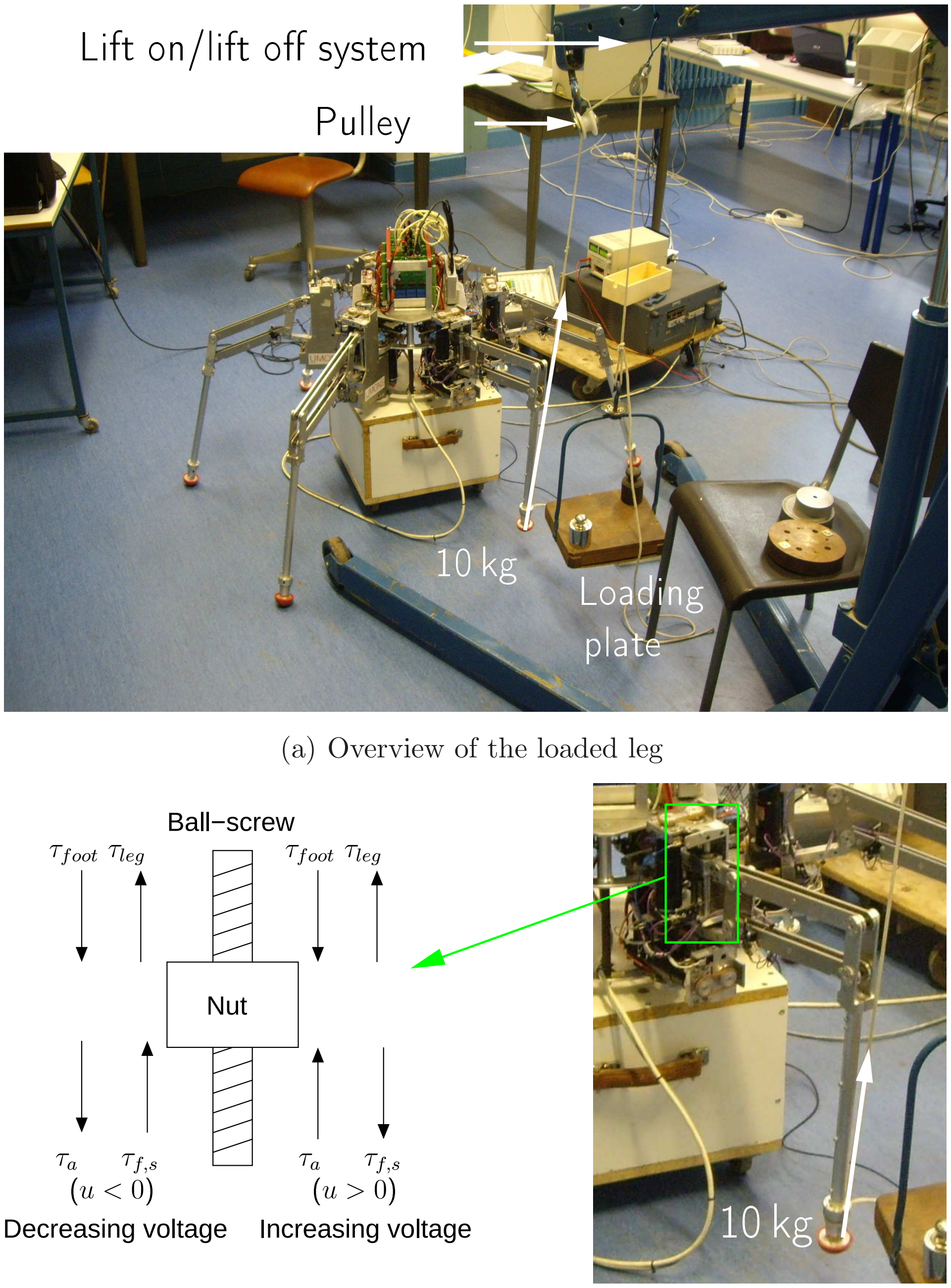
- for a negative voltage rate:

$$\tau_a = \tau_{f,s}^{-} + \tau_{leg}\ (-\ \tau_{load}) \tag{5.11}$$

where τ_{leg} is the gravity contribution brought back to the joint (=25.7 N), and τ_{load} is the influence of the foot load (=10 kg × 9.81 m/s^2 × 6 (pantograph amplification) = 588.6 N).

Figure 5.7 presents the time history of the currents and velocities for stiction identification on leg 0. Currents are presented in plain lines, and estimation of the velocity in dotted lines. In all cases, currents are increasing linearly up to the breakaway limit of the joint. Then velocity becomes different from zero. The measured current just before motion represents the breakaway torque. This measurement is quite difficult, because very sensitive to all the transmission elements. The breakaway instant has been chosen when the joint begins to move “frankly”, as indicated on Fig. 5.7.

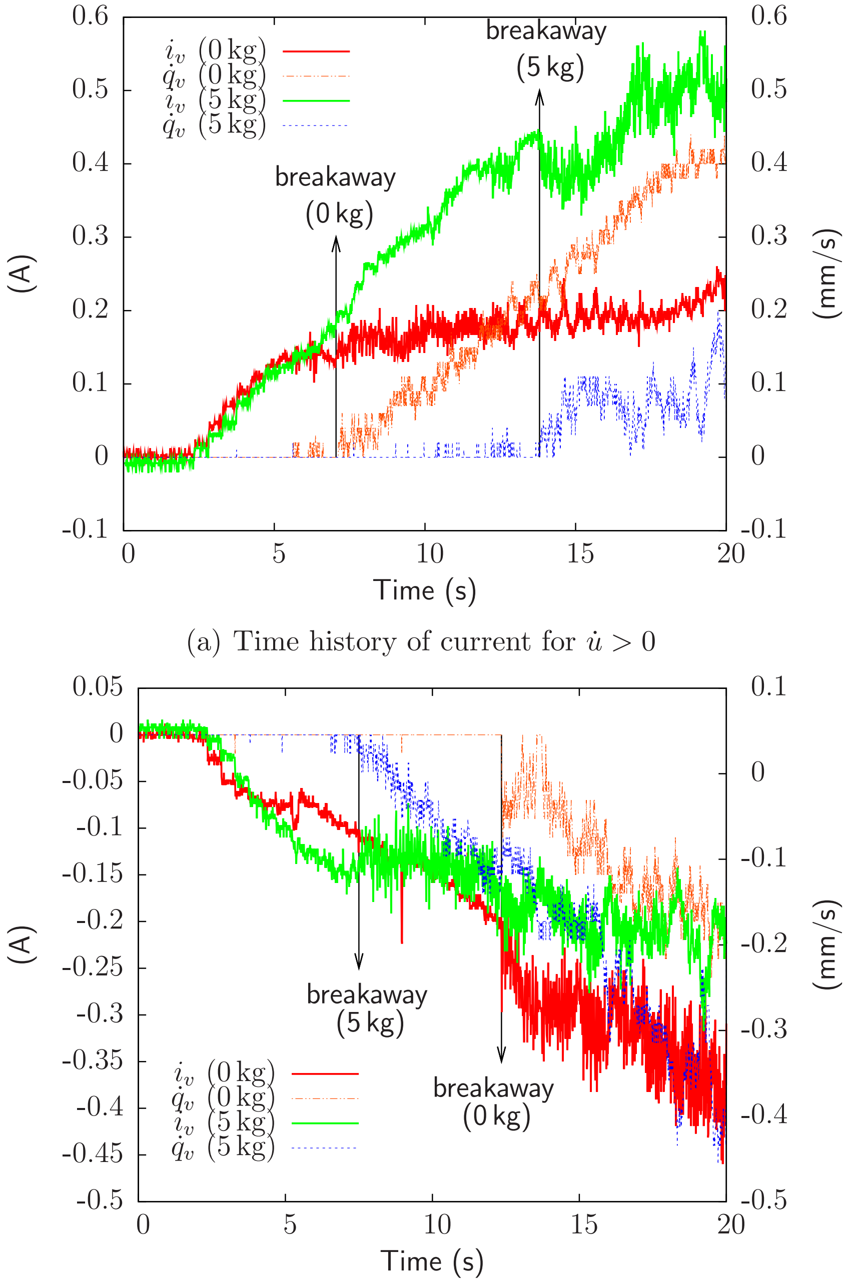
The identified breakaway torques are gathered in Table 5.4, with their mean value and their standard deviation, based on five measurements for each joint. Note that for some actuators, it has been impossible to find this parameter, for two reasons: 1) joint is reversible or 2) very low values of voltage are impossible to apply because of the H-bridge, which is not activated if the pulse width of the PWM signal is lower than 0.1 μs (this corresponds to 0.1 V and explains the flat shape of the current during the first two seconds); breakaway might occur under this limit. Note that for some joints, the level of breakaway torque is lower than dry friction.



(a) Overview of the loaded leg

(b) Zoom on the vertical joint forces

Figure 5.6: Loading the leg with 10 kg for stiction identification



(a) Time history of current for $\dot{u} > 0$

(b) Time history of current for $\dot{u} < 0$

Figure 5.7: Stiction identification (leg 0)

0 kg	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,s}^{+}$	166.6	46.3	169.1	38.5	154.2	43.5	201.2	55.6	162.9	24.2	141.1	14.1
$\tau_{f,s}^{-}$	-347.3	78.9	-236.0	92.0	-327.5	116.7	-318.9	48.9	-356.8	63.5	-316.4	70.5
5 kg	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,s}^{+}$	331.7	59.3	316.2	46.1	277.9	61.8	270.2	25.6	191.3	16.7	1010.4	55.2
$\tau_{f,s}^{-}$	-447.0	28.4	-397.1	13.8	-548.1	140.1	-372.4	7.7	-416.9	35.2	-435.8	31.9
10 kg	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
$\tau_{f,s}^{+}$	673.0	55.9	482.6	169.3	382.5	62.7	571.6	103.5	457.2	121.0	1431.9	84.8
$\tau_{f,s}^{-}$	-679.5	18.1	-	-	-601.8	13.7	-	-	-690.7	51.5	-663.6	11.2

Table 5.4: Stiction parameters for the 6 vertical joints (N), for different loading cases

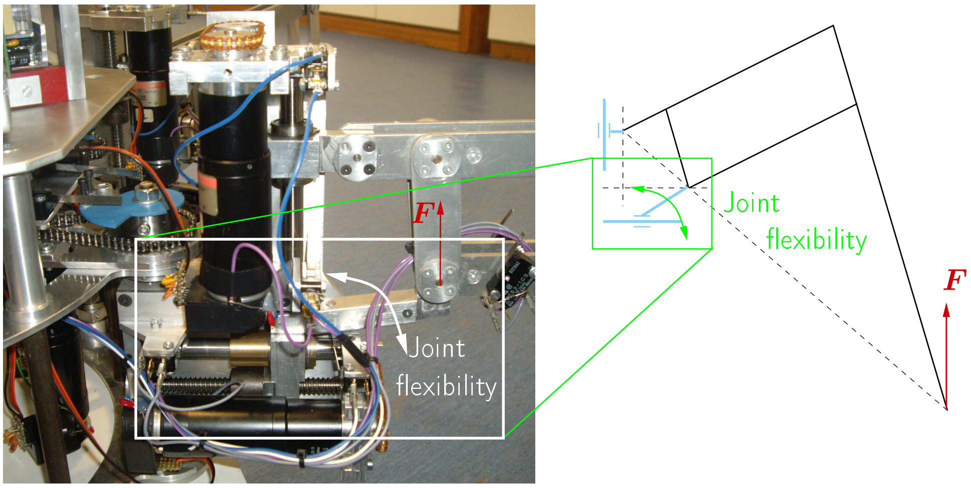


Figure 5.8: Flexbility in horizontal transmission

5.4 Horizontal joints

The bearings of the horizontal ball screws were initially identical to those of the vertical ones. However, practical use of AMRU5 (during its first steps) revealed an important weakness regarding force transmission on the bearing. In Fig. 5.8, it is shown that any load on the foot is not directly reported as a normal force, but as a bending torque which increases the stresses on the bearing. Initially, the stresses were so important that the initial bearings lost their balls during walking, and by the way deteriorated the sliding axle. To keep the leg structure, friction bearings have been manufactured in replacement of the linear ball bearings, with a maximal length such that the initial stroke was conserved.

The friction behavior of horizontal joints is mainly due to dry friction. In this case, it is well established that sliding friction depends on the joint load. To emphasize this phenomenon, constant velocity profiles have been imposed to the horizontal joints with the configuration illustrated in Fig. 5.6, for 5, 8 and 10 kg at leg end. Figure 5.9 illustrates the increasing of current according to the load at leg foot.

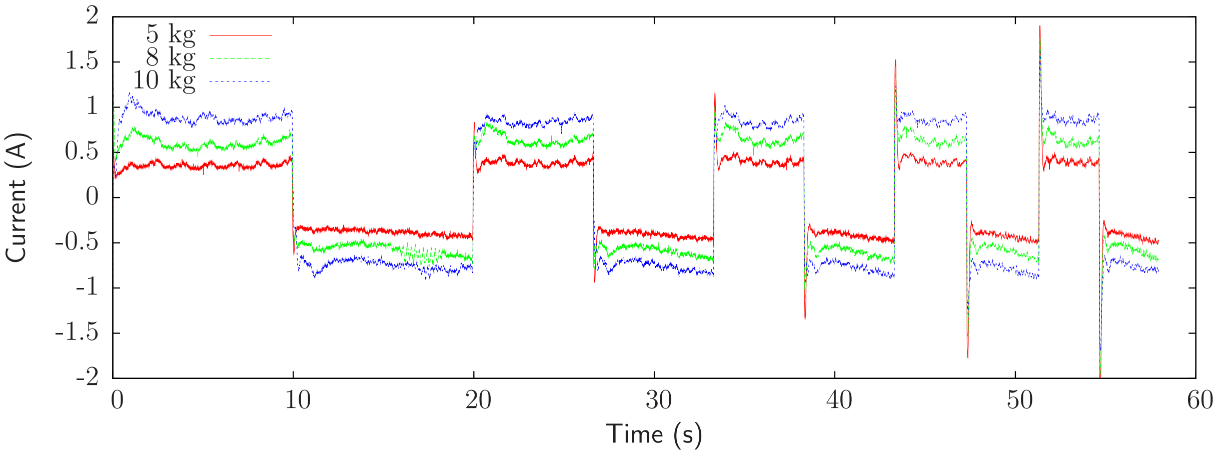
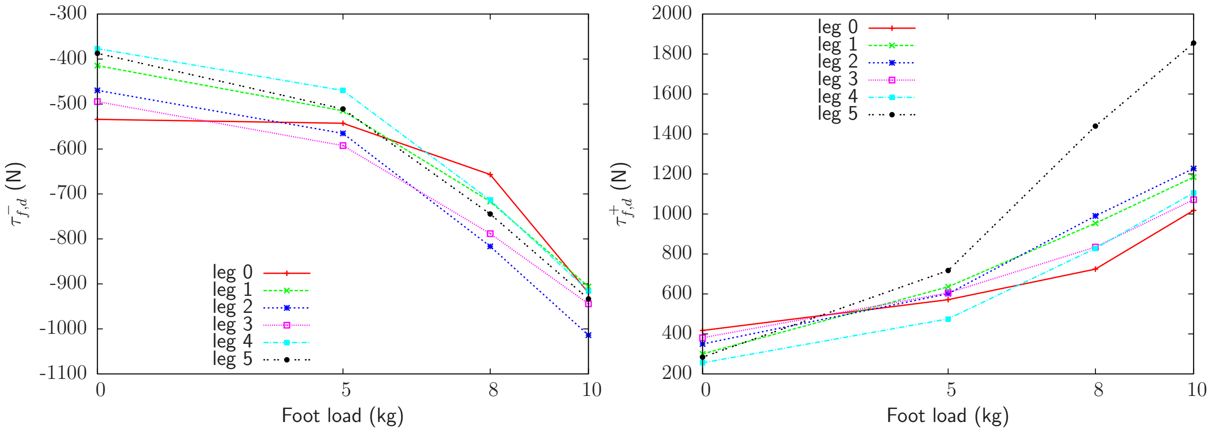


Figure 5.9: Influence of foot load on current in the horizontal actuators



(a) Negative velocities

(b) Positive velocities

Figure 5.10: Measured evolution of dry friction force versus foot load (horizontal joint)

The measured dry friction forces versus load on foot are depicted in Fig. 5.10. Practically, the procedure detailed in Section 5.2 has been applied on each leg, five times for each load, the only difference being that we consider only dry friction in the fitting, because viscous coefficient has been identified very low. This evolution is not linear as expected: it seems rather to follow a quadratic evolution. An explanation is that the bending torque acting on the joint (Fig. 5.8) induces a deformation of the linear guide with the leg load, making the friction still higher.

Practically, the evolution of friction has been fitted with a second order polynomial of the form:

$$\tilde{\tau}_{f,d} = a_2 \cdot x^2 + a_1 \cdot x + a_0 \tag{5.12}$$

where x denotes the load at leg, and a_0 the dry friction force identified in the case of an unloaded leg. Table 5.5 gives the coefficient a_2 and a_1 of the polynomials, as well as their mean value and standard deviation on five measurements. Note that for certain legs, the fitting leads to non-physical coefficients: they must be of the same sign to ensure that the function is always increasing with the force, and that there are no friction values lower (upper) than a_0^+ (a_0^-). In that case, the fitting has been done by plotting the friction torque versus the square of the load: this allows to drop the linear term without physical meaning.

Quadratic term (N/kg^2)	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
a_2^+	3.84	2.47	4.09	0.73	6.84	1.15	5.07	2.23	7.66	0.38	12.7	0.40
a_2^-	-3.84	1.66	-4.96	0.52	-5.6	0.68	-4.22	1.13	-5.52	0.44	-5.53	0.33
Linear term (N/kg)	0		1		2		3		4		5	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
a_1^+	20.3	18.7	48.0	22.7	21.47	5.10	17.62	8.46	9.44	6.35	34.61	15.93
a_1^-	0	0	0	0	0	0	-3.62	5.12	0	0	0	0

Table 5.5: Polynomial coefficients expressing the friction/load dependency

5.5 Remarks about previous work

It is worth noticing that all the identification procedure has already been made once before, with the initial ball bearings on the horizontal joints. The flexibility of the leg mechanism, and mainly the deformation of the horizontal joint, lead to higher friction parameters especially for stiction on the vertical joint: a reason is that load repartition is varying with the mechanism deformation.

With the replacement of ball bearings by friction bearings, it has been necessary to perform again the complete identification of friction parameters. The friction bearing brings more rigidity to the whole mechanism, but increases the friction on the linear guide. The results presented before are those of the new system.

5.6 Obstacle detection with current sensing

Ground detection is a key point for legged robot walking on uneven ground. The common solution to achieve this task is to use mechanical switches. They are of limited interest due to their single direction operating mode and their on-off operation. To obtain a better representation of the interaction forces in one, two or three directions, force sensors mostly based on strain gauges [ALEX97, BERN99, GASS01, SCHN06, GONZ07] or sometimes piezoelectric effect [GALV03] are usually embedded in the legs extremities. Other robots have such sensors directly inside their joints [ESPE96].

In this section, the feasibility of the motor current information to interpret contact with an obstacle is investigated. Unlike feet sensors, such measurements have the capability to inform when the leg encounters an obstacle, wherever the impact is located. They are also readily embeddable, widely spread and easy to implement in the DC motor hardware driver. However, one drawback is that all the dynamic effects between the joints and the foot have to be modeled. Also, high-geared motors give large torques for small currents, which makes the measurements very few sensitive to forces acting at foot. For example, if 100 N are applied vertically at foot, the equivalent current is approximately 0.4 A.

5.6.1 Principle

In the joint space of one leg (3×1), the joint torque balance is written:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{gnd} + \boldsymbol{\tau}_{iner} + \boldsymbol{\tau}_{grav} + \boldsymbol{\tau}_{fric} \tag{5.13}$$

where

- $\boldsymbol{\tau}$ are the joint torques;
- $\boldsymbol{\tau}_{gnd}$ are the contributions due to foot/obstacle contact;
- $\boldsymbol{\tau}_{iner}$ and $\boldsymbol{\tau}_{grav}$ are the contribution of inertia (including Coriolis and centrifugal effects) and gravity terms;
- $\boldsymbol{\tau}_{fric}$ are the contributions of friction already discussed in Section 5.1.

Assume that a dynamic model of the leg including friction can estimate the joint torque vector $\hat{\boldsymbol{\tau}}$ with:

$$\hat{\boldsymbol{\tau}} = \hat{\boldsymbol{\tau}}_{iner} + \hat{\boldsymbol{\tau}}_{grav} + \hat{\boldsymbol{\tau}}_{fric}. \tag{5.14}$$

Difference between Equations 5.13 and 5.14 leads to:

$$\boldsymbol{\tau} - \hat{\boldsymbol{\tau}} = \boldsymbol{\tau}_{gnd} + (\Delta\boldsymbol{\tau}_{iner} + \Delta\boldsymbol{\tau}_{grav} + \Delta\boldsymbol{\tau}_{fric}) \tag{5.15}$$

If contribution of modeling errors $\Delta\boldsymbol{\tau}$ and non modeled phenomenon $\boldsymbol{\tau}_{other}$ are not too important, we have:

$$\boldsymbol{\tau} - \hat{\boldsymbol{\tau}} \approx \boldsymbol{\tau}_{gnd} \tag{5.16}$$

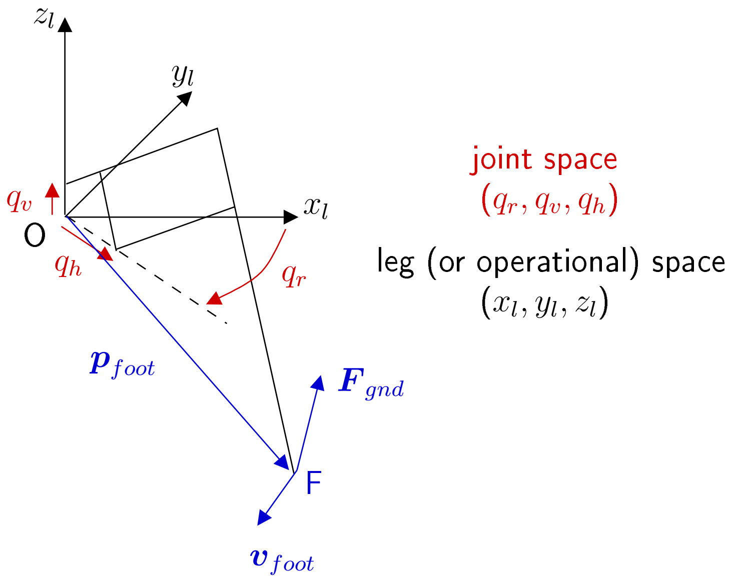


Figure 5.11: Leg and joint spaces

where the motor torques $\boldsymbol{\tau}$ are directly given by

$$\boldsymbol{\tau} = \boldsymbol{K} \cdot \boldsymbol{i}, \tag{5.17}$$

with $\boldsymbol{K}$ a diagonal matrix whose each element is the product between the torque constant of the motor and the reduction ratio between the joint and the rotor velocities.

Equation (5.13) can be transformed to find the ground reaction forces $\boldsymbol{F}_{gnd}$ (3×1) in the leg space. Consider Fig. 5.11: velocity of the foot is depending on the joint velocities by:

$$\boldsymbol{v}_{foot} = \frac{d\boldsymbol{p}_{foot}}{dt} = \sum_{j=1}^{3} \frac{\partial \boldsymbol{p}_{foot}}{\partial q_j} \cdot \dot{q}_j = \sum_{j=1}^{3} \boldsymbol{d}_{foot,j} \cdot \dot{q}_j. \tag{5.18}$$

As for robotic manipulator, we introduce the operational space, whose base is assimilated to the leg base. Therefore, velocity of the foot with respect to the leg, and projected in the leg frame, can be expressed by:

$$\{\boldsymbol{v}_{foot/l}\}_l = [\boldsymbol{J}]_l \cdot \dot{\boldsymbol{q}} \tag{5.19}$$

where $\boldsymbol{J}$ gathers contributions of $\boldsymbol{d}_{foot,j}$. According to the duality principle between joint and operational space [SICI09], we have:

$$\{\boldsymbol{F}_{gnd}\}_l = [\boldsymbol{J}]_l^{T^{-1}} \cdot \boldsymbol{\tau}_{gnd} \tag{5.20}$$

5.6.2 Torque estimation mechanism

Practically, a leg model has been developed to compute the joint torque estimation. The mechanism is highlighted in Fig. 5.12. Inputs are positions coming from incremental encoders; velocities and accelerations are numerically derived from the latter. Friction effects are directly deduced from computed velocities by means of a simple dry/viscous

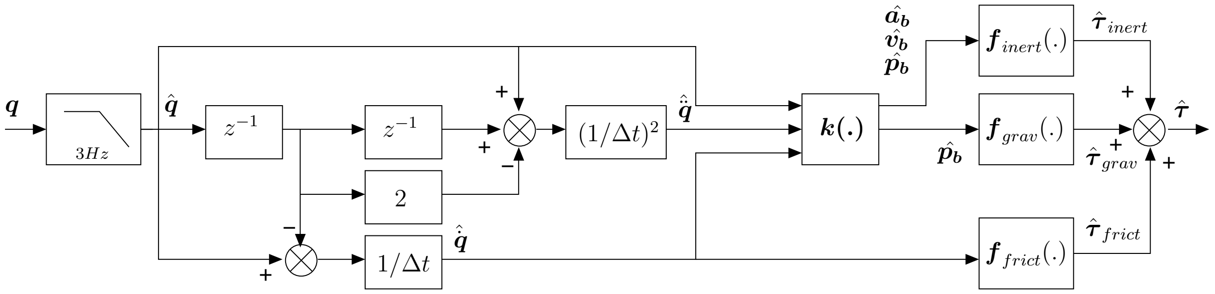

$\boldsymbol{q}$
$3Hz$
$\hat{\boldsymbol{q}}$
z^{-1}
z^{-1}
2
$1/\Delta t$
$(1/\Delta t)^2$
$\hat{\dot{\boldsymbol{q}}}$
$\hat{\ddot{\boldsymbol{q}}}$
$\boldsymbol{k}(.)$
$\hat{\boldsymbol{a}}_b$
$\hat{\boldsymbol{v}}_b$
$\hat{\boldsymbol{p}}_b$
$\hat{\boldsymbol{p}}_b$
$\boldsymbol{f}_{inert}(.)$
$\boldsymbol{f}_{grav}(.)$
$\boldsymbol{f}_{frict}(.)$
$\hat{\boldsymbol{\tau}}_{inert}$
$\hat{\boldsymbol{\tau}}_{grav}$
$\hat{\boldsymbol{\tau}}_{frict}$
$\hat{\boldsymbol{\tau}}$
+
−

Figure 5.12: Model-based estimation of the joint torques

	Sawteeth		**Sine**	
	Ampl. (m or deg)	Period (s)	Ampl. (m or deg)	Period (s)
q_r	10	8	15	5.6
q_v	0.01	8	0.01	10
q_h	0.01	8	0.01	10

Table 5.6: Profile of each joint for joint torque estimation

model which has been identified in Section 5.2. Gravity and inertia forces acting on the leg are computed with the virtual powers principle described in Eq. (3.37), which requires positions, velocities and accelerations of all the bodies. They are computed via the leg kinematics (function $\boldsymbol{k}(.)$) and estimated kinematic variables $\hat{\boldsymbol{q}}$, $\hat{\dot{\boldsymbol{q}}}$ and $\hat{\ddot{\boldsymbol{q}}}$.

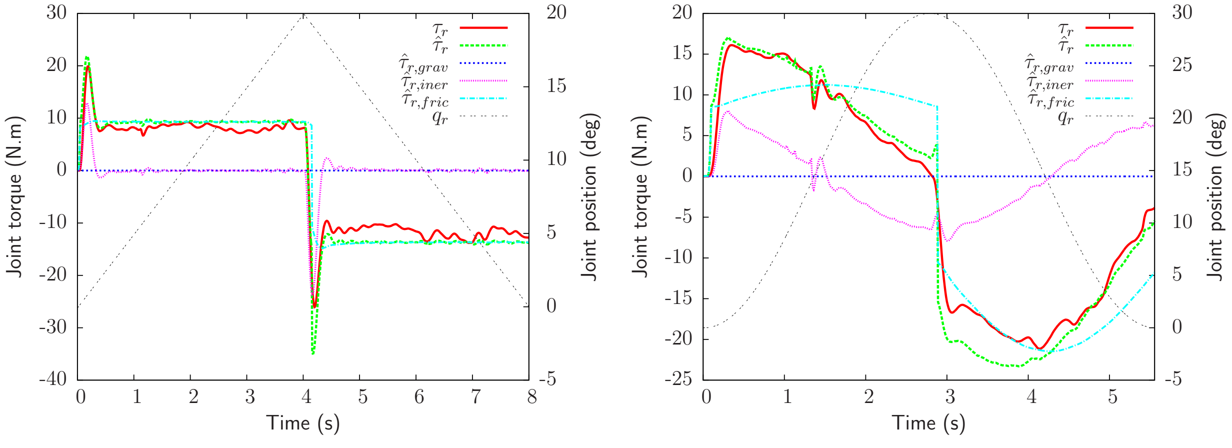
A simple numerical derivation of positions produces noisy estimations of velocities and accelerations, that are reduced thanks to a 3 Hz numerical low-pass filter at the input. The scheme of Fig. 5.12 takes roughly 0.1 ms of computation time per leg. The operator z^{-1} denotes a one sample time delay (discrete-time estimation).

To check the efficiency of the algorithm, sine and sawtooth unidirectional motions have been imposed to each joint separately. Characteristics of the motions are gathered in Table 5.6.

Figures 5.13, 5.14 and 5.15 show the model-based torque estimation and the real value measured from motor currents and computed by Eq. (5.17). Note that, in addition to the analog low-pass filter at 30 Hz, a digital low-pass filter at 3 Hz is required to smooth the current again. Without it, measurement noise is too important and disturbs the obstacle detection algorithm presented in Section 5.6.3.

Gravity contribution $\hat{\boldsymbol{\tau}}_{grav}$ is very small for q_v and q_h, and naturally null for q_r (if the robot is walking on a flat ground). For sawtooth cases presented in Fig. 5.13(a), 5.14(a) and 5.15(a), we can notice that inertia, Coriolis and Centrifugal effects vanish, excepted at velocity reversals, where estimations of joint torques are close to the real torques. Main contribution comes from friction effects, which are computed according to a dry-viscous model with a linear approximation for two reasons:

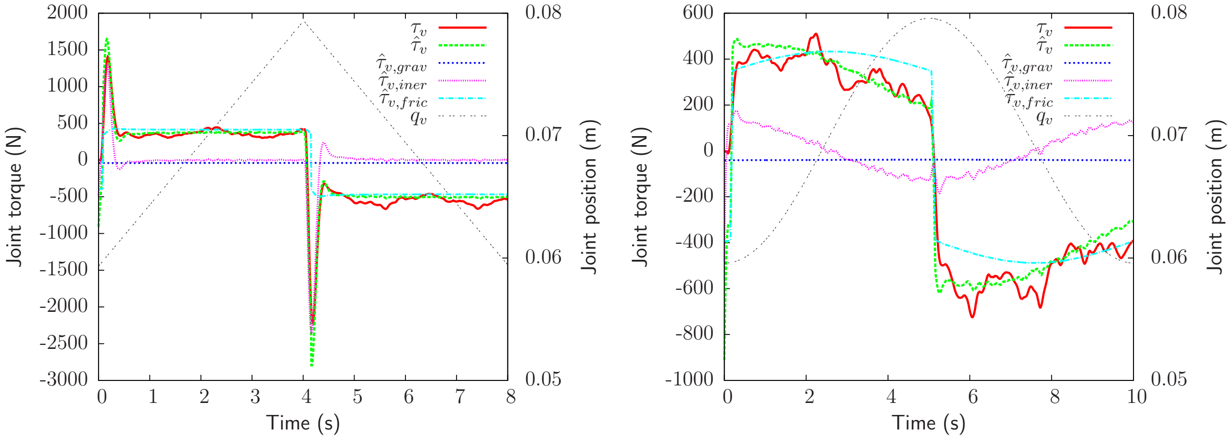
- this algorithm is executed in real-time when the robot walks;
- low velocity behavior is not relevant in the obstacle detection, because leg is moving quite fast during the landing phase.



(a) Sawtooth motion

(b) Sine motion

Figure 5.13: Torque estimation at joint q_r



(a) Sawtooth motion

(b) Sine motion

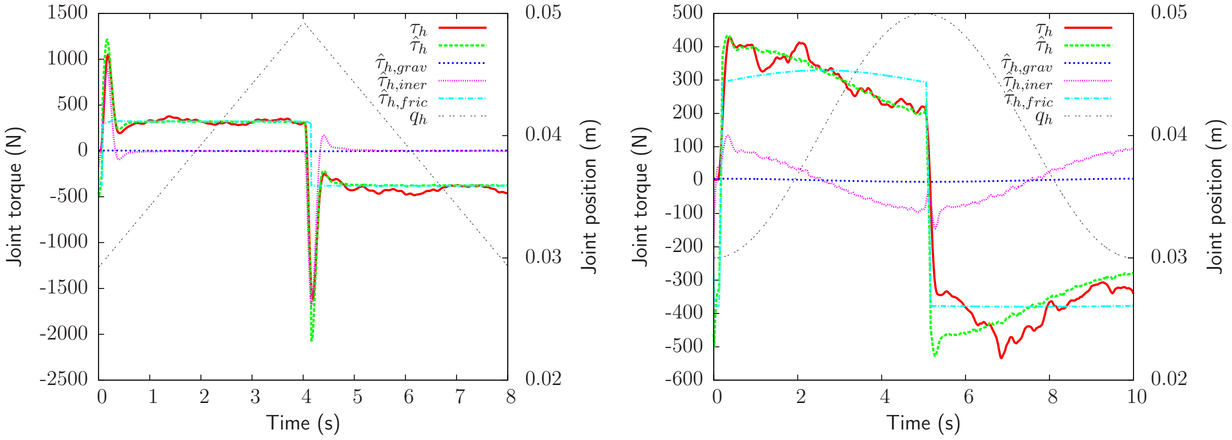
Figure 5.14: Torque estimation at joint q_v

Sine motions illustrate the importance of the dynamic effects $\hat{\boldsymbol{\tau}}_{iner}$ which are correctly reproduced by the model in Fig. 5.13(b), 5.14(b) and 5.15(b). Note, however, that the transmission elements are responsible for important current fluctuations, which are not caught by the model.

5.6.3 Ground detection

In the context of a free gait implementation (see Section 7.7), the robot must be able to detect the ground and adapt its posture consequently. Idea is to use the current sensing to detect ground. Because of the pantograph decoupling, developed form of Eq. (5.20) is:

$$\begin{pmatrix} F_{x,gnd} \\ F_{y,gnd} \\ F_{z,gnd} \end{pmatrix} = \begin{pmatrix} \alpha_{00} & 0 & \alpha_{02} \\ \alpha_{10} & 0 & \alpha_{12} \\ 0 & \alpha_{21} & 0 \end{pmatrix} \cdot \begin{pmatrix} \tau_{r,gnd} \\ \tau_{v,gnd} \\ \tau_{h,gnd} \end{pmatrix} \tag{5.21}$$



(a) Sawtooth motion

(b) Sine motion

Figure 5.15: Torque estimation at joint q_h

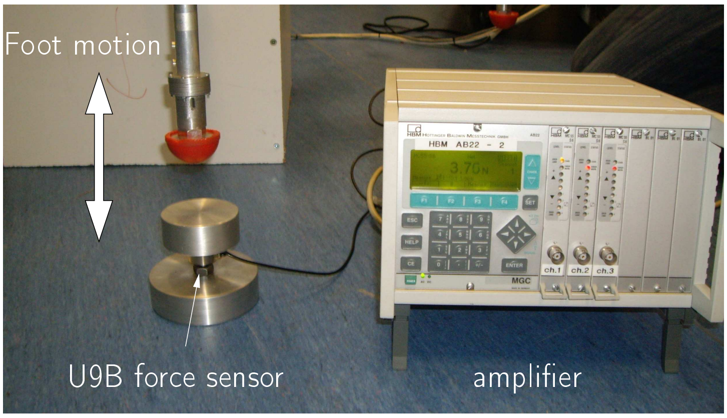


Figure 5.16: Experimental setup for ground detection

and simplifies, in the case of ground detection, into:

$$F_{z,gnd} = \alpha_{21} \cdot \tau_{v,gnd} \tag{5.22}$$

Ground detection has been tested in the following way: a sawtooth velocity profile is imposed to the leg, in such a way that ground is encountered by the foot during its descent. When ground contact force estimation attains a threshold value given by the user, the motion is stopped, and resumes as soon as the foot position reference becomes higher than the ground level. A mono-axial force transducer based on strain gauges (HBM-U9B) has been placed on the soil to measure directly the ground contact force. Experimental setup is shown in Fig. 5.16.

Figure 5.17 shows a comparison between the estimated and the real ground contact force on leg 0. Contact occurs at $\approx 2\,s$. When the estimated contact force $F_{z,gnd}$ reaches the value of $40\,N$, the motion stops (while the reference q_v^* is always computed), until the restarting of motion at $\approx 6\,s$. Note that estimation procedure goes on after the restarting of the motion, even if practically it is useless because the foot moves away from the soil. In fact, in addition to the threshold reaching, there is a condition on the joint velocity

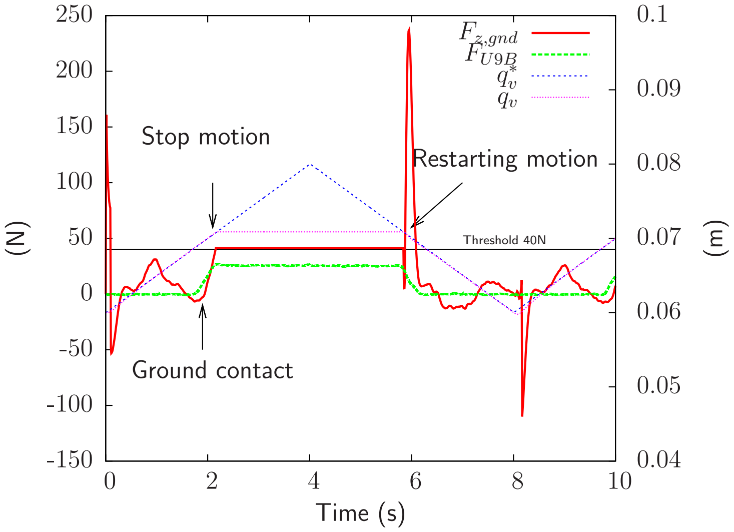


Figure 5.17: Comparison of estimated and real ground contact force for leg 0

which must be positive (positive joint velocity lowers the foot). If one of these conditions is not met, foot continues its motion.

Note on Fig. 5.17 that the meshing effect is quite visible between 0 and 2 s. The worse the mechanical quality of the leg, the higher the threshold level: indeed, for this experiment a threshold of 30 N would be too small, and the proposed algorithm would conclude that the leg is well on the ground while it is still in the air. This could have disastrous consequences in gait implementation. However, in the case of a tripod gait, the load on the leg is equivalent to approximately 100 N, which is high enough to catch the difference between disturbances of the transmissions and real ground contact.

Four tests of ground collision with three different thresholds (40, 50 and 60 N) have been made on each leg, to have a global estimation of the algorithm performance. Table 5.7 summarizes the real force measurement with the U9B transducer for each trial. The average $E(\epsilon_F)$ and standard deviation $\sigma(\epsilon_F)$ of the error between the real contact force and the one computed by the ground detection algorithm are also presented.

The method is quite repeatable, as emphasized by the standard deviation on the error. However, there are important discrepancies from one leg to another, mainly due to the meshing of the sprocket wheels. Because of the high reduction ratio of the joint transmission, a small variation of current has an important impact on the joint torque, thus on the ground force. The second point concerns the evolution of the average error with the load: in most of the cases, it increases, because the horizontal joint flexibility illustrated in Fig. 5.11 consumes motor energy: in other words, the term $\boldsymbol{\tau}_{other}$ of Eq. (5.13) becomes more important with the load, and then Eq. (5.16) becomes less and less valid.

Despite of important errors in contact force estimation, the proposed method remains applicable in the case of AMRU5 for ground detection, provided that sufficient threshold is defined in the gait algorithm.

	Threshold (N)	$F_{U9B}(N)$				$E(\epsilon_F)$ %	$\sigma(\epsilon_F)$ %
		1	2	3	4		
Leg 0	40	25.4	26.3	27.0	27.3	35	2
	50	32.1	29.9	31.5	30.5	38	2
	60	31.0	33.4	35.6	35.6	43	4
Leg 1	40	36.2	35.6	35.6	35.0	13	2
	50	39.7	37.9	40.4	39.5	22	2
	60	45.2	45.2	45.5	45.2	26	<1
Leg 2	40	33.3	34.0	32.0	32.0	19	3
	50	41.2	44.8	49.5	44.5	11	6
	60	52.8	56.1	54.8	56.5	9	3
Leg 3	40	38.2	36.6	34	32.3	13	7
	50	35.9	38.9	37.6	38.2	25	2
	60	42.8	43.2	42.8	43.9	29	<1
Leg 4	40	43.2	44.8	42.5	41.2	-5	4
	50	45.2	44.5	43.5	44.5	11.7	1
	60	48.1	50.1	48.8	50.1	20	1
Leg 5	40	25.3	36.3	39.0	39.5	13	16
	50	46.8	49.2	48.2	46.2	7	3
	60	50.8	49.5	51.2	51.5	16	1

Table 5.7: Measurements with U9B force transducer

5.7 Summary

This section has detailed the friction identification parameters for the eighteen joints. First, the same identification procedure has been made for the three kinds of joints: constant velocity profiles are imposed to each joint separately, with a foot in the air. The common steady-state friction versus velocity relation is deduced from this first experimental campaign. Then, the vertical and horizontal joints are more deeply investigated to derive additional parameters. Stiction (friction at zero velocity) is identified for the vertical joint because during the support phase, its velocity is very low and even null. To simulate the robot weight, a load of 10 kg has been placed at the foot. Horizontal joints undergo important deformations when robot is walking, increasing consequently the friction level. Hence, the friction/load dependency has been measured and fitted with a quadratic relationship.

The second part of the chapter has developed an original ground detection algorithm, based on a motor current sensing and a joint torque model-based estimation. This algorithm is implemented on the real robot for free gait (see Section 7). The mechanical defects of the legs gives a poor estimation of the ground contact force. A better design of the leg would lead to more reliable results in the ground force estimation.

CHAPTER 6

Validation of the AMRU5 model

The previous analysis about friction demonstrates that friction can not be neglected in the dynamic model of AMRU5 because it is responsible for an important current consumption. But it also illustrates the difficulty of friction identification on a “real system”, where almost each transmission should be characterized specifically. The purpose of this chapter is to validate the complete dynamic model of AMRU5.

Section 6.1 discusses about the friction models and the parameters used in the simulations. Section 6.2 illustrates how the LuGre model is able to produce an irreversibility of the joints when the robot is simply posed on the ground without power supply. Section 6.3 describes the test case used to verify the effectiveness of simulations.

In Section 6.4, both results coming from simulations and real experiments are analyzed. The identified parameters of Chapter 5 are directly included in the model. The joint tracking errors, motor currents and voltages, and energy expenditure are studied, for the three friction models. The aim of this study is to reproduce the evolution of current and voltage for each actuator with a maximum of reliability.

Section 6.5 describes a probabilistic approach developed to estimate a confidence interval which indicates how an error on friction influences the error on the energy expenditure and performances of the vehicle. A Monte-Carlo method is used: a large number of simulations are performed, where friction parameters are generated randomly according to a Gaussian distribution. The summary of Section 6.6 concludes this chapter.

6.1 Discussion about friction models and parameters for the simulations

Three cases are envisaged in the simulations: the first one does not involve friction at all. The second one is a classical dry/viscous model with linear approximations around zero velocity: the model is described by:

$$\tau_f = \begin{cases} \tau_{f,d}^{+} + f_v^{+} \cdot \dot{q} & \text{if } \dot{q} > \epsilon \\ (\tau_{f,d}^{+} + f_v^{+}\epsilon)/\epsilon \cdot \dot{q} & \text{if } 0 \leq \dot{q} \leq \epsilon \\ (\tau_{f,d}^{-} + f_v^{-}\epsilon)/\epsilon \cdot \dot{q} & \text{if } -\epsilon \leq \dot{q} < 0 \\ \tau_{f,d}^{-} + f_v^{-} \cdot \dot{q} & \text{if } \dot{q} < -\epsilon \end{cases} \tag{6.1}$$

This classical form allows to avoid the friction discontinuity at zero velocity: it facilitates the numerical implementation, but cancels the stiction effect. The coefficients used in Eq. (6.1) come from the steady-state curve identification of Section 5.2. The limit velocity ϵ is equal to 1e-4 m/s (or rad/s for the rotation), which is low enough with respect to the joint velocities (in the case of our validation test case).

The third set of simulations is performed with the LuGre model on each joint: it is implemented following Eqs. (3.85), (3.86) and (3.87), with some adaptation depending on the operating conditions of the actuator:

- Function $g(\dot{q})$ for a rotational joint is given by:

$$g(\dot{q}) = \frac{\tau_{f,d}^{+}}{\sigma_0}, \qquad \text{if } \dot{q} \geq 0 \tag{6.2}$$

$$g(\dot{q}) = -\frac{\tau_{f,d}^{-}}{\sigma_0}, \qquad \dot{q} < 0 \tag{6.3}$$

  Parameters are those of Table 5.3. The negative sign is necessary because $g(\dot{q})$ must be positive [OLSS96]. In the following, the ± superscript is abandoned for the sake of clarity. The implicit convention is that parameters with sign + are taken for $\dot{q} \geq 0$, and − for $\dot{q} < 0$.

- Function $g(\dot{q})$ for a vertical joint is given by:

$$g(\dot{q}) = \frac{\tau_{f,d} + (\tau_{f,s} - \tau_{f,d}) \exp\left(-(\dot{q}/\dot{q}_{st})^2\right)}{\sigma_0}. \tag{6.4}$$

  Parameters are those of Table 5.3 and of Table 5.4, where stiction parameters for the vertical joint are those for a loaded leg with 10 kg, because the zero velocity occurs while the leg is carrying the robot (for the case envisaged in Section 6.3). Stiction parameters which have not been identified (legs 1 and 3), are set to the same value as the dry friction for an unloaded leg. This stiction coefficient clearly depends on the load, but in this work, we keep it constant whatever the leg load.

- Function $g(\dot{q})$ for a horizontal joint is given by:

$$g(\dot{q}) = \frac{\tau_{f,d}}{\sigma_0}. \tag{6.5}$$

  where the dry friction torque is, in a first time, independent of the load at leg. This influence will be added afterwards and discussed in Section 6.4.4. Viscous effects are neglected because they have not been clearly identified.

<table>
<tr><th>Parameter</th><th>Rotation</th><th>Vertical</th><th>Horizontal</th></tr>
<tr><td>Bristle stiffness σ_0 (Nm for rotation, N/m for others)</td><td colspan="3">1e-9</td></tr>
<tr><td>Bristle damping σ_1 (Nms for rotation, Ns/m for others)</td><td colspan="3">0.02</td></tr>
<tr><td>Stribeck's velocity q_{st} (m/s)</td><td>-</td><td>1e-4</td><td>-</td></tr>
</table>

Table 6.1: LuGre parameters

- The non-physical viscous coefficients, i.e. the negative ones, are forced to zero.

The other unidentified parameters of LuGre model have been set to the values of Table 6.1.

We verified that the condition of Eq. (3.83) which concerns the dissipativity of the LuGre model is respected for any joint velocities considered in our simulations.

Parameters	Value
Gait type	Tripod
Step period	12 s
Straighforward velocity V_x	1, 1.5, 2, 2.5 cm/s
Number of steps	5
Double stance (or overlap)	5%

Table 6.2: Test case for the validation

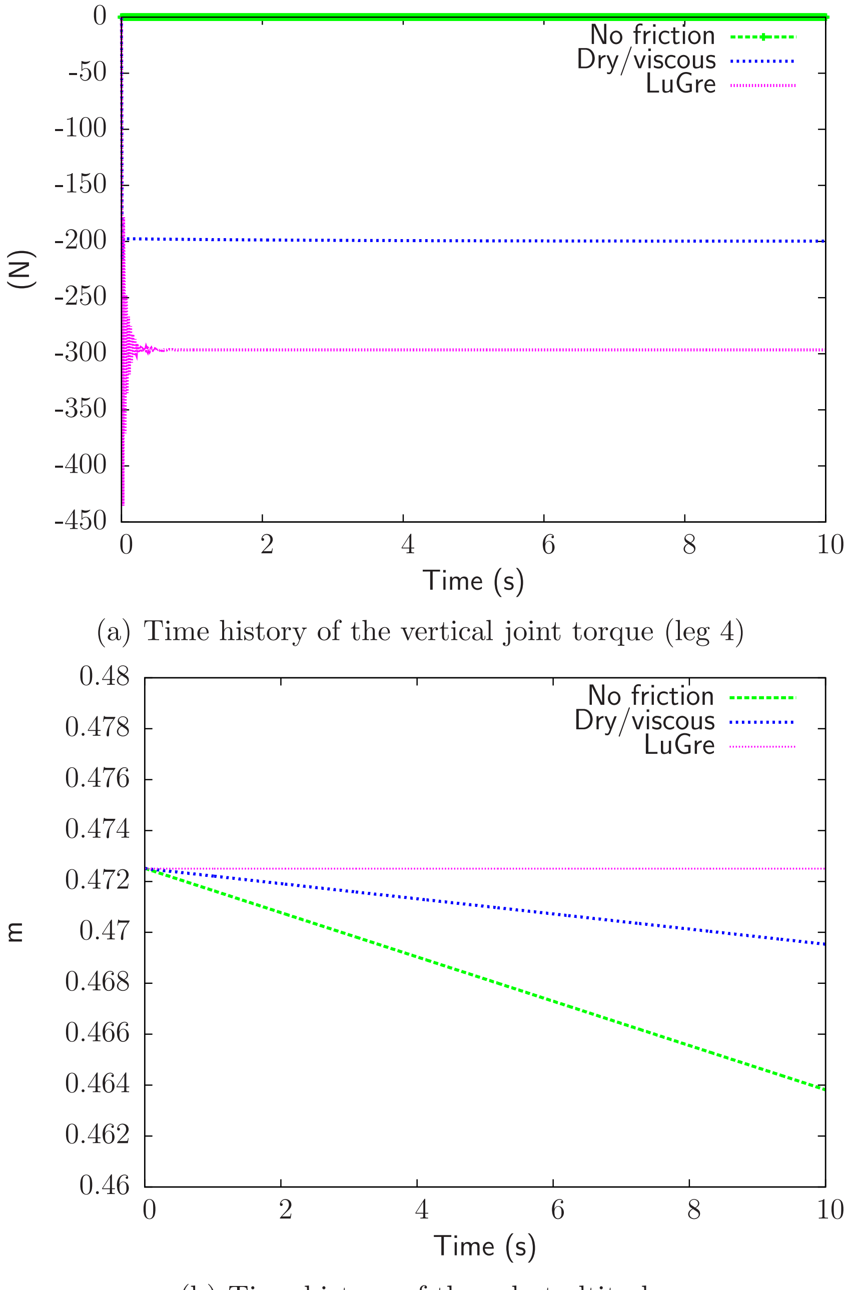
6.2 Simulation of the vertical irreversibility

The LuGre model has been chosen because it offers numerical facilities for simulation of friction around zero velocity. The first test presented simply consists in letting the robot on the ground with zero motors voltages. In the real-world, friction is high enough to keep the robot standing. In simulation, a static equilibrium is first performed with all the degrees of freedom locked, excepted those of the main body. Once the static equilibrium is reached, the integration begins. Figure 6.1(b) shows the evolution of the robot height for the three friction models described previously. When there is no friction, the robot height clearly decreases. This descent is quite low because the electrical circuits of the motors are closed, which provides a quite important damping to the system.

Adding dry/viscous model slows down the descent of the robot, but not as much as with the LuGre model, where the altitude remains constant. The time histories of the friction torques in the vertical actuators are presented in Fig. 6.1 for leg 4, according to the three models. Thanks to friction at low velocity with the LuGre model, the friction torque at joint is high enough to counteract the gravity. The bristle deformation for leg 4 is shown in Fig. 6.2(a) and the evolution of function $g(\dot{q}_v)$ with joint velocity is depicted in Fig. 6.2(b). The bristle oscillates at the very beginning of the simulation, then reaches its equilibrium around 5e-9 m.

6.3 Test case

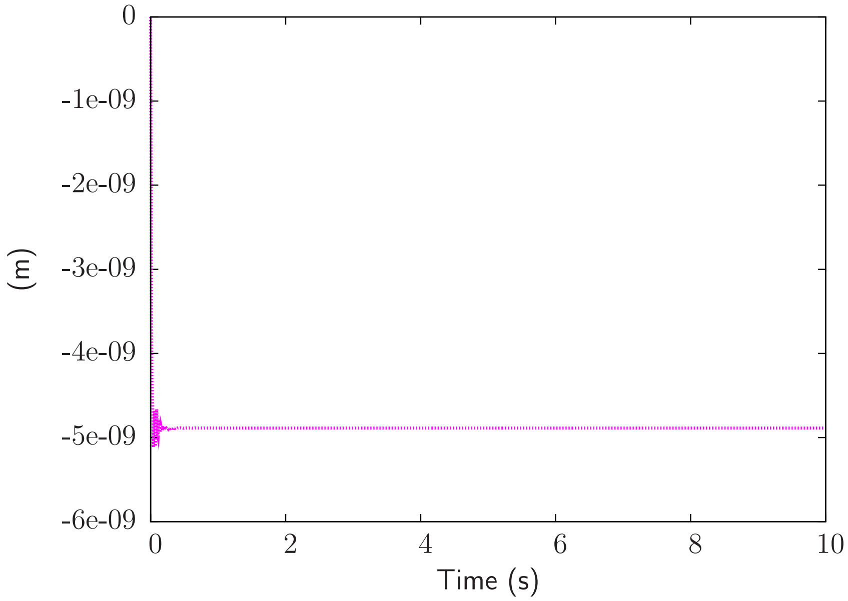
The parameters used for the test cases are given in Table 6.2. A sketch of the gait is shown in Fig. 6.3(a). The reference foot profiles of leg 3 are given in Fig. 6.3(b) in the XZ frame (leg local frame). These profiles are modified with the velocity of the robot, as the period for the step cycle remains constant. More details about the generation of such profiles are given in Chapter 7.



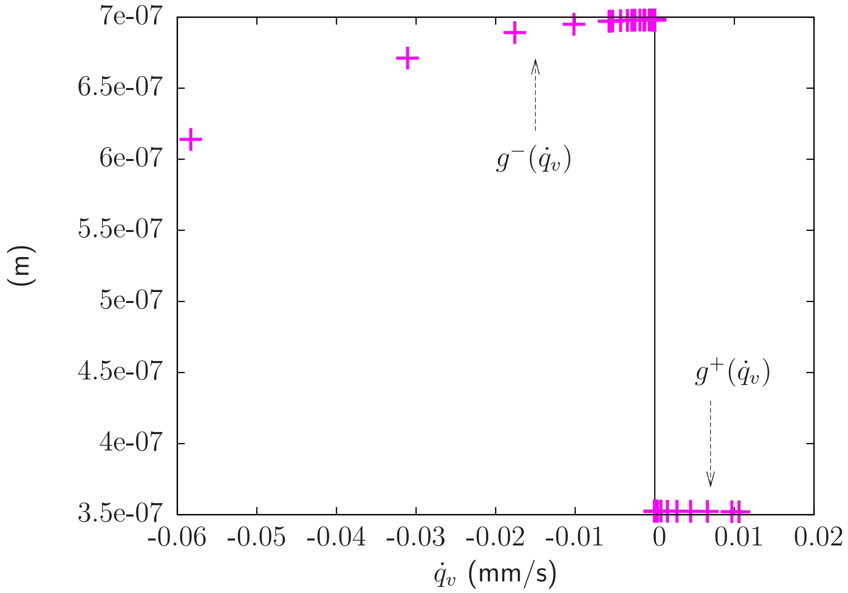
(a) Time history of the vertical joint torque (leg 4)

(b) Time history of the robot altitude

Figure 6.1: Simulation of AMRU5 no powered

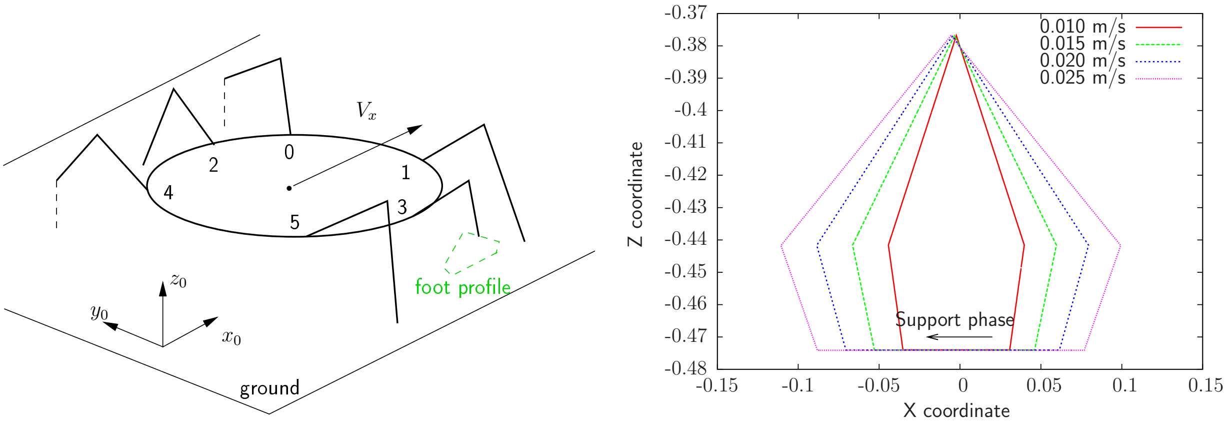


(a) Time history of the bristle deformation z



(b) Evolution of steady-state function $g(\dot{q}_v)$ with the joint velocity

Figure 6.2: Internal variables of the LuGre model (leg 4)



(a) Sketch of the test case

(b) Foot profile reference for leg 3 (zoom - XZ leg frame)

Figure 6.3: Illustration of the test case for model validation

6.4 Direct transposition of friction identification

The three actuators of legs 0 and 2 will be examined in the following, for a velocity of 1.5 cm/s. For each of them we plot the joint reference, the tracking error, the motor current and voltage. Note that the plotted voltages are not exactly the same in experiments or in simulation:

- **For experiments**, the voltages supplying the H-Bridge is directly measured, and the estimation of each actuator voltage is made by multiplying this value to the corresponding PWM duty cycle.

- **For simulations**, the output voltages of the controllers are applied to a simplified H-Bridge model, which consists of a resistance in series with the one of the motor armature coil.

We draw the reader's attention by the fact that there are non-modeled electrical parts:

- The wires on the robot have their own resistance, which is most of the time negligible but could be significant for high current peaks.

- The H-Bridge driving the motor has a complex behavior depending on the state of the PWM command. When MOSFET (metal-oxide-semiconductor field-effect transistor) are switched on (when PWM signal is high) to let the current flow through the LMD18200, a serial resistance of $\approx 0.6\,\Omega$ should be added in the circuit ($0.3\,\Omega$ per MOSFET). When they are switched off (PWM low), the recirculating current (inherent to H-Bridge) coming from the motor coil inductance crosses the protection diode, which gives a voltage drop equal to 1.2 V according to the LMD18200 datasheet.

In our model, we do not take into account the complete dynamics of the H-Bridge, but we simply add a serial resistance to the coil circuit, equal to $2{\times}0.3\,\Omega$.

Comparisons are made between experiments and simulations with three models of friction:

- without friction at joints;
- with a dry/viscous model such as the one of Eq. (6.1), without the load influence of the horizontal joint (this case will be examined apart);
- with the LuGre model with the steady-state functions described in Section 6.1.

For easier reading, the joint reference is at each time represented in the top sub-figure. The successions of support, landing, swing, takeoff and double stance phases of the foot have also been colored with different shades of gray.

6.4.1 Rotational joints

Figures 6.4 and 6.5 present the results obtained for the rotational joints of legs 0 and 2. Whatever the friction model, the tracking errors are the same for the three simulations, and are quite close to the experimental ones. Maximum error is about 0.5 degrees, while the range of the rotational joint varies between -25 and 25 degrees. It is evident that current consumption comes mainly from friction. There are few differences between the dry/viscous and the LuGre models. This is because the rotational joint is always moving in the test case we describe here, that is why friction at zero velocity is not relevant. Current is also better estimated during the swing phase, because identification of friction parameters has been made for an unloaded leg moving in the air. When leg is in support phase, there are three sources of errors:

- friction could change, even for the rotational joint, because there is a bending torque exerted on the vertical axis of the rotational joint;
- pantograph is not purely rigid, and flexibility has not been modeled;
- some of the ball bearings supporting the rotation axis are quite worn and have backlash. This point probably explains the noisy current curve for leg 0, while it remains relatively flat for leg 2.

The voltages are quite similar for the three models, and are under-estimated for reasons exposed in Section 6.4.

Figures 6.6(a) and 6.6(b) show the instantaneous power consumed by the rotational actuator of legs 0 and 2. The swing phase is better predicted by the model than the support phase.

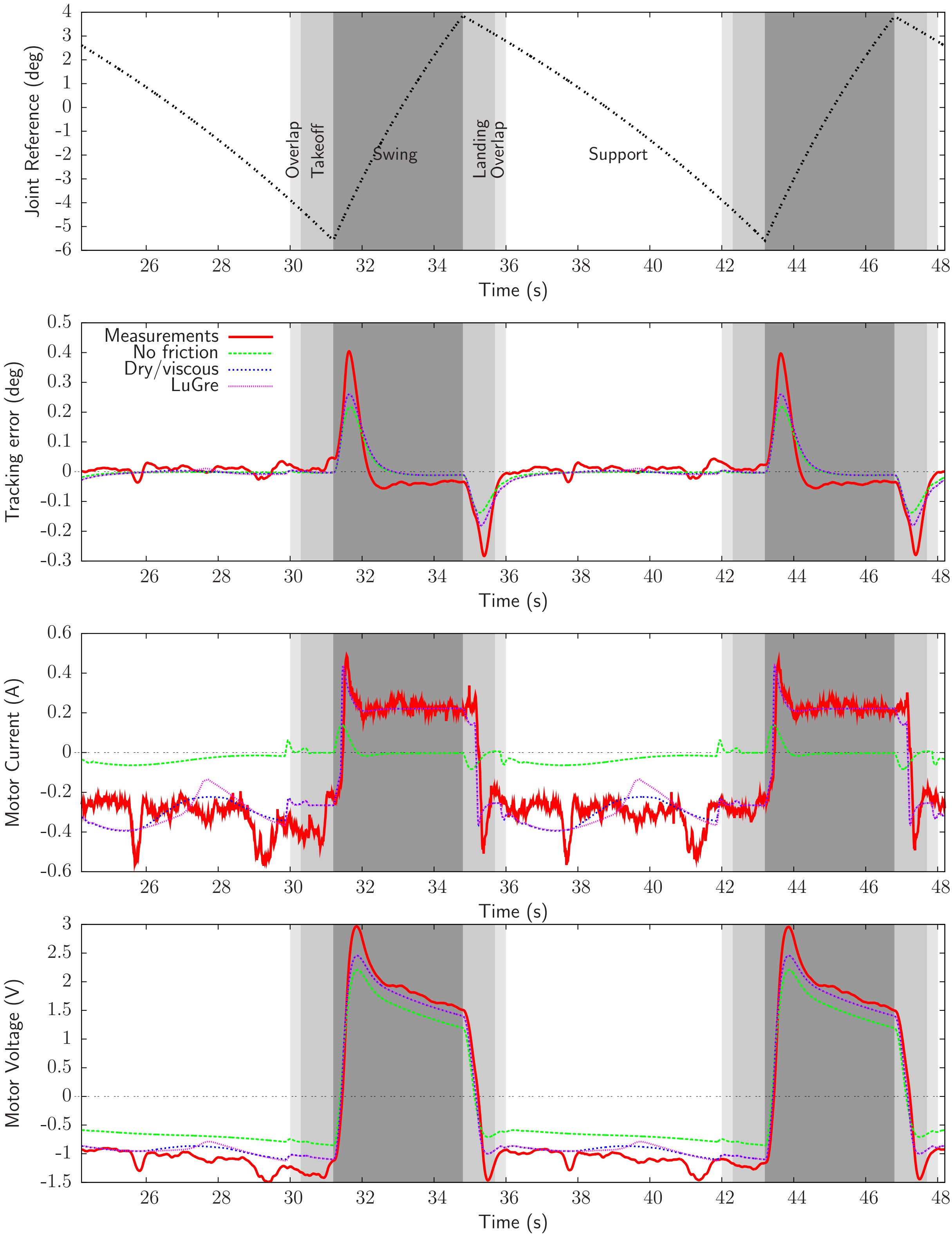


Figure 6.4: Results for leg 0 (rotational joint)

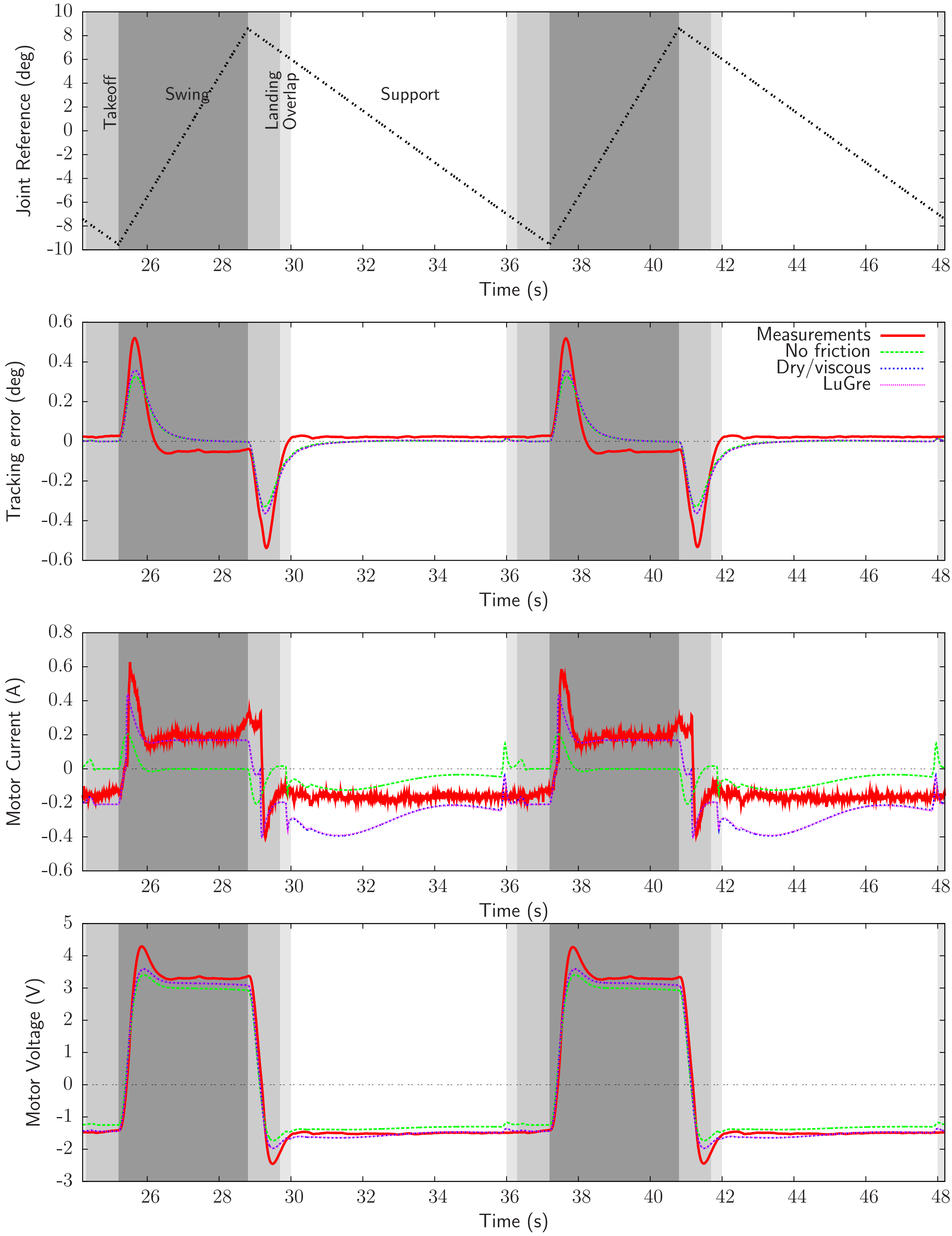
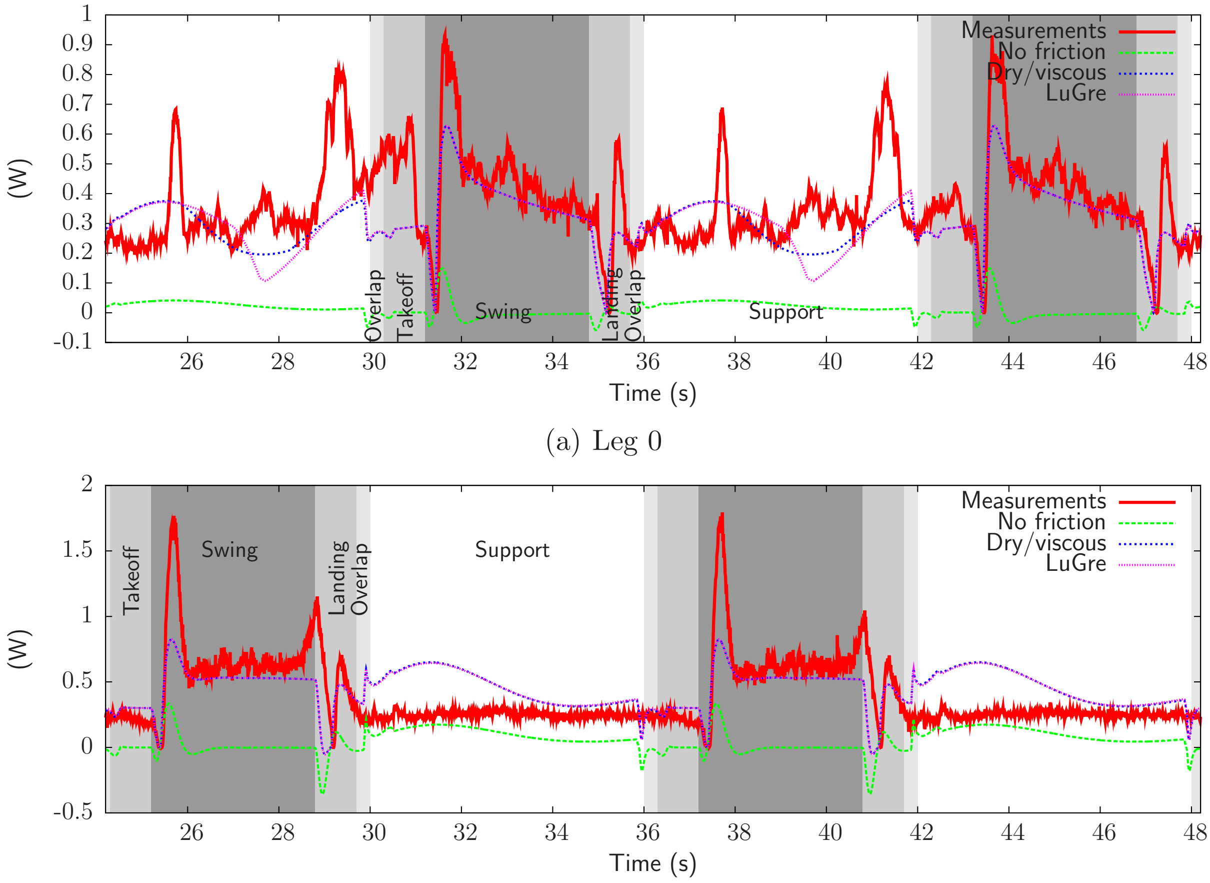


Figure 6.5: Results for leg 2 (rotational joint)



(a) Leg 0

(b) Leg 2

Figure 6.6: Instantaneous power consumption for the rotational actuator

Table 6.3 summarizes the averaged power for the rotational actuator of each leg. From this, friction appears unavoidable in the modeling to obtain realistic powers. As mentioned before, the LuGre model does not give improvement with respect to the experimental data, because rotational joints are always moving in this test case. The relative error between experiments and dry/viscous model is given in the last column.

	Averaged Power (W)				Relative error
Leg number	No frict.	Dry/viscous	LuGre	Exp	between Exp and DV (%)
0	0.01	0.33	0.33	0.41	20
1	0.01	0.23	0.23	0.20	-15
2	0.05	0.42	0.42	0.38	-11
3	0.05	1.14	1.14	1.09	-5
4	0.01	0.22	0.22	0.23	4
5	0.01	0.34	0.34	0.36	6

Table 6.3: Averaged power for the rotational actuators (V_x=1.5 cm/s)

6.4.2 Vertical joints

Figures 6.7 and 6.8 compare experiments and simulations results for legs 0 and 2. As for the rotation, the tracking errors are quite similar. Maximum error is about 0.8 mm while the joint total stroke is between 40 and 90 mm. The swing phases are quite correctly modeled, as stated by the current level.

On the contrary, the support phase involves a null velocity. Without friction, the current level is instantaneously raised to compensate the ground reaction force. With the dry/viscous model, current remains at a low level at the beginning of the support (t between 24-25/36-37 s). This is due to the velocity of the joint, which is not zero yet because of the controller dynamics. This velocity reaches 1e-4 m/s (limit velocity of dry/viscous model) at t ≈25 s. At this moment, friction torque is decreasing with the velocity according to Eq. (6.1): hence the motor has to give more power to counteract the effect of the ground reaction force. With the LuGre model, the joint decreasing velocity induces an increase of the friction torque according to Eq. 6.4, thus a less important current consumption. It is worth noting that the vertical irreversibility of leg 0 is correctly reproduced in Fig. 6.7 during the support phases (t=26-28/38-40 s). After this time, friction is no longer high enough to counteract the ground reaction force, and the joint becomes reversible.

The swing phase of leg 2 is less correctly predicted by the dry/viscous or LuGre models. The variation of robot height due to leg flexibilities has been recognized as an important problem in walking machines [JIME96]. These variations make that the foot is already at ground level while it has not finished its swing phase yet. Hence, the foot touches the ground whereas it should still be in the air. There is consequently an important current consumption during the landing in Fig. 6.8, which is not included in our model. Also, the evolution of the current during the support is quite different experimentally than with the LuGre or the dry/viscous model. At the beginning of the support (t=30-32 s), the dry viscous model seems to fit perfectly the measured current. Thus, there is a lost of irreversibility, which finds several explanations:

- the friction parameters can change with the foot load, which has not been modeled for the vertical actuators;
- a "jerky" motion of the body exists [GONZ06] when the leg is landing on the ground, even if precautions have been token to have a landing of the feet as smooth as possible; this jerky motion could "unlock" the vertical joint.

When support is established (t≥32 s), the real current is more important than with the models, because of the transmissions deformation. This defect has been identified and modeled for the horizontal joints (see Section 6.4.4), but not for the vertical ones.

Finally, the shapes of applied voltages are correctly predicted, excepted for leg 2 at the end of the landing, and during the support phase because of the loss of irreversibility.

Figures 6.9(a) and 6.9(b) show the instantaneous power consumed by the vertical actuator of legs 0 and 2. Even without friction, the power of the support phase is quasi-null: consequently, the LuGre model could be avoided for energy purposes (this will be the case in the next section). Differences come from the meshing in the transmission, from its defects (more or less important according to the considered leg), and from the early touching of the ground during the landing phase, while the foot should still be in the air.

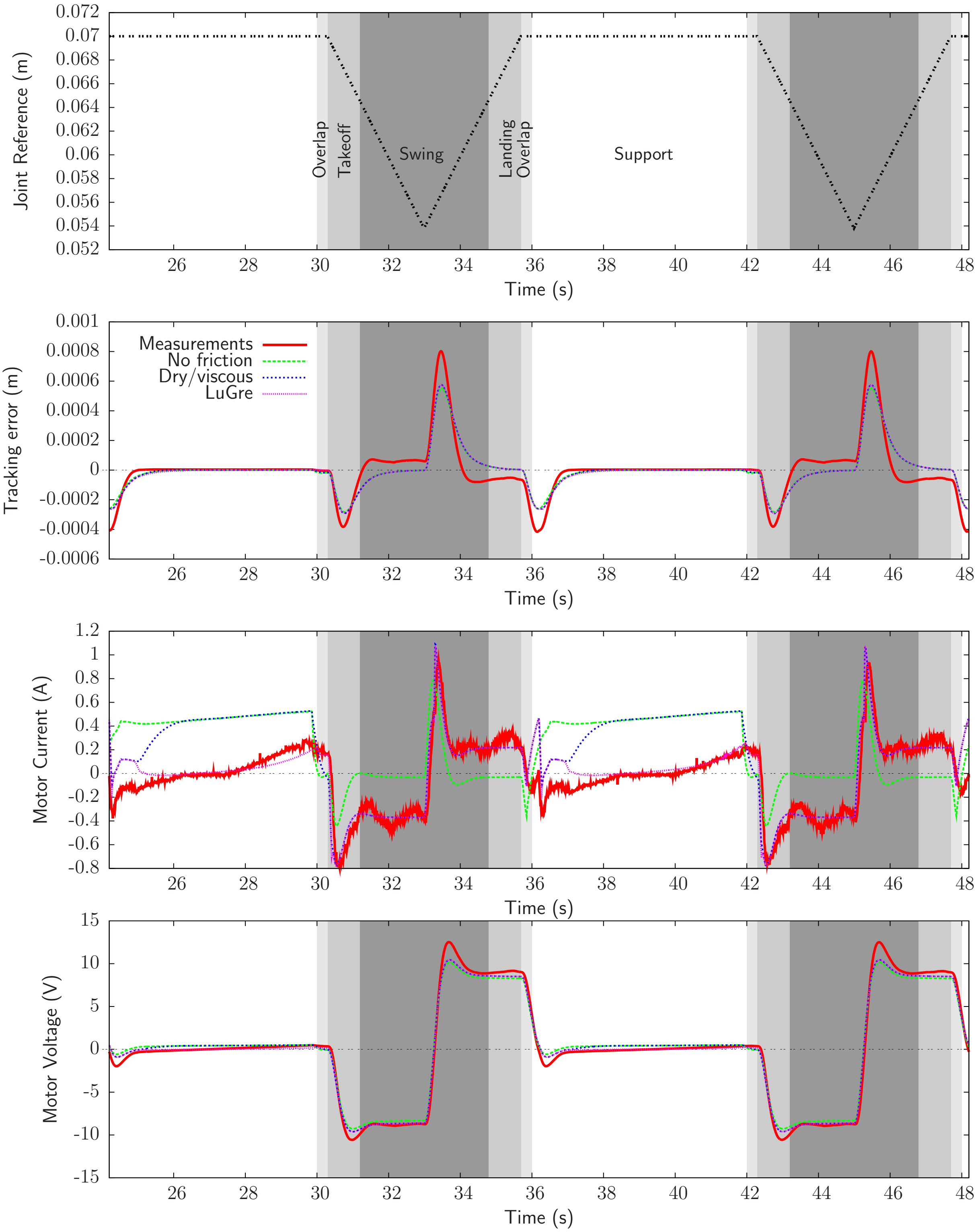


Figure 6.7: Results for leg 0 (vertical joint)

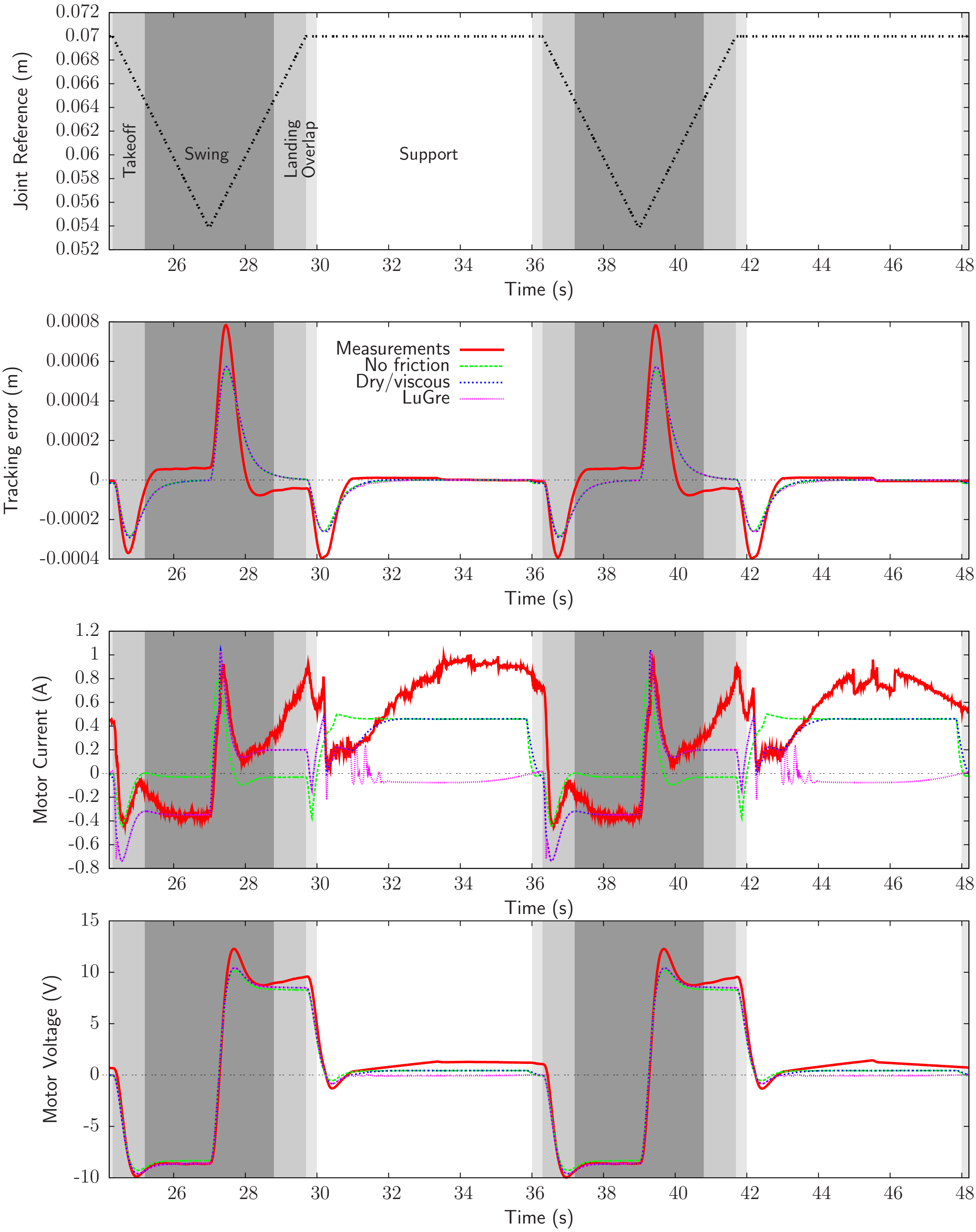
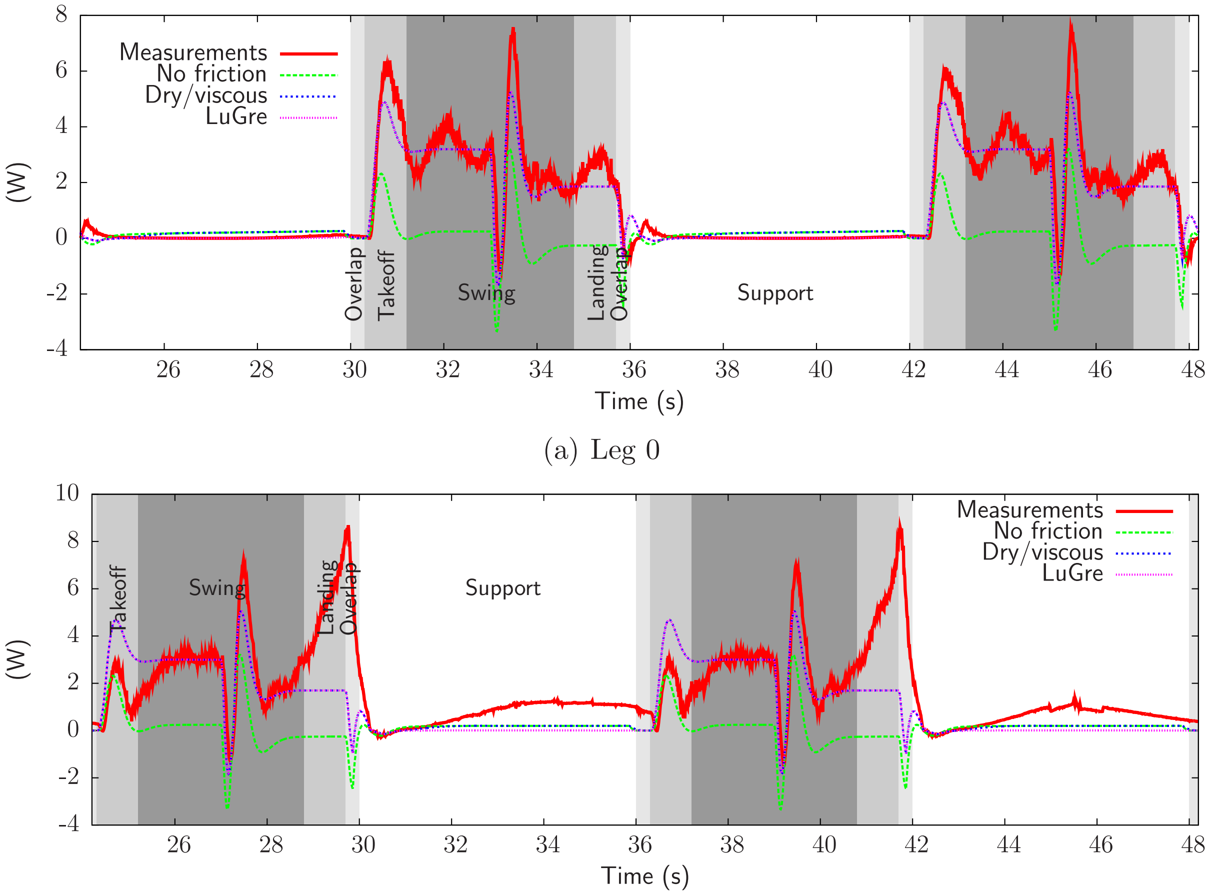


Figure 6.8: Results for leg 2 (vertical joint)



(a) Leg 0

(b) Leg 2

Figure 6.9: Instantaneous power consumption for the vertical actuator

Table 6.4 summarizes the averaged power for the vertical actuator of each leg. Friction is again unavoidable in the modeling to obtain realistic powers. Prima facie, the high friction of the vertical transmission could be beneficial in terms of energy expenditure, because no power is needed to keep the main robot body at its altitude. But it is not the case: in fact, even without irreversibility, power consumption during the support phase is very small, mainly thanks to the high gearing of the transmission. The relative error between experiments and dry/viscous model is given in the last column. The fact that the foot is already on the ground for its landing phase in the real-world is a major source of modeling imperfection.

	Averaged Power (W)				Relative error
Leg number	No frict.	Dry/viscous	LuGre	Exp	between Exp and DV (%)
0	0.12	1.24	1.20	1.43	13
1	0.11	1.14	1.09	1.51	25
2	0.11	1.14	1.07	1.69	33
3	0.11	1.31	1.30	1.83	28
4	0.10	1.14	1.11	1.50	24
5	0.10	1.22	1.16	1.59	23

Table 6.4: Averaged power for the vertical actuators (V_x=1.5 cm/s)

6.4.3 Horizontal joints

Figures 6.10 and 6.11 compare experiments and simulations for legs 0 and 2. Maximum tracking error is about 0.5 mm while the joint total stroke is between 5 and 55 mm.

Current and voltage of leg 0 swing phase in Fig. 6.10 are well reproduced, indifferently for the LuGre or the dry/viscous models. On the other side, the support phase is badly predicted. This is due to constant friction parameters for the horizontal joints, while they vary with the leg load (see Section 5.4).

The estimation of currents and voltages are not so good in Fig. 6.11. The horizontal joint velocity of leg 2 is very low, and the modeling of friction bearing at low velocity seemed to include a Stribeck's effect (a rising friction force with decreasing velocity) which has not been modeled. Moreover, as for leg 0, the load influence has not been inserted in the model.

The only noticeable difference between the dry/viscous and the LuGre model is that the viscous model makes the friction force vanish more quickly than with LuGre (see currents in Fig. 6.11 at $t = 32 - 34$ s). Nevertheless, the measured current undergos the zero crossing later, probably because of the modeling errors in the electrical circuit, which are growing at low speed.

Figures 6.12(a) and 6.12(b) show the instantaneous power for the horizontal actuators of legs 0 and 2. Is is clear that power in support phase is underestimated by the model, because friction on the horizontal joint does not adapt with the load. The shape of the power is also very oscillating, because of the friction bearings which have a less smooth behavior than linear ball bearings.

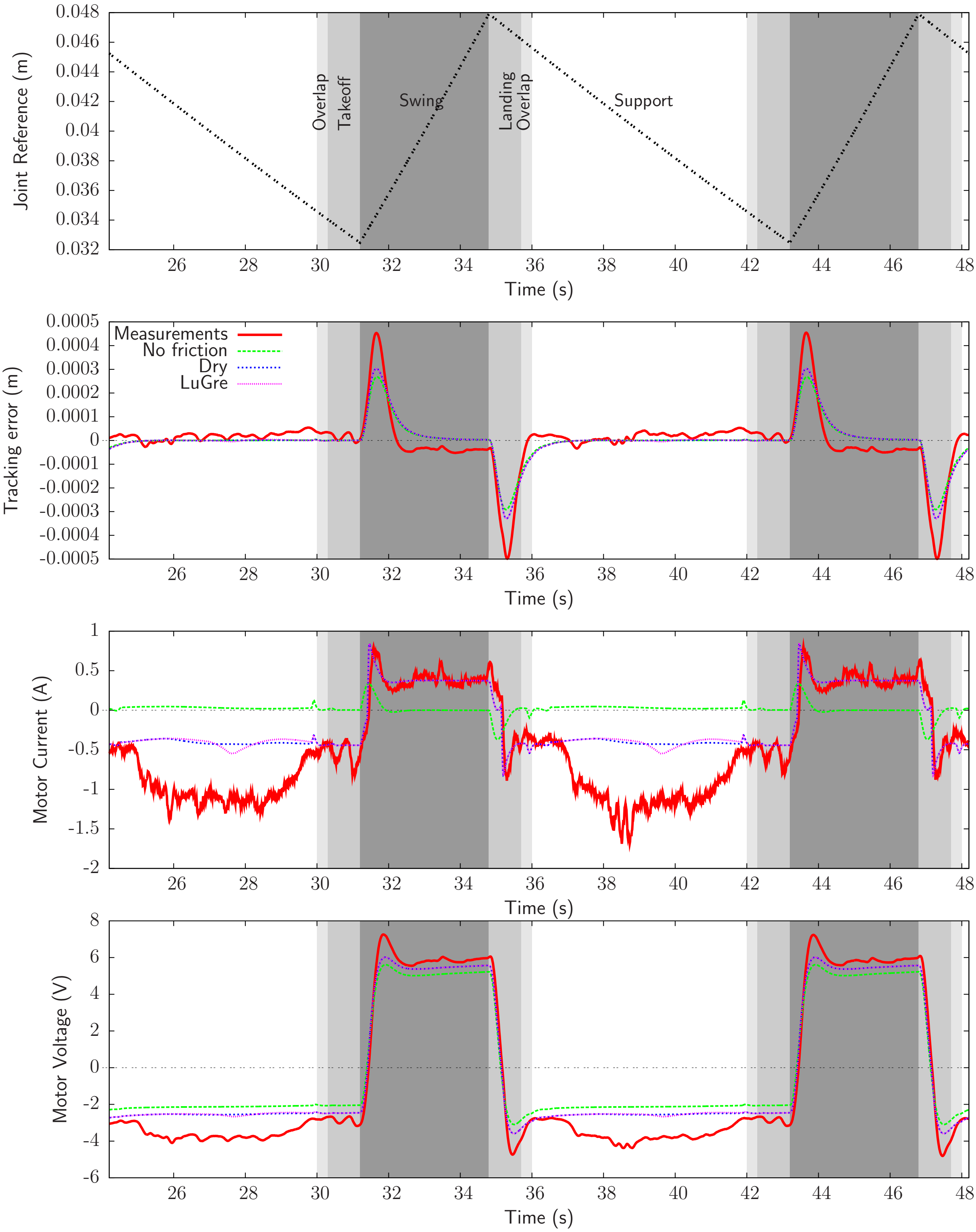


Figure 6.10: Results for leg 0 (horizontal joint)

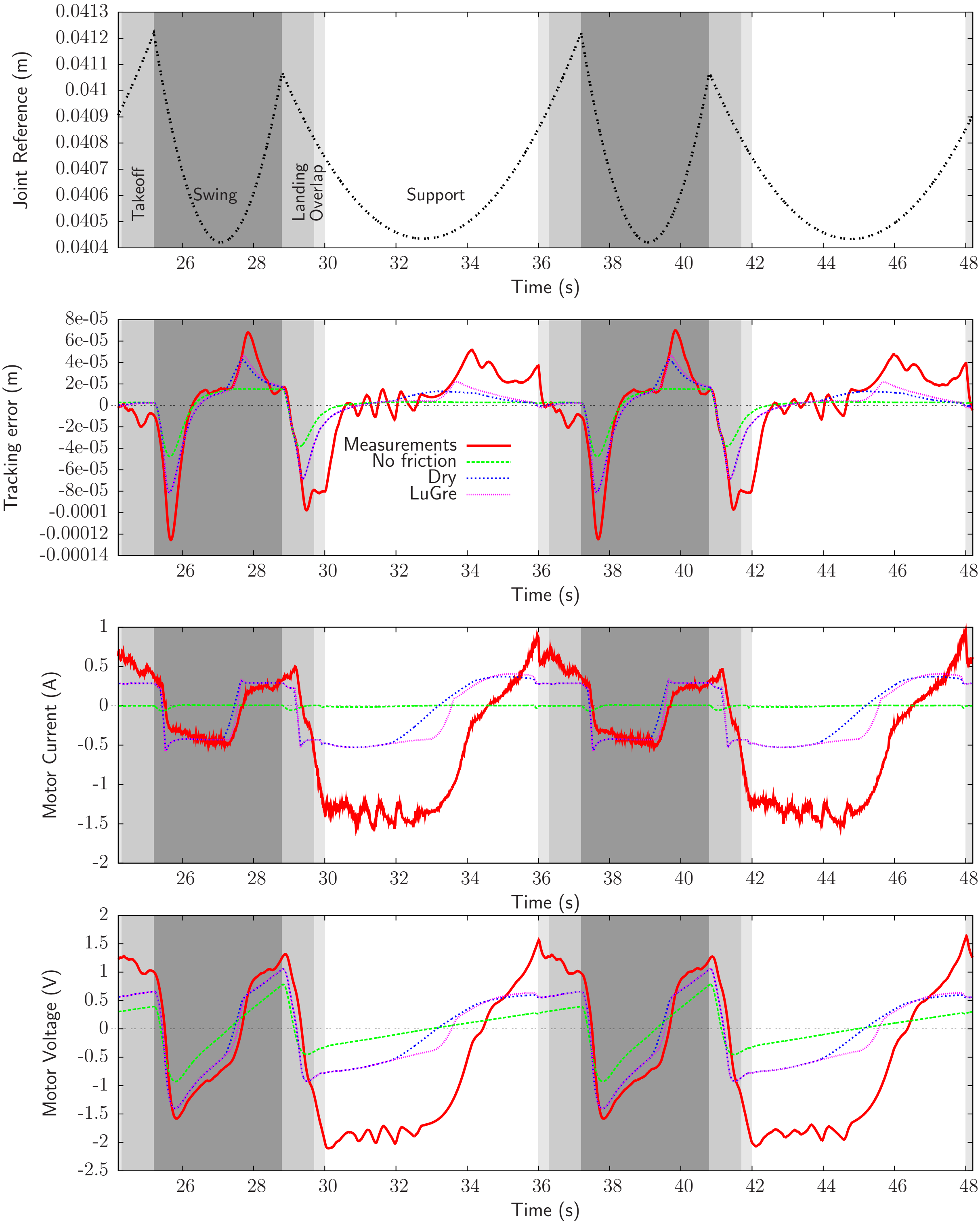
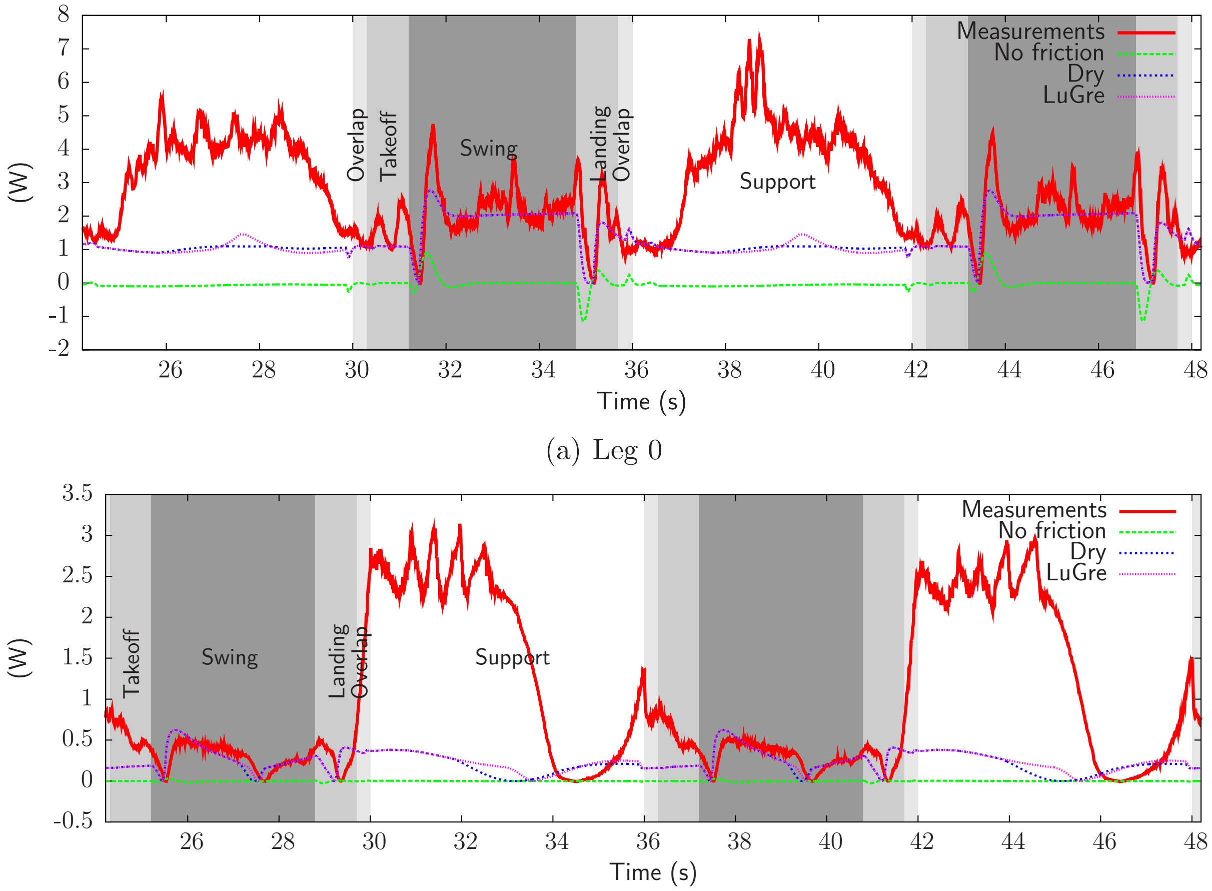


Figure 6.11: Results for leg 2 (horizontal joint)



(a) Leg 0

(b) Leg 2

Figure 6.12: Instantaneous power consumption for the horizontal actuator

The estimation of the powers per leg for the horizontal actuators are gathered in Table 6.5, for the three friction models.

	Averaged Power (W)				Relative error
Leg number	No frict.	Dry/viscous	LuGre	Exp	between Exp and DV (%)
0	-0.02	1.43	1.43	2.93	51
1	-0.02	0.97	0.97	2.42	60
2	0.003	0.28	0.30	1.01	72
3	0.004	0.30	0.31	0.93	68
4	-0.03	0.76	0.77	1.50	49
5	-0.04	0.77	0.78	2.77	72

Table 6.5: Averaged power for the horizontal actuators (V_x=1.5 cm/s)

6.4.4 Horizontal joint with variation of friction parameters

It is clear from previous simulations on the horizontal joints that the effect of load must be taken into account to have power consumptions closer to the measurements.

Figures 6.13 and 6.14 present the evolutions of currents and voltages in the horizontal joints for legs 0 and 2 respectively. The dry friction torques have been computed from Eq. (5.12) with coefficients of Table 5.5. The zero order term is replaced by the dry torque of Table 5.3.

Experiments are compared on one hand to the dry model without the load influence, and on the other hand to the dry model which includes the dry friction/load dependency. In the case of leg 0, it is clear that this contribution improves the current estimation. For leg 2, the shape of the current is lightly enhanced. Again, velocity of horizontal joint for leg 2 is quite low, where the modeling of friction is difficult.

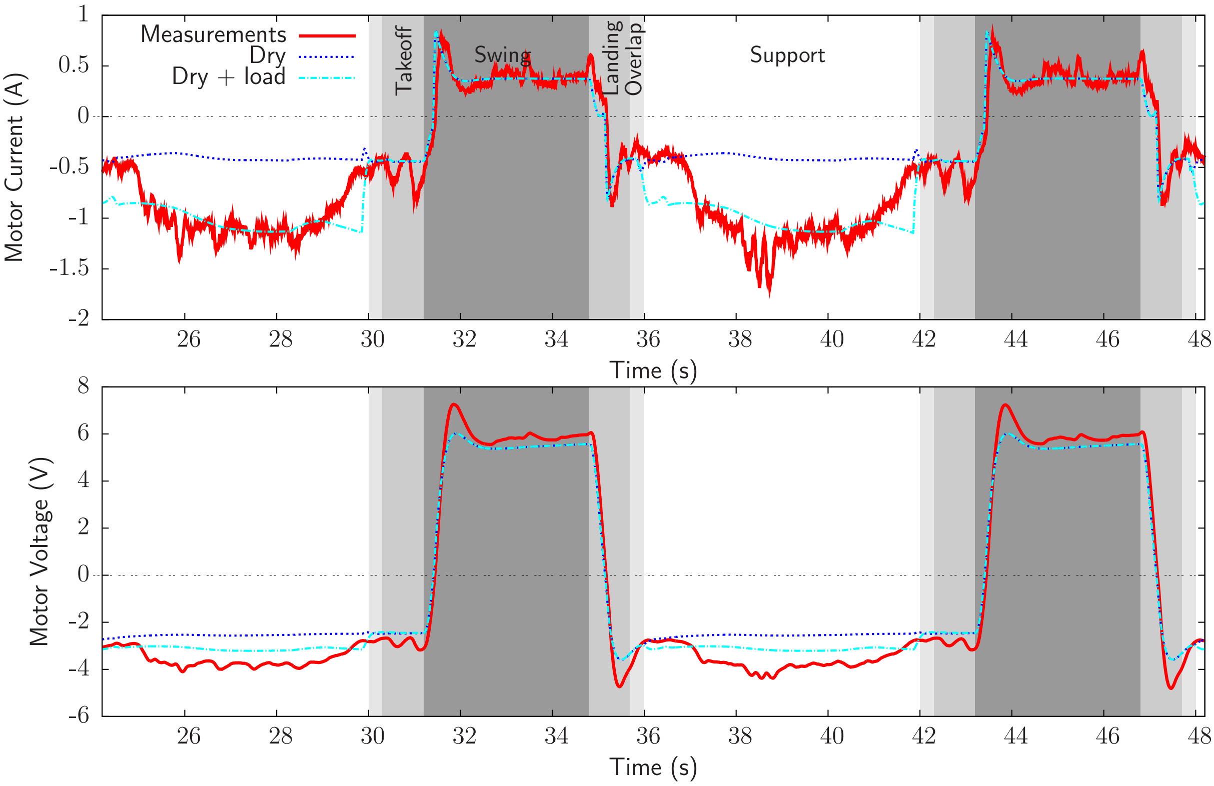


Figure 6.13: Results for leg 0, horizontal joint (with load influence)

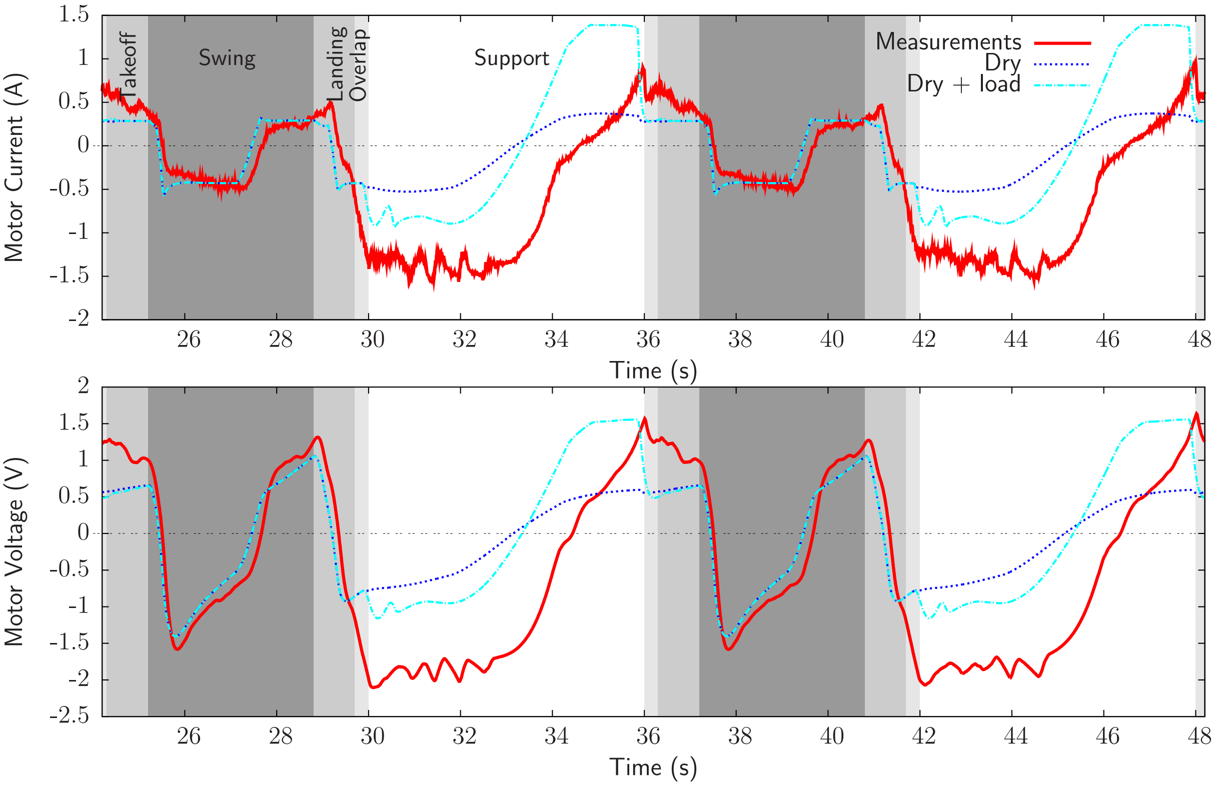
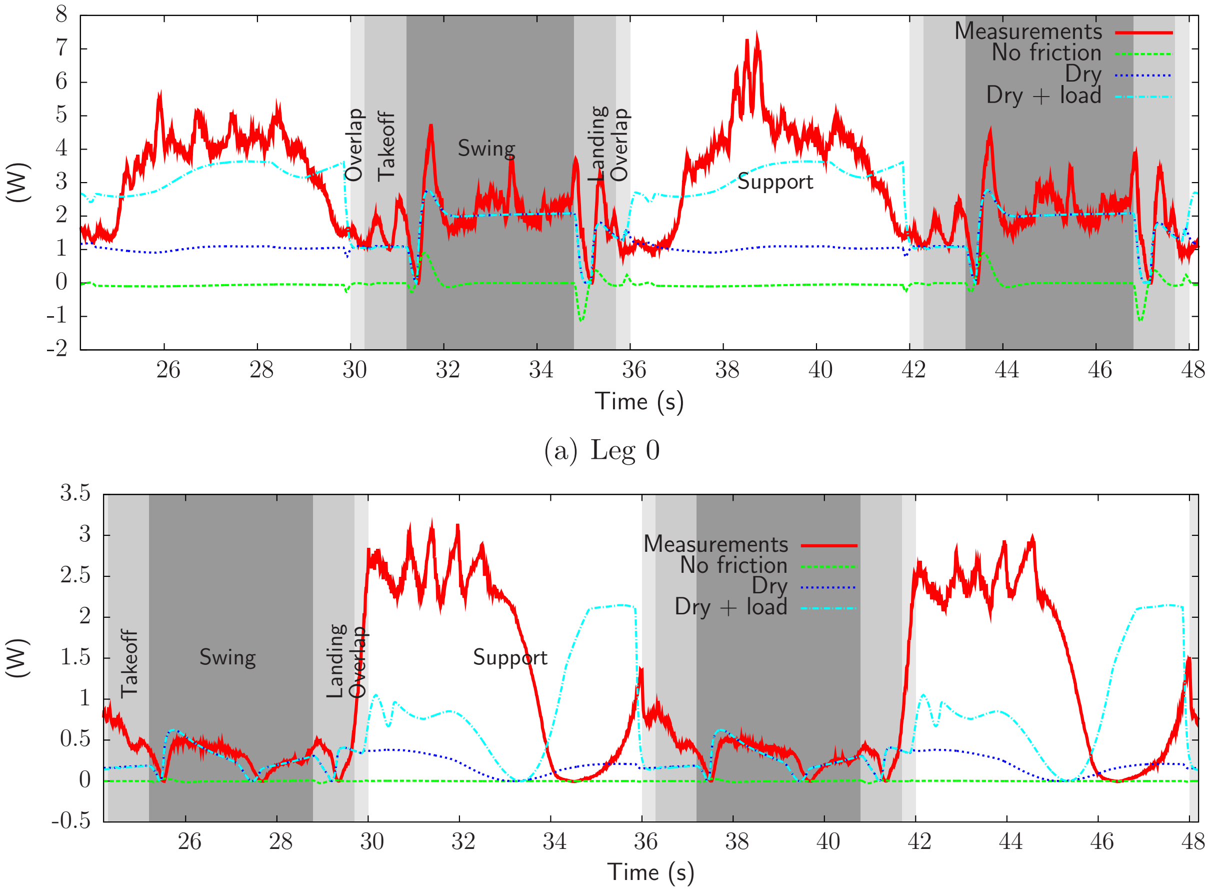


Figure 6.14: Results for leg 2, horizontal joint (with load influence)



(a) Leg 0

(b) Leg 2

Figure 6.15: Instantaneous power consumption for the horizontal actuator - load influence

Figures 6.15(a) and 6.15(b) compare the power consumption without friction, with a "simple" dry model and with a dry model taking into account the load influence. With load influence, powers are better estimated.

Table 6.6 shows the averaged power of the horizontal actuators in the three cases. The error is given between the dry+load effect model and the experiments, and should be compared to the errors of Table 6.5. The model gives bad estimation of power for legs 4 and 5, as illustrated in Figs. 6.16 and 6.17. Explanation is that the deformation of transmission during walking is certainly different from the one during the identification of Section 5.4; a more accurate model of friction with deformation is out of the scope of this work.

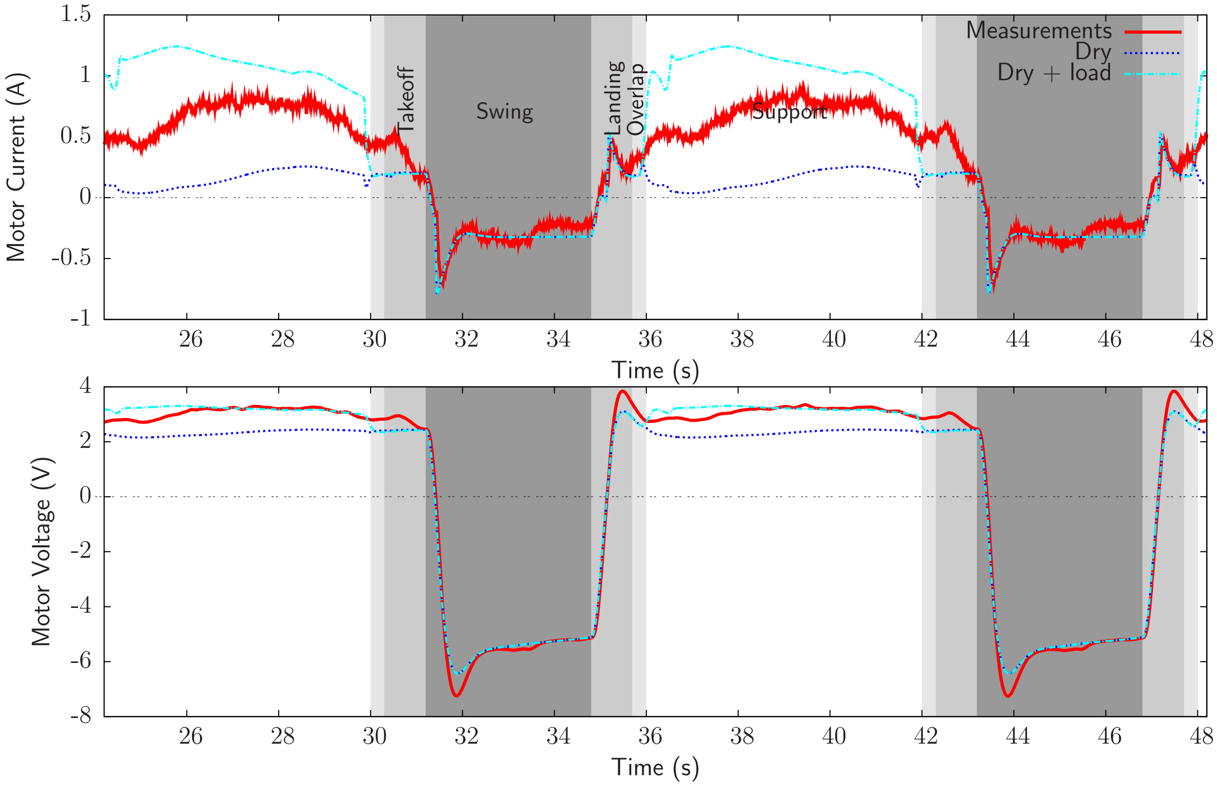


Figure 6.16: Bad estimation of friction/load influence for leg 4

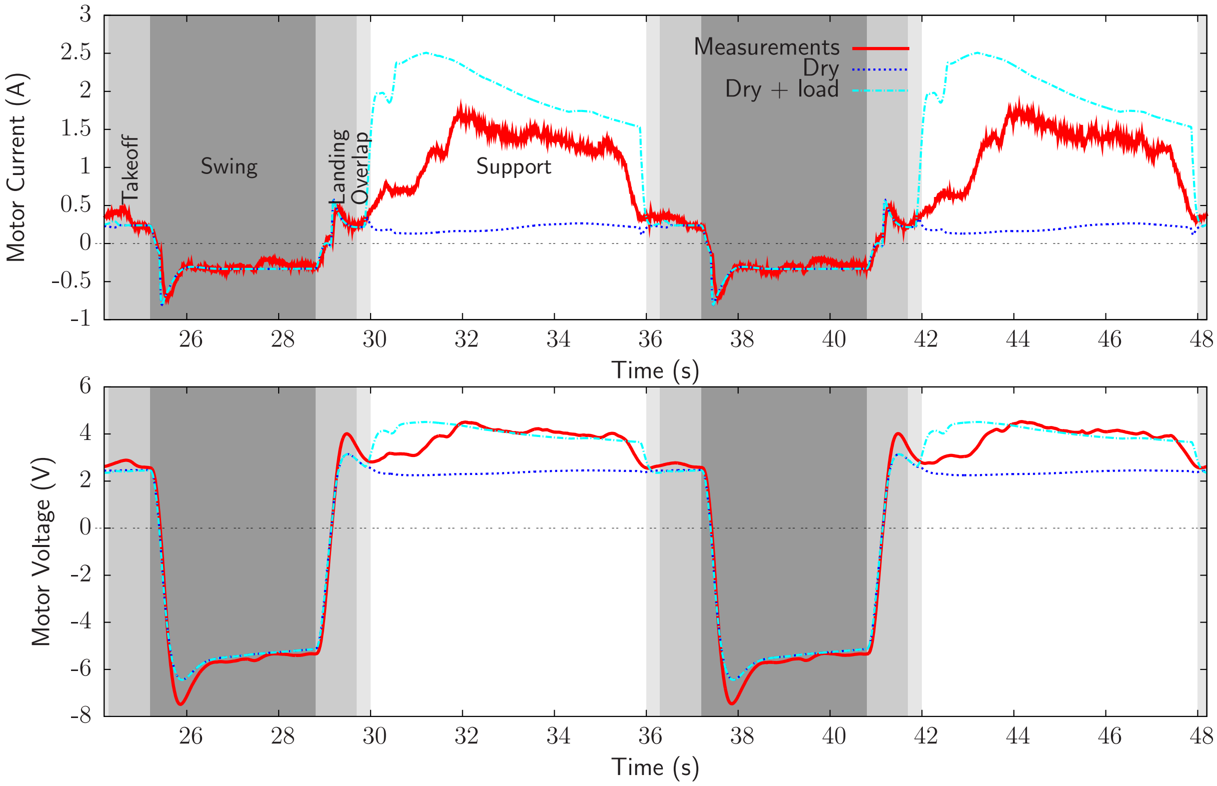


Figure 6.17: Bad estimation of friction/load influence for leg 5

	Averaged Power (W)			Relative error
Leg number	Dry/viscous	Dry/viscous + load	Exp	(%)
0	1.43	2.51	2.93	14
1	0.97	1.83	2.42	24
2	0.28	0.66	1.01	35
3	0.30	0.70	0.93	25
4	0.76	2.17	1.50	-45
5	0.77	4.59	2.77	-66

Table 6.6: Improvement of power consumption for the horizontal joints (V_x=1.5 cm/s)

6.4.5 Specific Resistance

The specific resistance is an indicator of the performance of a vehicle. It is defined by:

$$\xi = \frac{E}{mgl} \tag{6.6}$$

where E is the total energy expended for a vehicle of mass m traveling a distance l. Energy for actuator i is given by:

$$E_i = \sum_k u_k^i \cdot i_k^i \cdot \Delta t \tag{6.7}$$

where Δt represents the sampling time and k the samples.

The Gabrielli-Von Karman diagram allows a comparison of different locomotion modes: for information, it is shown in Fig. 6.18, where AMRU5 has been located thanks to the measurement and computation of the specific resistance presented hereafter.

Figure 6.19(a) represents the averaged power of the robot with respect to the velocity. The linear shape of this kind of curve has already been mentioned in [MARH97]. Friction is responsible for the bigger part of the power expenditure. The LuGre model gives similar energy consumption as dry/viscous model (without load) because it has been seen previously that power requirement of the vertical actuators during the support phase are very low. The influence of the load on friction coefficient in horizontal actuators can surely not be neglected. The effect of the change in friction changes slightly the slope of the curve, but gives a better approximation of the averaged power. Effectively, if V_x is increasing, the foot profile is larger, resulting in an higher deformation of the leg mechanism for all the legs. Averaged power at low velocity is proportionally higher at low velocity for experimental measurements: this is due to an increase of friction (Stribeck's zone).

Figure 6.19(b) illustrates the evolution of the specific resistance with respect to different velocities of the vehicle. This specific resistance decreases with the velocity: indeed, if we express the power P as a first order polynomial [MARH97] depending on V_x:

$$P = \alpha + \beta V_x \tag{6.8}$$

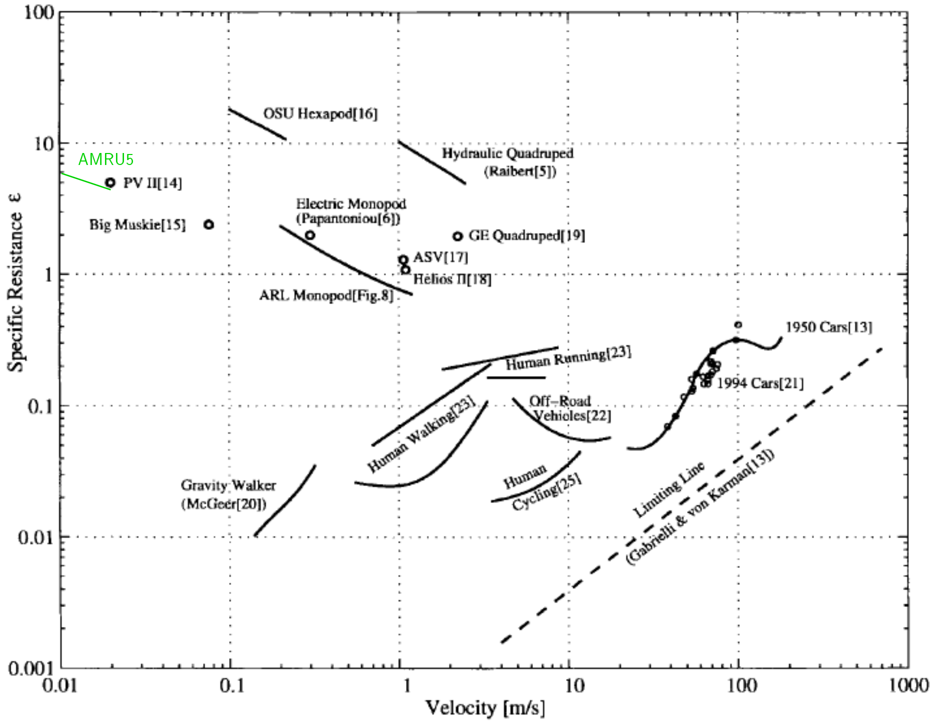


Figure 6.18: Gabrielli-Von Karman diagram (adapted from [GREG97])

where α could be interpreted as pure losses in the vehicle and βV_x as the useful power required to propel the robot. We can write the specific resistance as:

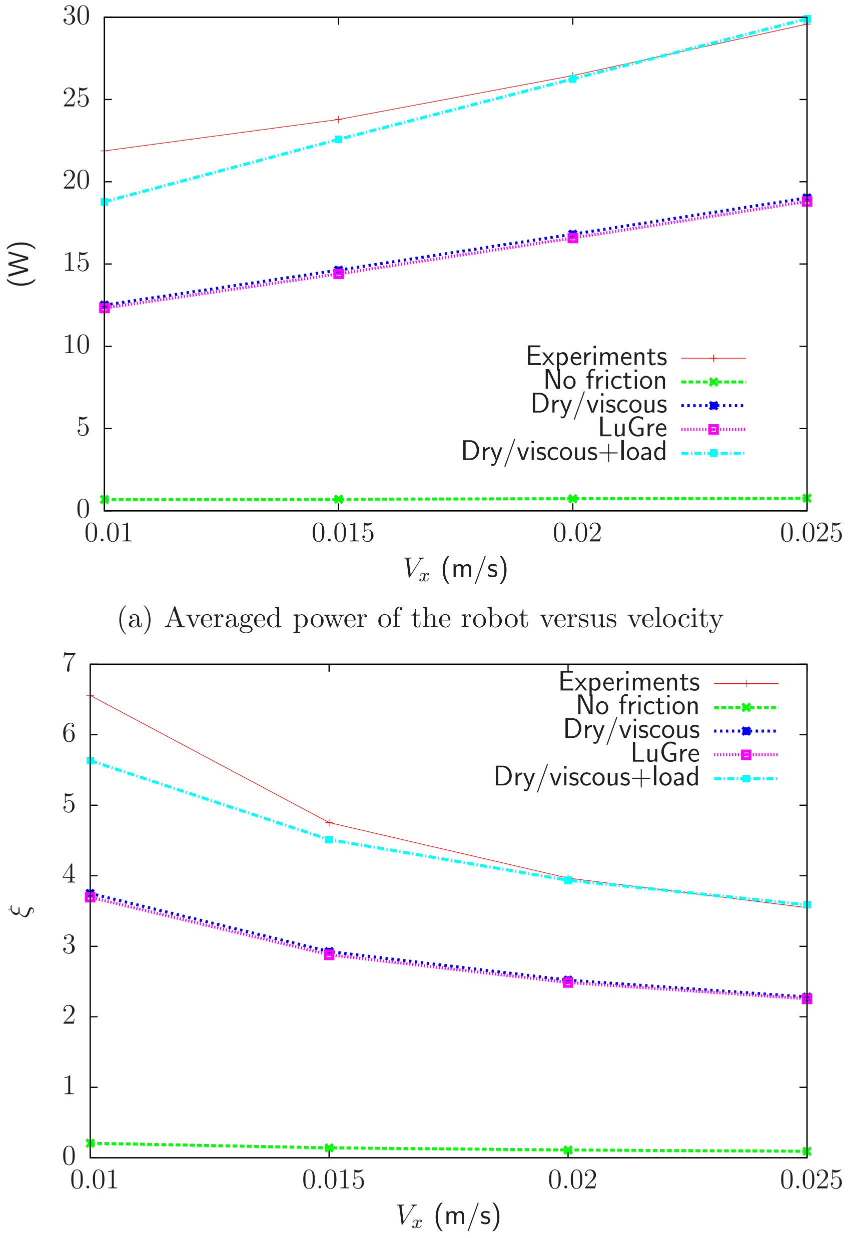
$$\xi = \frac{P \cdot T}{mgl} \tag{6.9}$$

$$= \frac{P}{mgV_x} \tag{6.10}$$

$$= \frac{\alpha}{mgV_x} + \frac{\beta}{mg} \tag{6.11}$$

where T and l are the travel time and distance respectively. Relation 6.11 shows that for high velocities, the power losses of the robot (α contribution) become small with respect to the second term $\frac{\beta}{mg}$. Specific resistance asymptotically tends to the latter value.

For the lowest velocity, the relative error of specific resistance between the model with dry/viscous (plus load effect) and the experiments is of 12 %. For higher velocity, the simulation gives a good prediction of the specific resistance.



(a) Averaged power of the robot versus velocity

(b) Specific resistance versus velocity

Figure 6.19: Energetic indicator for different friction models

6.5 Extensive simulations with averaged friction

Friction proved to be an important cause of the energy expenditure, but is also full of uncertainties. To validate the reliability of the model and thus the estimation of the specific energy, it would be interesting to perform an important number of simulations, in which friction parameters would be generated randomly according to a Gaussian distribution. This way of proceeding is called a Monte-Carlo process.

In this section, we consider only a dry/viscous model on the rotation and the vertical joints, and a dry model with/without the load influence on the horizontal joint. The extra energy consumption due to the irreversibility of the vertical joint is not modeled by LuGre, because power consumption is similar whatever the model for the vertical actuator (see Table 6.4). We also consider that friction parameters are identical for each leg, and are determined with the following deterministic approach:

- One average value μ^* for each parameter of Table 5.3 is computed by taking the mean value of the six legs, but with some terms are omitted in the averaging:
  - the viscous effect of the horizontal joint is neglected;
  - the friction parameters for the rotational joint of leg 3 are omitted (see Fig. 5.3), because they are far from other values;
  - in the same way negative viscous coefficient for the rotation are neglected because not physical;
  - the term a_2^+ of Table 5.5 for leg 5 is neglected, as well as a_2^- for leg 1, because they are too important with respect to the other values.
- The standard deviation σ^* is computed, based on the totality of the measurements performed in Section 5.2 (excepted the ones excluded in the previous point).
- The same process is performed on polynomial coefficients; mean and standard deviations are in Table 5.5.
- Parameters are then generated randomly, by assuming a Gaussian distribution whose probability density function $f(x)$ is:

  $$f(x) = \frac{1}{\sqrt{2\pi\sigma^{*2}}}\, e^{-\frac{(x-\mu^*)^2}{2\sigma^{*2}}}; \qquad (6.12)$$

  For each simulation, random values are generated, and are omitted if they are either non-physical, or outside of the ±3σ interval, which corresponds to a confidence interval of 99,7 %.

Table 6.7 shows the mean μ of each friction parameter and its standard deviation σ. Dry friction torques are in N for translational joints and in Nm for the rotation; viscous coefficient are in Ns/m for translational joints and in Nms for rotation.

Four thousands simulations have been performed, one thousand for each velocity envisaged in the graph of Fig. 6.19(b). Figure 6.20 shows the means and 3σ intervals

	Rotation		Vertical		Horizontal	
Friction parameter	μ	σ	μ	σ	μ	σ
$\tau_{f,d}^{+}$ (Nm or N)	8.04	3.58	311.4	20.8	328.5	83.9
f_v^{+} (Nms or Ns/m)	3.46	2.55	7949.3	3843.6	-	-
$\tau_{f,d}^{-}$ (Nm or N)	-8.97	3..56	-382.0	27.4	-452.9	76.0
f_v^{-} (Nms or Ns/m)	16.17	7.57	13607.0	5593.6	-	-

Table 6.7: Mean and standard deviation of the friction parameters used in the Monte Carlo method

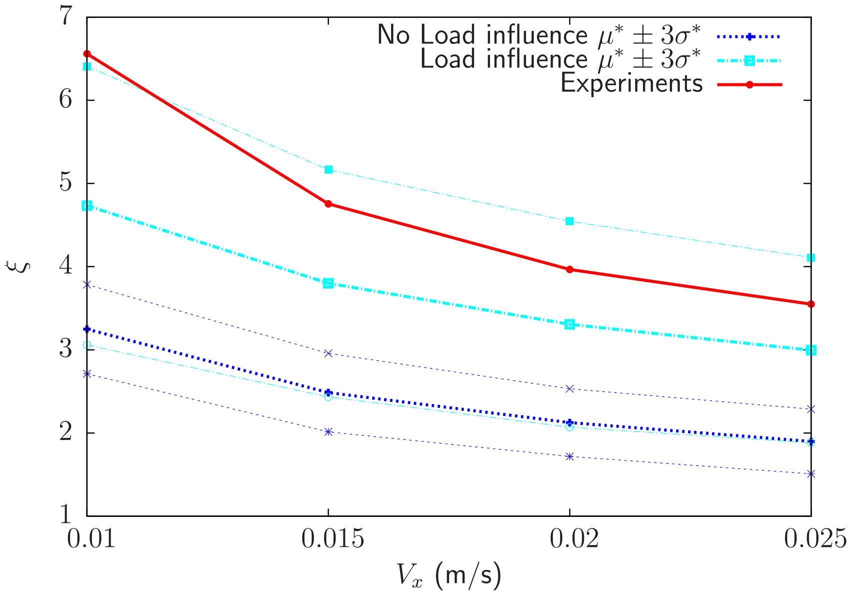


Figure 6.20: Specific resistance for different friction models

obtained after the Monte-Carlo process. Mean values and standard errors are summarized in Table 6.8 The load influence on the horizontal joint clearly penalizes the robot specific resistance. Also, the system is quite sensitive to this parameter, as shown by the important ±$3\sigma^*$ interval. The curve without load influence could be interpreted as the specific resistance of the robot is this mechanical defects were corrected. The efficiency of the robot would be improved by a factor 2.

V_x (m/s)	ξ (no load infl.)		ξ (load infl.)	
	μ^*	σ^*	μ^*	σ^*
0.010	3.25	0.18	4.73	0.59
0.015	2.49	0.16	3.80	0.46
0.020	2.13	0.14	3.31	0.41
0.025	1.90	0.13	3.00	0.37

Table 6.8: Mean and standard deviation for specific energy

6.6 Summary

The main purpose of this chapter was the validation of the dynamic model. In a first step, it has been shown that friction could not be neglected at all, because in the case of AMRU5, it is responsible for more than 80% of the power consumption. The contribution of the LuGre model is interesting for accurate simulation of irreversibility, which is often ignored in other works because the friction model is simply dry/viscous, with a linear approximation at zero velocity. However, as far as power expenditure is concerned, the LuGre or the dry/viscous model can be used indifferently.

For the particular case of AMRU5, it has been necessary to take into account the load effect on the horizontal joint friction. The load is responsible for the transmission deformation, increasing friction. Neglecting this effect leads to large under-estimation of the power consumption.

Finally, we performed a Monte Carlo process in which the uncertainties about friction parameters have been made under a Gaussian distribution law. For the dry/viscous model without load dependency, the error interval ($\pm 3\,\sigma$) on the specific resistance is approximately $\pm 20\%$. If load effect is considered, the $\pm 3\,\sigma$ interval rises to $\pm 35\%$. Again the deformation of the horizontal joint proved to be an important source of uncertainty, that should be considered for realistic estimation of the robot power expenditure.

CHAPTER 7

Gait control

Gait control consists in moving the robot body along a given trajectory while keeping the vehicle stability. Several methods exist in literature and can be classified in two categories :

- the *engineering* or descending approach where **the legs motion is a consequence of the gait generation**, which depends on a central decision organ;
- the *bio-inspired* or ascending approach by which **the gait generation is an emerging behavior of the legs motion**. Legs have local rules to determine their motion. There is just a central organ for coordination. This approach comes from stick insects and cockroaches observations.

The engineering method has been developed in this work because AMRU5 is slow and is not a biological inspired robot, i.e. it does not have an insect-like structure. Emphasis has been put on a reliable algorithm to control the robot precisely on an unstructured ground.

This chapter begins with Section 7.1 where some basic notions used in gait control of legged robots are exposed. Thereafter, a state of the art describes in Section 7.2 the numerous existing methods and successfully implemented on legged vehicles. After this review, the algorithms developed within the framework of the thesis are presented. First, kinematic control of a leg is detailed in Section 7.3. The purpose of this algorithm is to find the joint variables which produce the desired position of the foot in the leg local frame. Developments are made such that the leg local frame and the foot are equivalent to the base and the end-effector of a robotic manipulator respectively. Input of this kinematic control comes from the coordination level, which determines if a leg is in support or in swing phase. Section 7.4 shows how to move the body smoothly from legs in support phase.

Swing phase in the case of a flat ground is explained in Section 7.5. Well-known periodic gaits are first presented, with an improvement for support-swing transitions. This

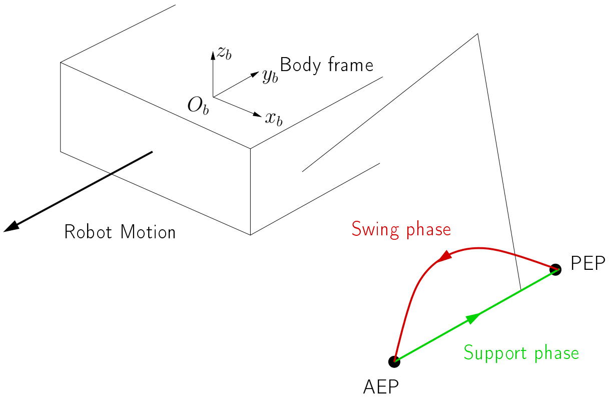


Figure 7.1: Basic leg step cycle (main body point of view)

improvement has been mandatory for AMRU5 because flexibility in leg mechanism can not ensure that the foot was lifted up as soon as the vertical joint was moved. Periodic gait is the algorithm used for the modeling validation of Chapter 6. Secondly, the omnidirectional ability is developed on the robot, by adapting continuously the swing phase with the body velocity.

Finally, free gait is developed in Section 7.7. Support and swing phases are identical to Sections 7.4 and 7.5 respectively. Differences come from the uneven ground which requires the detection of the foot-ground contact. Navigation is investigated and posture adaptation is envisaged to keep the robot parallel to the ground profile. The chapter ends with Section 7.8 where a summary of the results obtained is presented.

The different gait algorithms have been tested with the dynamic simulation, but without friction at joint, because it has been exposed in Section 6 that friction does not influence the tracking efficiency of the joints. On the contrary, the slipping at ground and the dynamics of the controllers are influent.

7.1 Definitions

The following definitions have been widely adopted in literature. Most of them come from [SONG89].

7.1.1 Typical parameters in legged locomotion

A basic leg cycle (or step) is drawn in Fig. 7.1 for an observer linked to the central body frame $O_b x_b y_b z_b$. It is composed of two phases:

- the *swing phase* of duration τ is the period during which the leg is not in contact with the ground (also called *transfer phase* or *protraction* for biologists);

- the *support phase* of duration s is the period during which the leg is in contact with the ground and propels the body (also called *stance phase* or *retroaction* for biologists) .

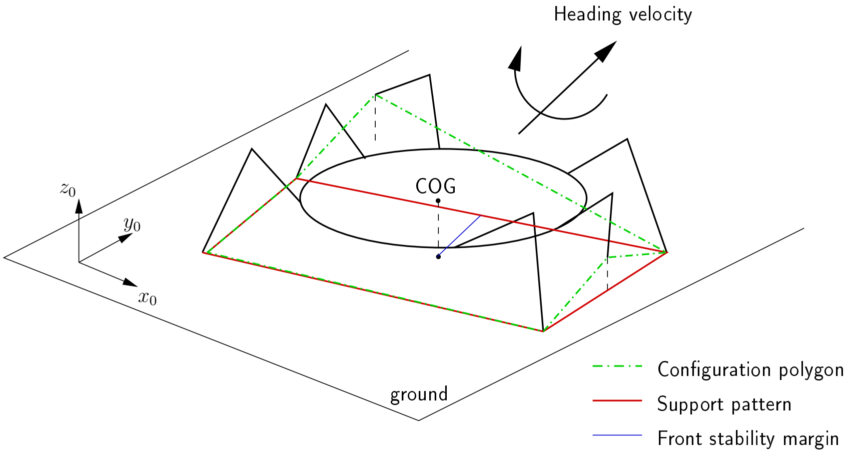


Figure 7.2: Support pattern, configuration polygon and stability margin

The cycle time $T = \tau + s$ is the time for a complete cycle of locomotion, i.e. the complete execution of a transfer and a support phase.

The position reached at the end of the swing phase is called the *Anterior Extreme Position* (AEP) while point reached at the end of the support phase is named the *Posterior Extreme Position* (PEP).

The *duty factor* $\beta = s/T$ of a leg is the time during which the leg is on the ground, brought back to a cycle time.

The *phase shift* $\Delta\phi_{ij}$ is the cycle time fraction between the beginning of the leg i support phase and the one of a reference leg j.

The *stroke* R is the relative distance covered by a leg during the support phase, i.e. the distance between AEP and PEP.

The *stride* λ is the translational distance covered by the main body after a complete cycle.

The previous definitions are mainly used when talking about periodic gaits. The development of omnidirectional and free gaits makes the following notions useful:

- The *support pattern* is the polygon determined by the feet touching the ground. In the case of a static walker, there are permanently at least three contact points.

- The *front stability margin* is the distance between the vertical projection of the robot center of gravity and the support pattern, along the motion direction. There are several kinds of stability margins: for more details see [SONG89].

- The *configuration polygon* is the polygon determined by the six feet.

- The *heading velocity* is a compound of the XY global velocities and the yaw rate of the robot.

These four concepts are sketched in Fig. 7.2.

Heading velocity determines the direction of the robot in a plane parallel to the ground. On unstructured area, orientation and height of the body must be considered. Therefore we define:

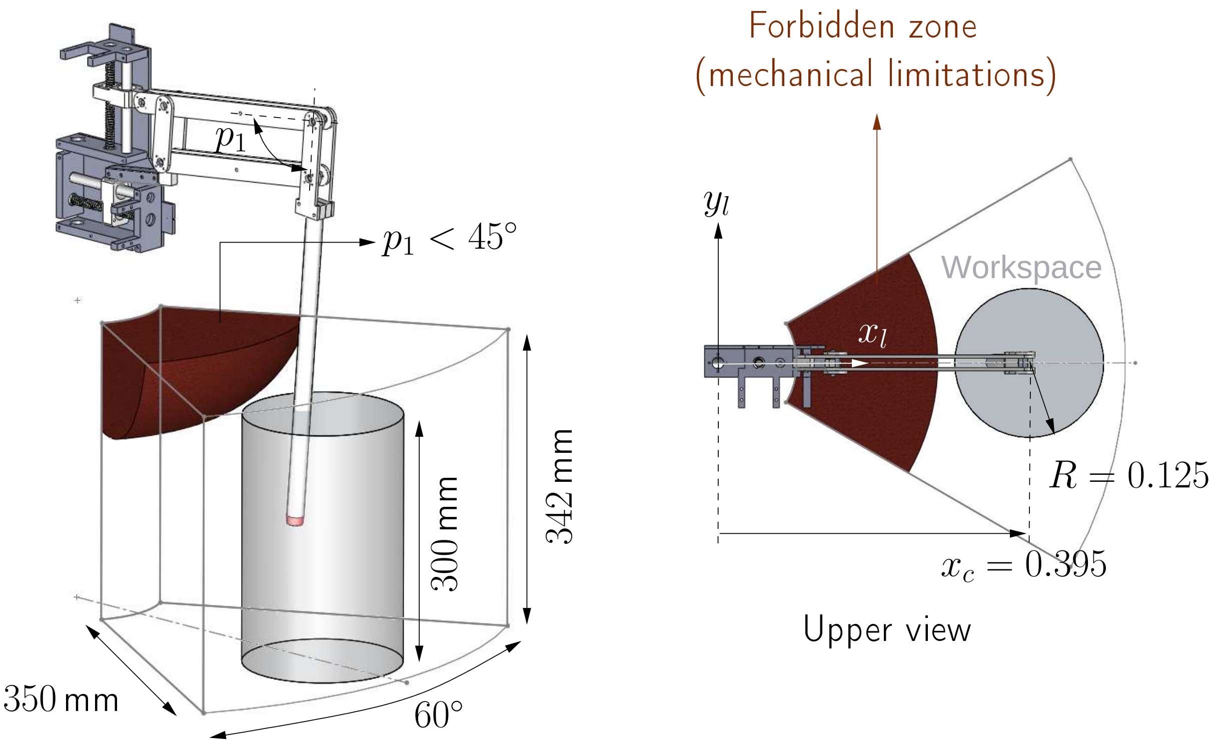


Figure 7.3: Workspace defined for AMRU5

- the *attitude* of the robot which concerns the pitch and the roll of the robot with respect to the vertical global axis;
- the *altitude* of the robot is the height of its center of mass with respect to the ground reference;
- the *posture* of the robot is defined by the position of the six legs with respect to the main body [PORT04].

The *working zone* (or workspace) of a leg is the kinematically admissible space reachable by the foot. If workspace is restricted to admit a certain degree of security on the joint maximal values, it is referred to as the *constrained working volume* [LEE88]. The workspace of an AMRU5 leg is represented in Fig. 7.3. This volume has been voluntary restricted to a simple cylinder which is called workspace throughout the following, instead of "constrained working volume" as it should. The advantage of this volume is that it is quasi impossible to reach a non stable state, because the robot COG is quasi-always inside the support pattern. This helps in this work, because stability issues are not studied. Numerous references discuss the stability of hexapod in the literature; see for example [SONG89, PREU91].

7.1.2 Classes of gaits

The coordination between the legs gives rise to different classes of gaits: for flat ground, *periodic* gaits are recommended because ground level is known in advance. The *gait*

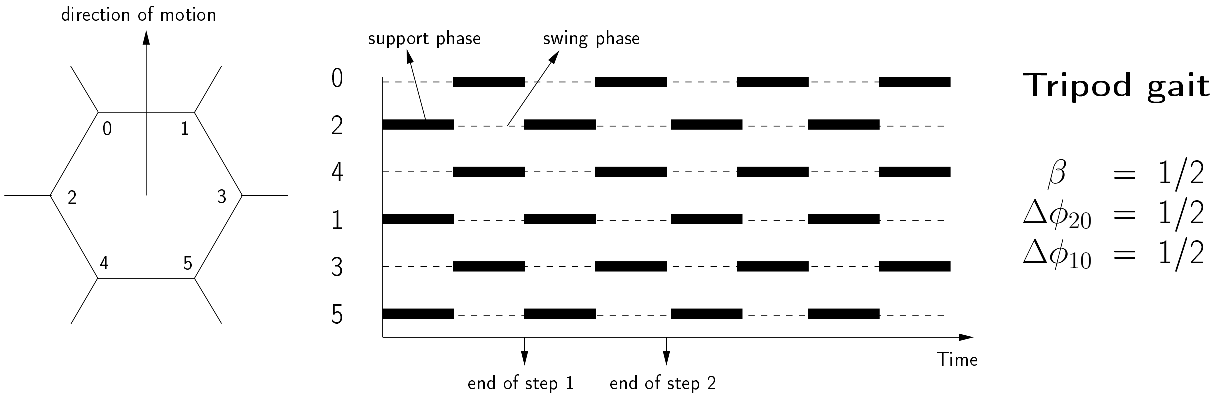


Figure 7.4: Tripod gait pattern

matrix informs about legs phases sequence, or gait pattern. The example of the famous tripod gait is depicted in Fig. 7.4.

When legs motion is not known in advance, or if the time at which a foot touches the ground is unknown, *non periodic* gaits are implemented.

Periodic gaits A periodic gait means that the step cycle is the same for the six legs. The local phase ϕ of a leg is defined by

$$\phi = \frac{t - n_{step}T}{T} \tag{7.1}$$

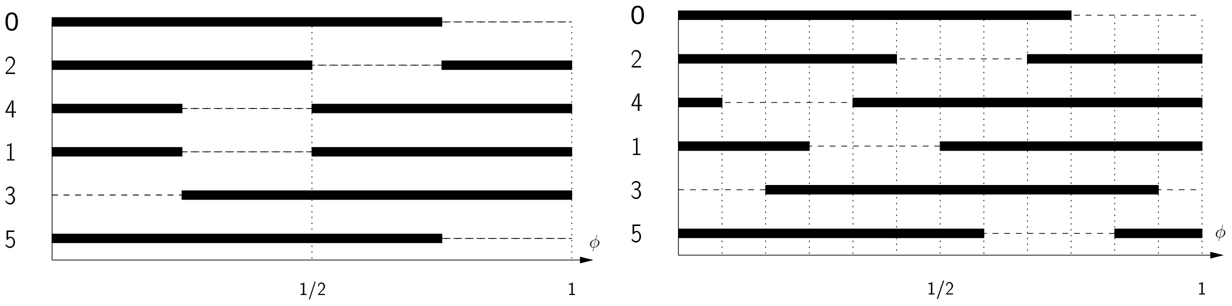
where n_{step} is the number of steps already done, t the current time and T the leg cycle time. It varies from 0 to 1 and helps in the determination of the leg state.

Several periodic gaits exist depending on the duty factor and the phase shift between legs. A gait is called:

- *regular* if each leg has the same duty factor;
- *symmetric* if contralateral[1] legs have a phase shift of 1/2;
- *with constant phase increment* if legs of the same robot side have a constant phase shift ($\Delta\phi_{20} = \Delta\phi_{42} = \Delta\phi_{04}$).

The gait is *forward* if the stepping action occurs from rear to front of the robot. Among this family, the *wave gaits*, used by many animals, give the optimum stability. The gait pattern of the forward wave gait is shown in Fig. 7.5(a). The *equal phase gaits* distribute the landing and takeoff of the feet evenly during the locomotion cycle, which minimize power fluctuation. The gait pattern of the so-called ”Forward Full Cycle Equal Phase Gait” is illustrated in Fig. 7.5(b). Opposite to forward periodic ones, the *backward periodic* gaits begin the stepping at the front, which is propagated to the back. Discussion about stability of periodic gaits is given in [ALEX97]. Their extensive study is found in [SONG89].

[1] *Contralaleral* legs denote opposite legs with respect to the longitudinal symmetry plane of the robot (axis $O_b x_b z_b$ in Fig. 7.1): for instance, leg 0 and 1 are contralateral. The *ipsilateral* legs are on the same side of this symmetry axis. For example, leg 0 and 2 are ipsilateral.



(a) Forward Wave Gait

(b) Forward Full Cycle Equal Phase Gait

Figure 7.5: Forward periodic gait patterns with $\beta = 3/4$ (normalized step cycle time)

Non periodic gaits Obstacles or unstructured ground disturb the periodic coordinations of the legs. The older way to control six-legged robots consisted in moving each leg “manually” by an operator (see GE quadruped in Fig. 2.2(b) for example). Thereafter, this tedious method was replaced by the *follow-the-leader* gait, where only both frontal legs were commanded to attain safety footholds. The other four legs just “followed“ the two frontal legs by automatically reaching these safety footholds.

For thirty years, several gait algorithm have been studied and implemented. Section 7.2 attempts to draw a state of the art of these methods.

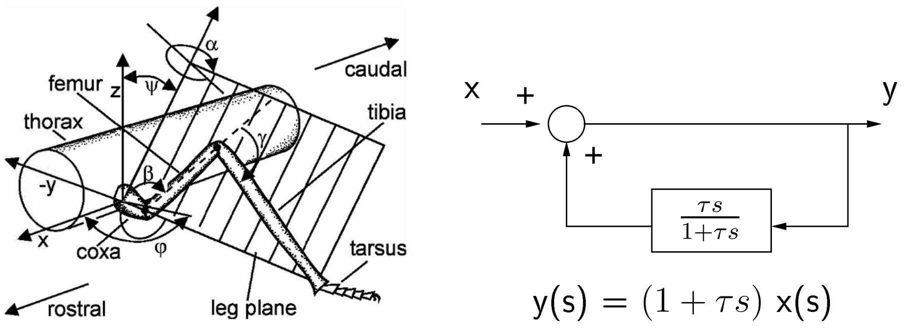
7.2 State of the art in gait generation

In the early 90’s, Brooks sharply criticized the classical top-down approach, which involved an environment building followed by a decision algorithms to define the right thing to do. He qualified it of *sense-model-plan-act*, which takes a lot of time and is not “natural”. He proposes the *subsumption* architecture [BROO91], the first alternative to traditional AI Artificial Intelligence). This bottom-up architecture decomposed the complex task of walking in several modules. For instance, the bottom layer would be the “avoid obstacle” task, linked to the “walk” higher task, itself linked to “explore the world”. Particularity is that modules are organized in layers, all of them accessing the sensor informations.

From this, biologically inspired methods emerged [DELC04]: the aim of these methods is to have a natural behavior of the robots, with “reflexes” and natural reactions conditioning the walking, instead of high-level calculations determining the leg motion. An overview of biologically inspired methods is exposed in Section 7.2.1. The hybrid methods, which are halfway between biological and “engineer” methods are briefly described in Section 7.2.2. Finally, more conventional methods are detailed in Section 7.2.3.

7.2.1 Bio-inspired approaches

Two approaches are denoted in the literature: reflex-based and pattern-based mechanisms. The first solution exploits a lot of sensory stimuli to produce leg motion and gait coordination. The second one delegates the coordination to a central level using CPG (central pattern generator) which can be influenced by external events.



(a) Stick insect leg [CRUS06]

(b) A network with positive feedback and high-pass filter (adapted from [CRUS95])

Figure 7.6: Cruse's et al. researches on stick insect

Reflex-based control Cruse et al. [CRUS09] have a long tradition in reflex-based control. They have mostly studied the stick insect, whose a schematic diagram is presented in Fig. 7.6(a). Each of the support and the swing phases is managed by a low-level neural network, while the transition between these states is provided by the selector network.

The swing network is composed of a simple two-layer feedforward net with three outputs (the rates $\dot{\alpha}, \dot{\beta}, \dot{\gamma}$) and six inputs (actual position of the joints and desired final position at the end of the swing). Some extra inputs can be inserted to emulate the avoidance reflex when tibia or femur of a leg is stucking on an obstacle. This reflex acts like an attractor towards the final swing position. As underlined by the authors, the compensation of disturbances occurs because the system does not compute explicit trajectories, but simply exploits the physical properties of the world.

The control of the support movement [CRUS95] is local for each leg and ensures good coordination thanks to the high-pass filtered positive feedback strategy shown in Fig. 7.6(b). Basically, this scheme of control is unstable, but with the high-pass filter, an impulse on the input leads to a constant value of the output. Practically, Cruse imposes a slight impulse to some leg joints, and the other ones follow and drive smoothly the body because of the positive feedback. As body height is concerned, a classical negative feedback is made on β angle of Fig. 7.6(a) to fight against gravity forces.

The selector network is interpreted as the coordinator of the gait. Six rules, called *coupling mechanisms*, have been identified in behavioral experiments on stick insect [CRUS06]: for instance, a swinging leg prevents the swing of both legs located at the same side of the robot. Another coupling is that the beginning of a support phase excites the start of a swing phase for legs located next to the considered leg. This encourages the swing of a leg, and thus high speed of the robot, but guarantees stability because of the first mechanism. Complement to this, an other neuron takes care of ground contact to start the support phase.

The control architecture employed by Cruse et al. is thus totally decentralized and based on reflexes and inter-leg coupling mechanisms.

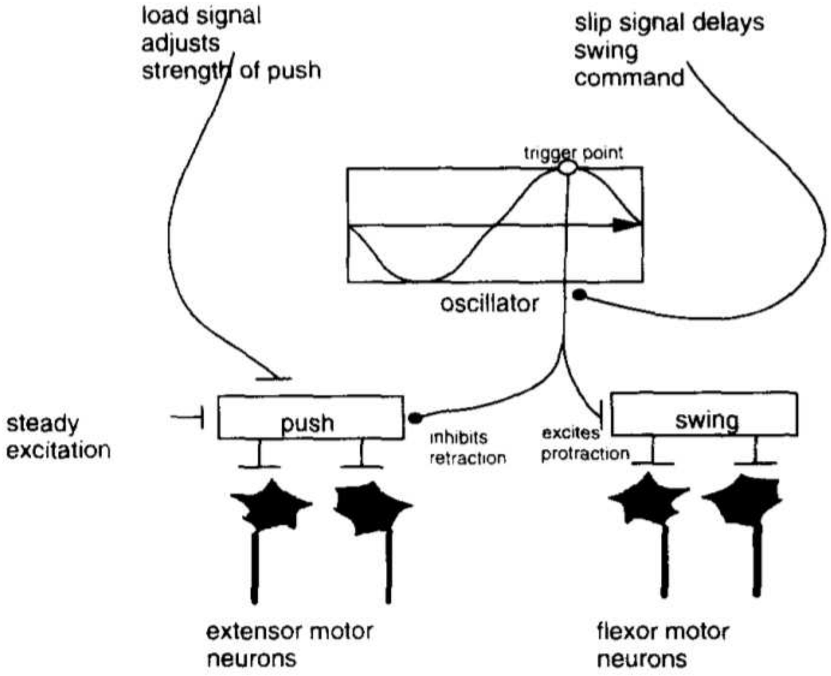


Figure 7.7: Pearson's neural circuit [FERR95]

Central pattern generators The idea of CPG is that walking is a natural process of legged living beings, like breathing. Consequently a central intelligence, where complex calculations occur (associated to the brain) is not required. Instead, a "rhythmic" element, whose pace depends on the environment or the desired type of gait, should give the motion to all the legs.

Pearson and colleagues investigated in the 70's the neural systems that control walking in the cockroach [PEAR76]. In their model (Fig. 7.7), the stepping rhythm of a leg is given by an oscillator. When the peak occurs (trigger point), the swing of the leg is activated and the flexor motor neuron moves the leg forward. The push action is inhibited. Swing time is constant. During the remainder of the cycle, the oscillator generates the support phase by activating the extensor motor neuron. The speed of the support phase depends on the oscillator frequency, which is modified thanks to the sensory feedback. More details can be found in [FERR95].

Collins and Richmond [COLL94] used CPG in quadrupedal locomotion, and showed that a network of non linear oscillators with inhibitory couplings between some legs could produce transitions between the crawl (swing of one leg at a time), the trot (swing of two opposite legs, front right and rear left for example, simultaneously) and the bound (swing of front or rear legs simultaneously), simply by varying the oscillator parameters. The current researches focus on the choice of the oscillators and their adaptation to external events perturbing the leg motion [INAG03, CAPP07].

7.2.2 Hybrid approaches

Literature is rich in original ideas for walking machines: the following gait algorithms are halfway between bio-inspiration and classical engineering.

Reinforcement learning A method to reach a desired behavior for an artificial system (called an *agent*) is to learn by example or experience. If the actions to execute are known in advance, the agent can learn in a supervisory way. Otherwise, it must learn by trials and errors: here comes the notion of *reinforcement*. The agent executes arbitrary actions and is rewarded or penalized depending on the results of its action. The final aim is to maximize the rewards. Zennir [ZENN04] developed a distributed Q-learning reinforcement where each leg executes an action based on the knowledge of the other legs states (on ground or in the air). This learning gives rise to gaits similar to those used by insects on flat ground, among which the tripod gait. Another example is the one of Erden [ERDE08] where a hexapod is amputated from a leg. He shows that continuous patterns are possible to achieve with five legs by using reinforcement learning.

Cyclic genetic algorithm Knowing a priori nothing about how to walk, and defining a few binary numbers (which characterize the elementary motions of the legs), Parker [PARK01] implements a genetic algorithm to learn, in a first time, the step cycle motion to one leg, and in a second time, the coordination between legs to produce the tripod gait.

Behavior-based control Albiez et al. [ALBI03a, ALBI03b] define behavior (or *reflex*) units connected in a network structure (Fig. 7.8). Reflexes can be seen as functions whose inputs are:

- the classical sensory feedbacks (feet forces and joints positions);
- the activation ι varying between 0 and 1 to balance the importance of the block output.

The *behavior activity* a ("how busy is the behavior?") and the *target rating* r ("how far is the behavior from its target?") are also taken into account between the behavior units. Implementation on the quadruped BISAM permits a highly versatile behavior of the robot which adapts to obstacle, supports different load distribution, can walk on a slope etc...

Reactive free gait Porta and Celaya [CELA98, PORT04] developed a very clever and simple free-gait algorithm for gait generation on rough terrain. Unlike other approaches, they keep in mind that, in addition to deal with ground irregularities, the robot maybe has to follow a planified trajectory with more or less precision. If no map of the environment is available, which is mostly the case, the alternative is to use what they call an *alternative control paradigm*. The leg sequence is not planned in advance, but is a result of the robot-environment interaction. Basically, the algorithm is the following:

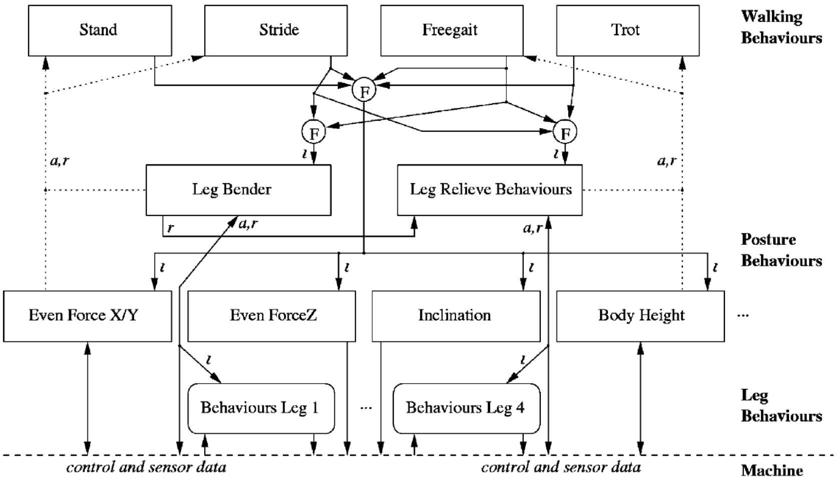


Figure 7.8: Albiez behaviors network [ALBI03a]

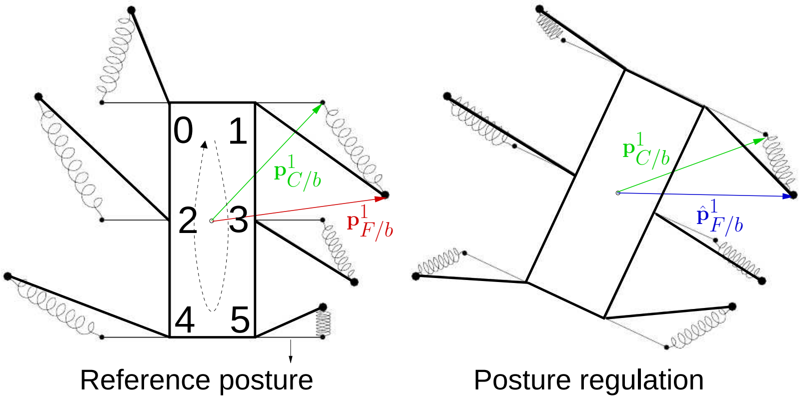


Figure 7.9: Reactive free gait (adapted from [CELA98])

- adequate legs execute their swing and land at relevant footholds;
- heading velocity of the robot is defined according to an arc of circle;
- posture of the robot is simultaneously adapted to the terrain;
- some rules define the next swing candidate(s).

Rules to define the swing candidate(s) are extremely simple: the first one says that a leg can lift up if the *neighboring legs* are on the ground. Neighboring legs appear one next to the other in the clockwise circuit of Fig. 7.9. For instance, neighbors of leg 1 are 0 and 3, and neighbors of leg 4 are 2 and 5.

The second rule is the priority concept, that's to say the selection of the leg to swing. The priority is determined by computing the advance position of each foot with respect to the current turning center of the trajectory. To evaluate this advance position, the angle between the projection in the XY plane of the current feet positions and the center of the corresponding workspace is computed. Porta and Celaya show that with these two principles, and for any arbitrary configuration of the legs, tripod gait is reached on a flat ground.

Note that these two rules belong to a set of global rules called *restrictedness* [FIEL04], where legs workspace interaction and joint mechanical limitations intervene in the choice of the swing leg candidate.

The second task of the controller is the coordination motion of the legs supporting the robot. They use the so-called *posture control mechanism* (posture in the sense of feet position with respect to main body, i.e. including its six DOF), which minimizes a criterion based on the quadratic error between the position to reach for foot i, $\boldsymbol{p}^i_{F/b}$, and a reference posture $\boldsymbol{p}^i_{C/b}$. The posture error is given by:

$$D = \sum_{i=0}^{n} ||\boldsymbol{p}^i_{F/b} - \boldsymbol{p}^i_{C/b}||^2. \tag{7.2}$$

A 2-D example is shown in Fig. 7.9 where leg 1 is considered. At left, the robot is represented with its reference posture with thin lines. Actual robot legs are shown in thick lines, with big dots corresponding to the actual feet positions. Physically, the minimization procedure acts like fictive springs connected between actual and reference positions. They should balance themselves to lead to the right part of the figure, where the equilibrium is reached, and corresponds to the minimal value of D in Eq.(7.2). The gradients relative to heading velocity are constrained to follow a circular trajectory, instead of letting the body completely free. In this way, the body tries to minimize its posture error while following a circular trajectory defined by a radius and an angle.

In this method, the robot body is a consequence of the leg placement. That's why it is classified into the hybrid method, though close to the engineering approach due to the posture minimization algorithm.

7.2.3 Engineering approaches

Two tasks are clearly identified in the engineering approach: the first is the cooperation of the legs in support phase to produce a smooth motion of the main body and to avoid internal constraints; the second concerns the legs synchronization, and can be decomposed into two subquestions: 1) which leg has to swing ? and 2) where to land ?

Body motion control The first task can be solved with kinematic control: on a flat ground, the foot velocity can be computed from the heading velocity of the robot. Hence, the foot is considered as the end-effector of a robot manipulator: its kinematics is inverted to find the desired joint variables from the foot velocity, and a controller is designed to track them accurately. Two dimensional examples of this method can be found in [ORIN82, PREU94].

In addition to the trajectory tracking, terrain adaptation should be realized on uneven ground. In other words, the six DOF of the main robot body have to be taken into account for the kinematic control. Halme et al. [HALM94] developed three modes for attitude and altitude control:

- open adaptability: the vehicle operator commands the body attitude and altitude;
- auto-level: the vehicle body is kept horizontal, whatever the ground profile;
- terrain-following: the body follows the support plane whose situation is estimated from the feet in support phase.

An other widespread solution for terrain adaptation consists in using force sensors to share at best the load of the robot among the supporting feet. Insertion of active compliance at foot has been initially studied in [KLEI80], and has been successfully implemented later on diverse walking machines [DEVJ83, SCHM96, PREU97]. Klein and Briggs [KLEI80] observed that vertical compliance could be implemented to act like a "shock absorber" which adapts to uneven ground, while lateral foot forces give information about the antagonist forces due to the closed kinematics existing in walking machines. Gorinevski and Schneider [GORI90] used the vertical force sensing to distribute smoothly the load of the robot among the supporting legs. Attitude control and leg force optimization are still a current research activity [UCHI00, GALV03, GONZ07, WANG08].

Leg swing For the second task, the question of selecting the adequate candidate has been introduced more than thirty years ago in [MCGH79]. They defined the kinematic margin which is the remaining distance that a leg has inside its working space during the walking. Lee and Orin [LEE88] redefined it with the temporal kinematic margin t_{Si} of leg i by:

$$t_{Si} = \frac{d_i}{v_i} \tag{7.3}$$

where d_i is the distance between the foot and the boundary of the leg working zone, and v_i its velocity. Several algorithms based on kinematic margin have been used in the development of free gaits [LEE88, HALM94, ALEX97].

However, a deadlock situation can occur if two legs arrive quasi-simultaneously at the end of their working zone. For instance, if the two front legs are risen at the same time, the hexapod can loose stability. If only one is swung, the other one could reach the limit of its workspace before the end of the swing. Additional rules for determining the leg candidate have consequently been developed by researchers:

- graph search approach [PAL91]: Pal and Jayarajan search, among the future states of the vehicle, the most appropriate solution to avoid deadlock situations;
- diagonal rules [HIRO84]: Hirose developed three rules based on geometrical constructions to avoid deadlock, guarantee stability and converge to a forward wave gait on flat ground;
- additional common sense rules [HALM94]: Halme et al. based the choice of the leg candidate on the following rules:
  - lift up only one leg at a time;
  - stay in support phase while it is possible;

  - avoid to lift up two neighboring legs (the *neighboring* rule [PORT04]);
  - if many candidates exist, choose the one which maximizes the stability margin.

- an enhancement of Halme et al.'s algorithm has been developed by Alexandre in [ALEX97]: the main contribution is the convergence to a forward wave gait when the robot walks on a flat ground.

The second subproblem refers to the landing position of the leg after the swing phase, also called the foothold. The first strategy is a constant swing time whatever the AEP; the second assumes varying transfer time with a fixed AEP target. In the latter case, the AEP is usually chosen:

- To maximize its kinematic margin after landing [MCGH79,ALEX97]: the drawback is the risk of deadlock situation: for instance, if the heading velocity is sharply modified while the foot has just landed, the leg could easily go out of its working zone.

- To pass through a reference point at the middle of the support phase [ORIN82]: of course, if body velocities are changing, the mid-stance could not occur exactly at the reference point.

In the case of constant swing time, the AEP is either defined as one of the solutions aforementioned, or depends on the maximum predicted step length in the working space. In the latter situation, steps size varies with the body velocity [HALM96].

7.2.4 Gait generation principle for AMRU5

From this review, it seems that recent trends in research focus on gait control with architectures as decentralized as possible, to promote leg local behavior. Walking machines on which these algorithms have been implemented are light and full of sensors to enforce a natural emergent behavior coming from interaction with the environment.

Nevertheless, the aim of this work is not to develop a new bio-inspired algorithm, but a reliable gait control based on an important dynamic study and used to validate a model, thus requiring a completely deterministic behavior of the legs. For each algorithm presented hereafter, the inputs are $\boldsymbol{v}_{b/0}$ and $\boldsymbol{\omega}_{b/0}$, the global velocities, defined as:

$$\boldsymbol{v}_{b/0} = (V_x, V_y, V_z) \tag{7.4}$$

$$\boldsymbol{\omega}_{b/0} = (\Omega_x, \Omega_y, \Omega_z) \tag{7.5}$$

The development of the gait generation follows three steps:

1. The periodic gaits are implemented, and particularly, the tripod gait. The motion of the robot is parallel to the ground, considered as a flat surface. The *heading velocities* (V_x, V_y, Ω_z) are kept constant, the foot and the body clearance are fixed, and the cycle time of a step is constant.

2. The periodic gait with omnidirectional ability is then detailed. Ground is still flat, but the trajectory of the foot in swing is constantly adapted with respect to the global motion of the body: the main body can follow any complex trajectory, defined by arbitrary heading velocities (V_x, V_y, Ω_z).

3. The free gait is finally developed for uneven ground. Two majors differences arise regarding point 2):

   - The gait pattern is not known in advance anymore: some rules are thus required to find the leg swing candidate.
   - The posture correction tends to make the robot body parallel to the estimated ground plane: the approach for computing the ideal posture is similar to the one developed by Porta and Celaya [PORT04]. However, the accurate tracking of trajectory requires a correction algorithm, which computes the heading velocities (V_x, V_y, Ω_z) and the *posture velocities* $(V_z, \Omega_x, \Omega_y)$, different from zero.

The support phase consists in generating the correct joint variable to achieve the desired body motion. It is identical for the three steps aforementioned, and is similar to usual inverse kinematics. The swing phase is different for the three kinds of gaits, particularly for the AEP research.

Even if the top-down approach used here is criticized by most of the bio-inspired researches, it allows an accurate tracking of complex trajectories. This could be desirable for operations where robot could replace the human being, like demining [HUAN02] or drilling [SCHM96]. Moreover, a certain degree of reactivity is ensured by the posture control and the leg synchronization based on the low level ground detection. To summarize, the free gait developed in this work is a compromise between accurate motion of the body and a capability to manage uneven ground without environment modeling.

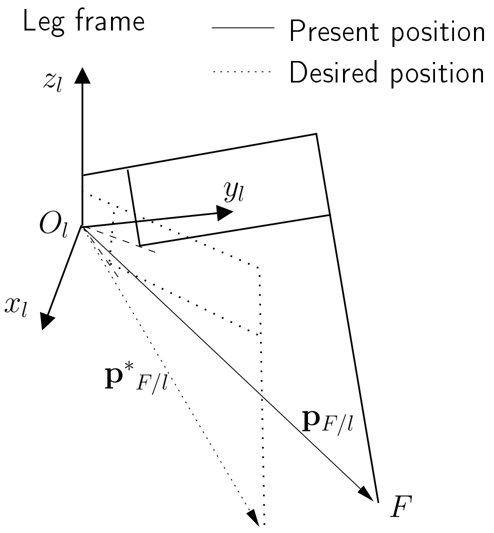


Figure 7.10: Inverse kinematics algorithm

7.3 Kinematic control of a leg

The lowest level of control consists in generating adequate references for the leg joints, from a foot velocity command coming from higher level, depending on whether the leg is in support (Section 7.4) or swing phase (Section 7.5 and 7.7). Thereafter these joint references are sent to PI position controllers which track them. This method is called the *kinematic control*.

Following Fig. 7.10, the foot point position can generally be expressed in the leg local frame by the following relationship:

$$\{\boldsymbol{p}_{F/l}\}_l = \boldsymbol{k}(\boldsymbol{q}) \tag{7.6}$$

where $\boldsymbol{k}$ is the leg kinematic function, relating the joint variables $\boldsymbol{q} = (q_r, q_v, q_h)$ to the foot cartesian position $\boldsymbol{p}_{F/l}$.

From Eq. (7.6), an iterative Newton-Raphson procedure can be used to find the desired joint positions $\boldsymbol{q}^*$ which give the desired foot position $\boldsymbol{p}^*_{F/l}$. The k^{th} iteration of this procedure is

$$\boldsymbol{q}^{*,k} = \boldsymbol{q}^{*,k-1} - \boldsymbol{J}^{-1} \cdot (\boldsymbol{k}(\boldsymbol{q}^{*,k-1}) - \{\boldsymbol{p}^*{}_{F/l}\}_l), \tag{7.7}$$

with $\boldsymbol{J}$ the 3×3 jacobian matrix:

$$\boldsymbol{J} = \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{q}} = \frac{\partial \{\boldsymbol{p}_{F/l}\}_l}{\partial \boldsymbol{q}} \tag{7.8}$$

which is also equal to the matrix gathering the partial velocities (see Eq. (3.25)):

$$\boldsymbol{J} = \frac{\partial \{\boldsymbol{p}_{F/l}\}_l}{\partial \boldsymbol{q}} = \frac{\partial \{\boldsymbol{v}_{F/l}\}_l}{\partial \dot{\boldsymbol{q}}} \tag{7.9}$$

The computation of the jacobian can be estimated in two ways:

- by numerical derivation;

- with direct computations.

The numerical derivation consists in estimating the j^{th} column of the jacobian from the velocity of the foot, by putting successively all the joint velocities to zero excepted one. The sum of Eq.(3.24) vanishes, and the result corresponds to the j^{th} column of the jacobian. This derivation proves exact because of the linearity of the velocity $\boldsymbol{v}$ with respect to $\dot{\boldsymbol{q}}$. The direct computation requires the closed form expression of the partial velocities.

A simplified model of the AMRU5 foot kinematics in the leg local frame has been built with `EasyDyn`. In this case, the Jacobian only gathers translational velocities and no orientations. Practically, both methods are fast enough to use them indifferently: the closed form solution is used herein. Convergence of Eq.(7.7) is reached after 3-4 iterations.

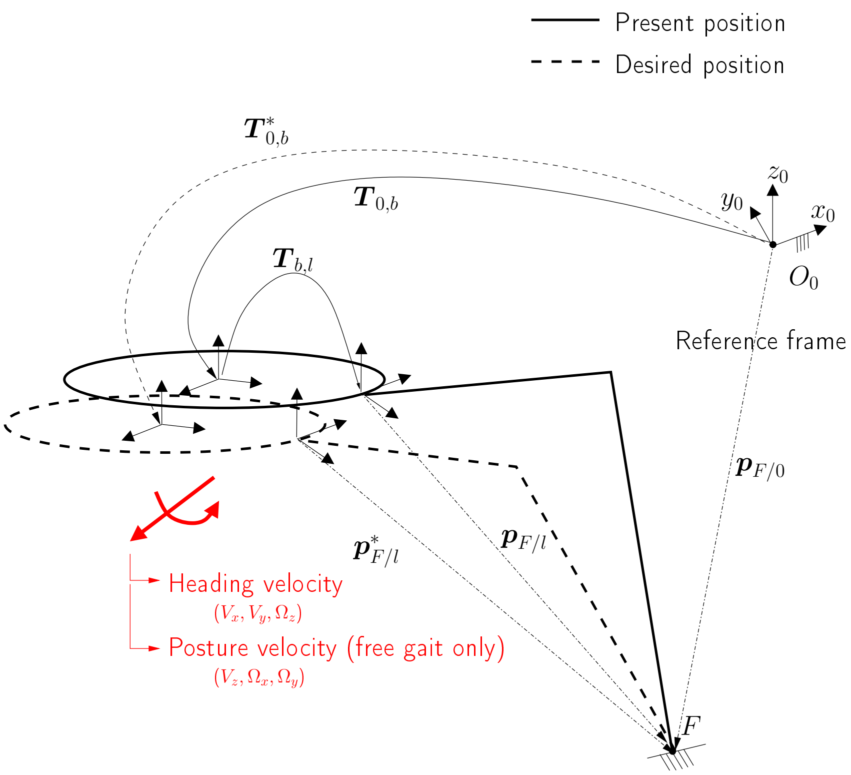


Figure 7.11: Determination of the foot motion from high level commands

7.4 Support phase

The legs in support phase have to cooperate together to produce to desired body motion while keeping internal constraints ideally null.

Figure 7.11 shows a leg in support phase and propelling the body. We can write:

$$\{\boldsymbol{p}_{F/0}\}_0 = \boldsymbol{T}_{0,b} \cdot \boldsymbol{T}_{b,l} \circ \{\boldsymbol{p}_{F/l}\}_l \tag{7.10}$$

Position of the foot in the global reference frame, $\{\boldsymbol{p}_{F/0}\}_0$, is called the *anchor* point in the following, and should be constant. The desired local position of the leg $\{\boldsymbol{p}^*_{F/l}\}_l$ of the leg can be found with:

$$\{\boldsymbol{p}^*_{F/l}\}_l = (\boldsymbol{T}^*_{0,b} \cdot \boldsymbol{T}_{b,l})^{-1} \circ \{\boldsymbol{p}_{F/0}\}_0 \tag{7.11}$$

where $\boldsymbol{T}^*_{0,b}$ is the desired homogeneous transformation matrix of the body, computed from the heading velocities (V_x, V_y, Ω_Z) in the case of gait on flat ground, and from the heading + posture velocities $(V_z, \Omega_x, \Omega_y)$ for the free gait. In the following, the most general case including the six components is presented.

The homogeneous transformation matrix $\boldsymbol{T}^*_{0,b}$ is built every control time step, with the combination of a 3D displacement and three successive rotations (yaw-pitch-roll conventions). The aim is thus to compute $\boldsymbol{T}^{*,k+1}_{0,b}$ from $\boldsymbol{T}^{*,k}_{0,b}, \boldsymbol{v}^k_{b/0}, \boldsymbol{v}^{k+1}_{b/0}, \boldsymbol{\omega}^k_{b/0}, \boldsymbol{\omega}^{k+1}_{b/0}$. At time $k+1$ we have:

$$\boldsymbol{T}^{*,k+1}_{0,b} = \boldsymbol{T}^d(X^{k+1}, Y^{k+1}, Z^{k+1}) \cdot \boldsymbol{T}^{R_z}(\Phi^{k+1}) \cdot \boldsymbol{T}^{R_y}(\Theta^{k+1}) \cdot \boldsymbol{T}^{R_x}(\Psi^{k+1}) \tag{7.12}$$

The estimation of displacements $(X^{k+1}, Y^{k+1}, Z^{k+1})$ from global velocities is straightforward:

$$X^{k+1} = X^k + \frac{V_x^k + V_x^{k+1}}{2} \cdot \Delta t \tag{7.13}$$

$$Y^{k+1} = Y^k + \frac{V_y^k + V_y^{k+1}}{2} \cdot \Delta t \tag{7.14}$$

$$Z^{k+1} = Z^k + \frac{V_z^k + V_z^{k+1}}{2} \cdot \Delta t \tag{7.15}$$

For the rotations, the relation between the yaw-pitch-roll rates $(\dot{\Phi}, \dot{\Theta}, \dot{\Psi})$ and the global rotational velocities $(\Omega_x, \Omega_y, \Omega_z)$ is not linear. Effectively, we have:

$$\{\boldsymbol{\omega}_{b/0}\}_0 = [\boldsymbol{J}_\omega]_0 \cdot \begin{pmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi} \end{pmatrix} \tag{7.16}$$

where the matrix $\boldsymbol{J}_{\boldsymbol{\omega}}$ is the velocity transformation matrix of the robot body depending on (Φ, Θ, Ψ) (see Eq. 3.33), and where $(\Omega_x, \Omega_y, \Omega_z)$ have been gathered in $\{\boldsymbol{\omega}_{b/0}\}_0$. The computation of $(\Phi^{k+1}, \Theta^{k+1}, \Psi^{k+1})$ is made by the following algorithm:

- $\boxed{\text{step 1}}$: first estimation (prediction) of the yaw-pitch-roll angles with the rates computed at the previous control time :

$$\Phi^{k+1} = \Phi^k + \dot{\Phi}^k \cdot \Delta t \tag{7.17}$$

$$\Theta^{k+1} = \Theta^k + \dot{\Theta}^k \cdot \Delta t \tag{7.18}$$

$$\Psi^{k+1} = \Psi^k + \dot{\Psi}^k \cdot \Delta t; \tag{7.19}$$

- $\boxed{\text{step 2}}$: compute the new jacobian matrix $\left[\boldsymbol{J}_\omega^{k+1}\right]_0$;

- $\boxed{\text{step 3}}$: compute the new yaw-pitch-roll rates:

$$\begin{pmatrix} \dot{\Phi}^{k+1} \\ \dot{\Theta}^{k+1} \\ \dot{\Psi}^{k+1} \end{pmatrix} = \left[\boldsymbol{J}_\omega^{k+1}\right]_0^{-1} \cdot \{\boldsymbol{\omega}_{b/0}^{k+1}\}_0; \tag{7.20}$$

- $\boxed{\text{step 4}}$: correction of the first estimation:

$$\Phi^{k+1} = \Phi^k + \frac{\dot{\Phi}^k + \dot{\Phi}^{k+1}}{2} \cdot \Delta t \tag{7.21}$$

$$\Theta^{k+1} = \Theta^k + \frac{\dot{\Theta}^k + \dot{\Theta}^{k+1}}{2} \cdot \Delta t \tag{7.22}$$

$$\Psi^{k+1} = \Psi^k + \frac{\dot{\Psi}^k + \dot{\Psi}^{k+1}}{2} \cdot \Delta t; \tag{7.23}$$

Note that step 3 and 4 could be iteratively processed, but practically one iteration is accurate enough. The desired HTM $\boldsymbol{T}_{0,b}^{*,k+1}$ of the main body is then computed and the new position of the foot in the leg local frame (Eq. (7.11)) can be updated for each leg in support phase, with $\{\boldsymbol{p}_{F/0}\}_0^k$.

In simulation, the tracking is very good because the body position and anchor point are known exactly. However, on the real robot, there is presently no navigation system, thus (X, Y, Z) and (Φ, Θ, Ψ) are estimated by assuming that the robot is following perfectly its trajectory. Moreover, the anchor point is computed once at the beginning of the support phase, assuming there is no slipping. In this work, the robot never walks during a long period and over long range, which limits the tracking error. Problem of accurate location of walking machines in outdoor environment is addressed in [COBA08, COBA09].

A video illustrating the support phase algorithm, with six legs on the ground, is shown on YouTube [2]. Each component of $\boldsymbol{v}_{b/0}$ and $\boldsymbol{\omega}_{b/0}$ evolutes separately, according to a sine motion.

[2] http://www.youtube.com/watch?v=T8VoPO41Pns

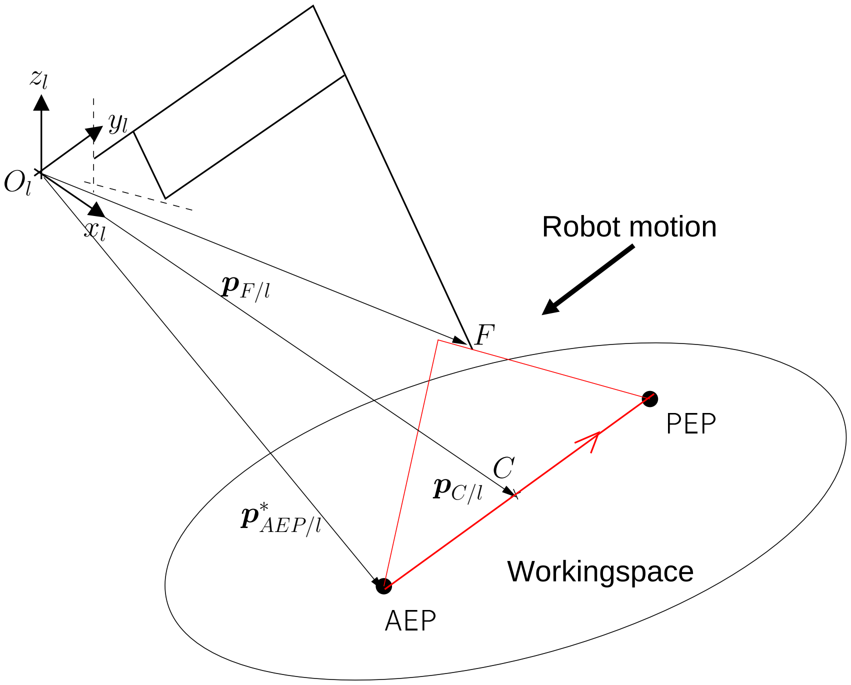


Figure 7.12: Basic cycle of a foot

7.5 Swing phase

For swing phase, the important point is to determine which position the foot has to reach at the end of its swing. The step cycle time is constant in our approach, therefore the AEP will change with the robot velocity.

7.5.1 Periodic gait with constant straightforward velocity

This ideal case is often used to illustrate the different kinds of periodic gait patterns. The straightforward velocity of the robot V_x is constant, while V_y and Ω_z are null. The swing profile is depicted in Fig. 7.12. The position of the AEP to reach at the end of the swing is given by:

$$\boldsymbol{p}^*_{AEP/l} = \boldsymbol{p}_{C/l} + \boldsymbol{v}_{C/0} \cdot \frac{\beta T}{2} \tag{7.24}$$

where C is the center of the leg workspace (rigidly connected to the central body), $\boldsymbol{p}_{C/l}$ its position in the leg frame, and $\boldsymbol{v}_{C/0}$ its velocity in the global frame. The latter is expressed by:

$$\boldsymbol{v}_{C/0} = \boldsymbol{v}_{b/0} + \boldsymbol{\omega}_{b/0} \times \boldsymbol{p}_{C/b} \tag{7.25}$$

which is also valid for omnidirectional gait. The interest of passing through the point C is to obtain a stroke centered in the leg workspace. This swing velocity is simply expressed by:

$$\boldsymbol{v}_{swing/l} = \frac{\boldsymbol{p}^*_{AEP/l} - \boldsymbol{p}^*_{PEP/l}}{(1-\beta)T} \tag{7.26}$$

and gives the velocity of the foot with respect to the leg. The swing velocity is constant if walking is established. Effectively, during the first step, the stroke is not centered in the workspace, thus $\boldsymbol{v}_{swing}$ is not the same as during the "steady-state" walking.

The generation of the foot position in swing at time $k+1$ is differentiated between the plane parallel to the robot body and its normal direction. For the planar motion, we have:

$$\begin{aligned} p^{*,k+1}_{x,F/l} &= p^{*,k}_{x,F/l} + v_{x,swing}\Delta t \quad (7.27)\\ p^{*,k+1}_{y,F/l} &= p^{*,k}_{y,F/l} + v_{y,swing}\Delta t \quad (7.28)\end{aligned}$$

while a simple up and down profile is assigned for the foot vertical motion. An example of straight walk is shown on YouTube [3] where the robot moves forward at 2 cm/s, during 60 s. Then it stops and rotates of 90 degrees, and comes back. The position error after 2 m (the first straight line) is about 5 cm, and the angular error is approximately 10 degree.

7.5.2 Omnidirectional periodic gait

Omnidirectionality is the capability of the algorithm to manage any changes in the heading velocities, at any moment. Another definition is that the robot has to follow **continuously** a trajectory in a plane, **whatever its complexity**. We work with the following assumptions:

- a flat ground;
- a constant period T for the leg cycle;
- a velocity adapted in such a way that legs stay inside their workspace.

Note that, in Section 7.7, these hypotheses are given up to come to the free gait.

The support phase already includes omnidirectional ability. The keypoint consists in reaching an adequate AEP so that the robot can continue its motion fluently. As in Section 7.5.1, the vertical and horizontal motion of the foot are decoupled; the vertical one follows the same rule as before, while the horizontal motion is a bit more complex. Unlike the case of periodic gait with a fixed heading velocity, the foothold can not be computed in advance.

The AEP is now recomputed at each control time, because velocity can change at any moment. The foot has to travel a distance $\boldsymbol{p}^*_{AEP/l} - \boldsymbol{p}_{F/l}$ in a time $T_{swing} - t_{local}$ where T_{swing} is the time allowed for the swing (i.e. $T \cdot (1-\beta)$) and t_{local} is the time elapsed since the beginning of the swing:

$$\boldsymbol{v}_{swing} = \frac{\boldsymbol{p}^*_{AEP/l} - \boldsymbol{p}_{F/l}}{T_{swing} - t_{local}}. \quad (7.29)$$

The denominator of Eq. (7.29) can lead to very high velocity at the end of the swing phase: to avoid a runaway, a saturation limits the velocity. The latter is twice the swing velocity in the case of a steady-state motion of the robot. The new reference position of

[3] http://www.youtube.com/watch?v=dhpga3h2m9I

the foot $p^{*,k+1}_{x,F/l}$ is computed as in Section 7.5.1. With the continuous computation of the target point, the feet stroke is automatically centered around the workspace center in 1 or 2 steps.

7.5.3 Results of omnidirectional motion

This section illustrates the efficiency of the algorithm for complex trajectory. The dynamic model of AMRU5 without friction in the joints has been used to develop the omnidirectional gait: actually, friction is not essential for development of gait algorithms, and considerably slows down the simulation time.

Inputs of the systems are the heading velocities (V_x, V_y, Ω_z). Because there is no joystick or other input stream allowing a direct command of these references, three kinds of paths have been computed off-line. Time to travel the distance is fixed so that the resulting velocities are not too important to avoid a leg going out of its working space. The paths are shown in Fig 7.13 and consist of:

- a back and forth motion: the beginning of the motion is commanded by V_x=0.015 m/s and changes abruptly to V_x=-0.02 m/s at t=140 s;
- straight lines with sharp direction changes: the alternating between reference velocities is indicated in Fig. 7.13; the yaw rate remains equal to zero;
- and a spline with continuously varying heading velocity: here the three components (V_x, V_y, Ω_z) are different from zero.

Back and forth motion This case highlights the capability of the swing phase to deal with a velocity reversal of the body motion. The reference motion X and the tracking error $X - q_0$ are shown in Figure 7.14. The tracking is perfect, because we assume that there is a navigation system accurate enough to give the exact position of the main body. Because of the dynamic nature of the feet/ground contact, the transitions between leg triplets give rise to some disturbances, which do not exceed 2 mm.

Fig. 7.15 highlights the adaptation of foot 1 trajectory when the velocity reversal occurs. The velocity change has been encircled and arrows indicate the direction of the path traveled by the foot. While the foot is swung forward, the change in heading velocity results in a new target point, located approximately at the opposite of the previous one, but further such that the stroke is higher because of a greater velocity.

Straight lines In this application, straight lines are traveled by the robot body (Figures 7.16). Again the tracking is excellent, and the deviations from the reference trajectory occur because of transition on leg triplets. They do not exceed 3 mm.

The foot 0 trajectory adaptation is shown in Fig. 7.17 for $t = 50$ s. The swing phase is successfully adapted, as well as the stroke. Note, however, that the stroke on the ground is not a line, while heading velocity is along one direction. Effectively, the foot is slipping on the ground and is deviated from its expected trajectory. This is not a problem if the

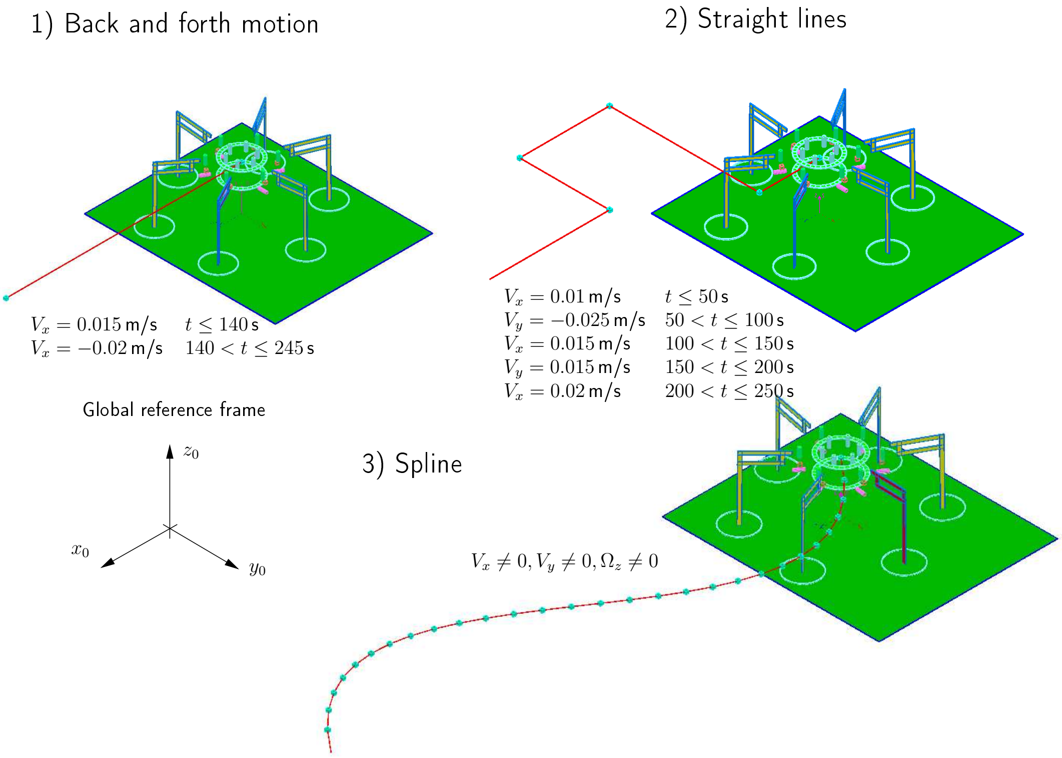

1) Back and forth motion
$V_x = 0.015\,\mathrm{m/s}$ $t \leq 140\,\mathrm{s}$
$V_x = -0.02\,\mathrm{m/s}$ $140 < t \leq 245\,\mathrm{s}$
2) Straight lines
$V_x = 0.01\,\mathrm{m/s}$ $t \leq 50\,\mathrm{s}$
$V_y = -0.025\,\mathrm{m/s}$ $50 < t \leq 100\,\mathrm{s}$
$V_x = 0.015\,\mathrm{m/s}$ $100 < t \leq 150\,\mathrm{s}$
$V_y = 0.015\,\mathrm{m/s}$ $150 < t \leq 200\,\mathrm{s}$
$V_x = 0.02\,\mathrm{m/s}$ $200 < t \leq 250\,\mathrm{s}$
Global reference frame
z_0
x_0
y_0
3) Spline
$V_x \neq 0, V_y \neq 0, \Omega_z \neq 0$

Figure 7.13: Example of paths requiring an omnidirectional ability

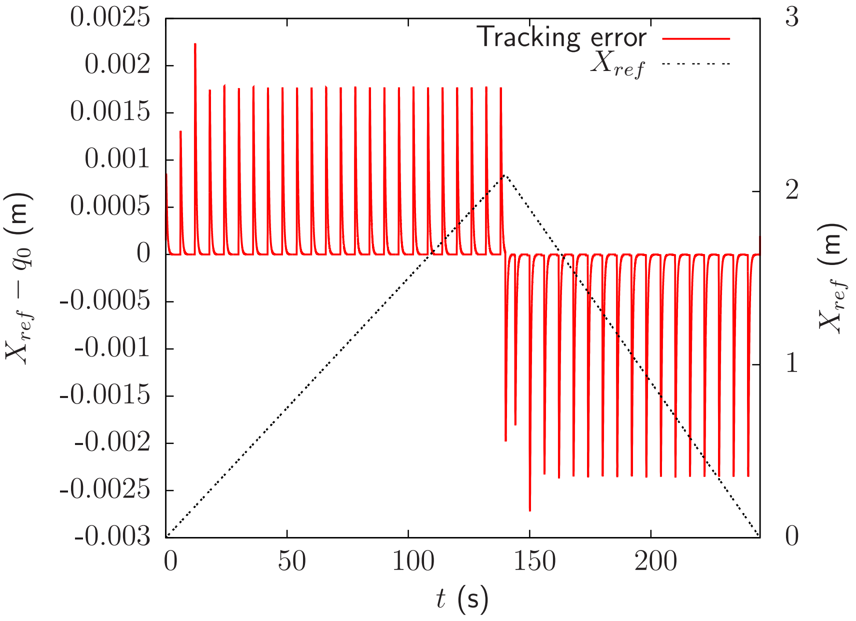


Figure 7.14: Body motion for a back and forth motion

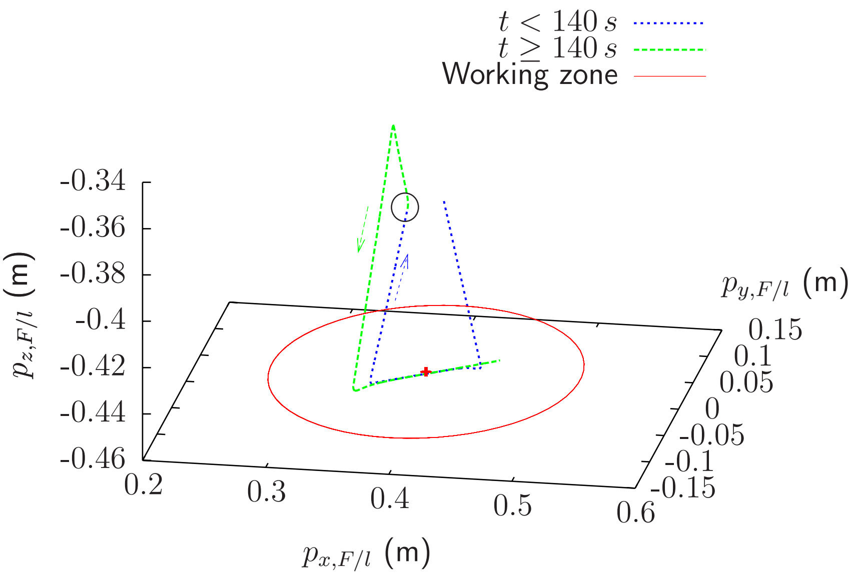


Figure 7.15: Foot trajectory adaptation for a heading velocity reversal (foot 1)

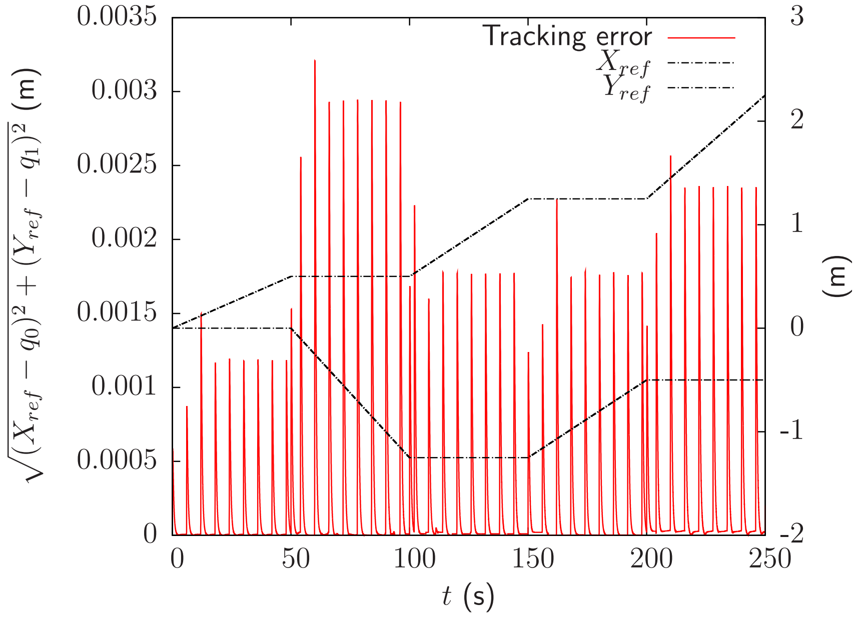


Figure 7.16: XY references and tracking error magnitude vs. time (straight lines)

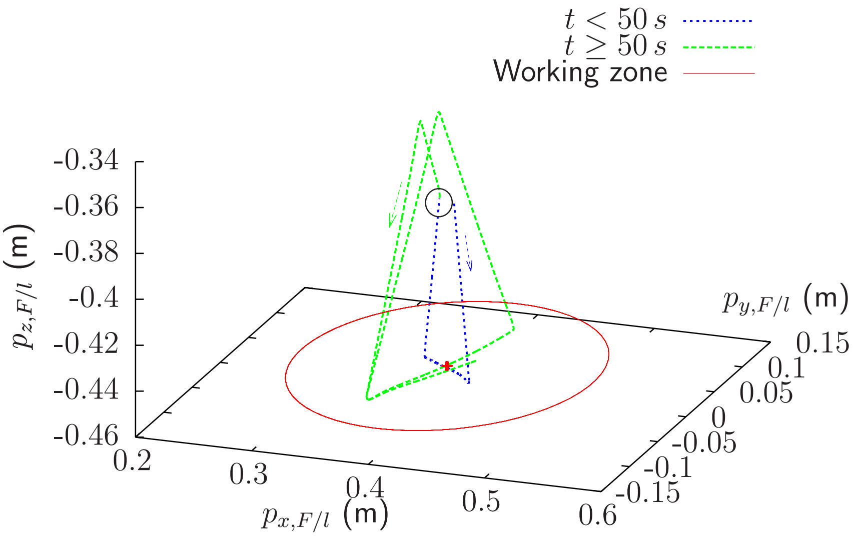


Figure 7.17: Foot 0 trajectory adaptation in case of a motion change of 90 degrees

navigation system is able to give the real position of the robot body. If no navigation is available, the main body position is supposed to follow perfectly the reference. This leads to unavoidable errors in the determination of the anchor point. Consequently the tracking is not so good, as shown in Fig. 7.18.

Spline Spline is a good illustration of continuously changing velocity The robot body must follow the spline and adapts its yaw to stay tangential to the XY trajectory. Figure 7.19 shows the evolution of the robot with and without navigation system. The corresponding tracking errors are illustrated in Fig. 7.20.

Figure 7.21 shows the continuous adaptation of the foot trajectory for time between 0 and 100 s. Because of the yaw rate different from zero, the stroke of the foot is now curvilinear.

A video available on YouTube[4] shows the evolution of the real robot for a spline trajectory. The video has been accelerated four times. Despite of the constant variations of the heading velocities, the legs stay correctly centered in their workspace.

[4] http://www.youtube.com/watch?v=IEa94B92L04

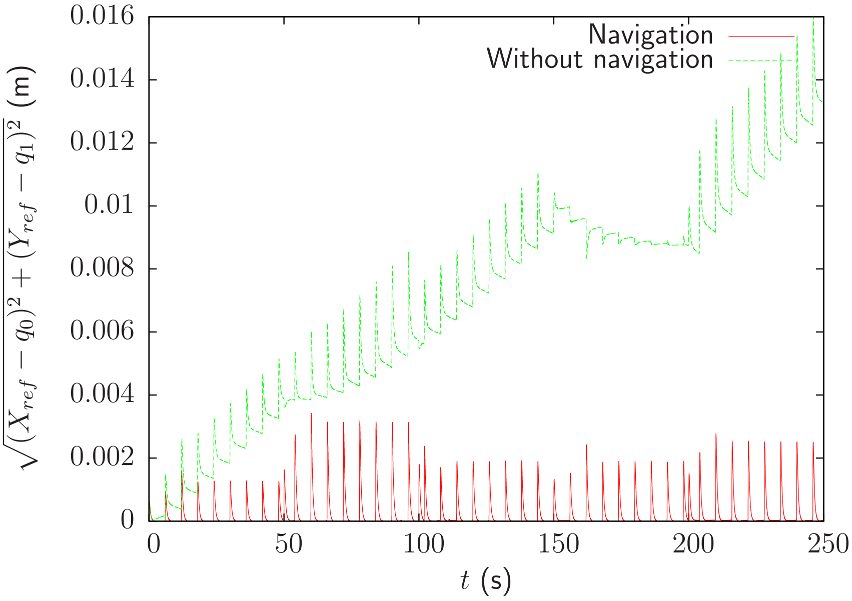


Figure 7.18: Tracking error magnitude with/without navigation system (straight lines)

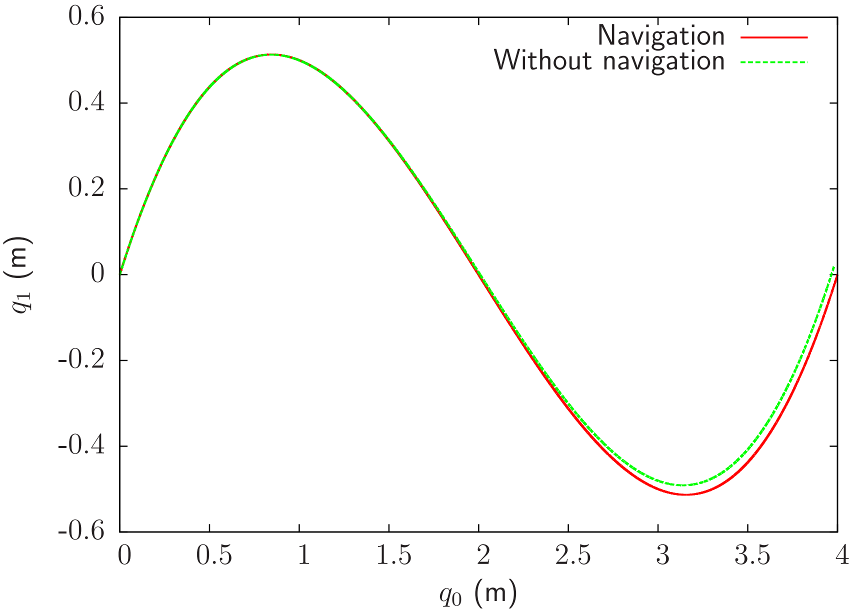


Figure 7.19: Body trajectory in the ground plane (spline)

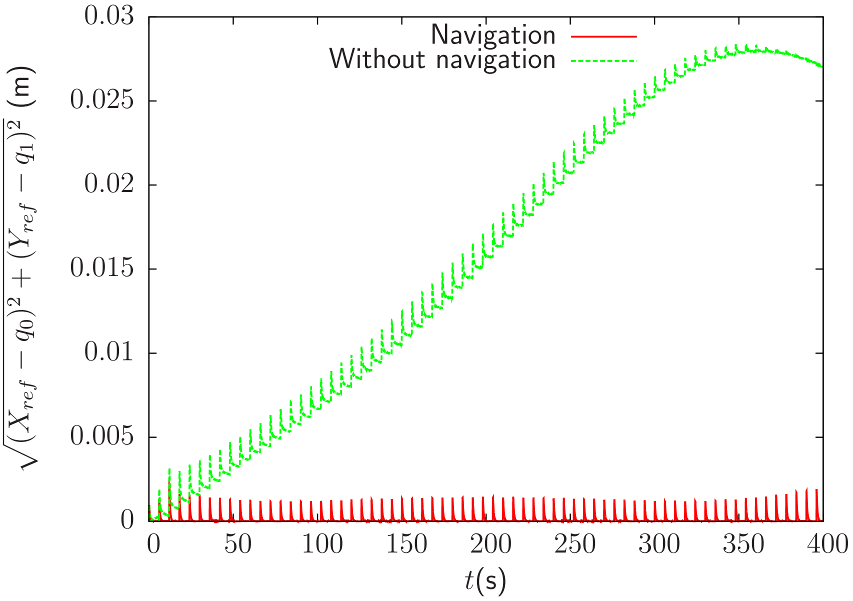


Figure 7.20: Tracking error with/without navigation system (spline)

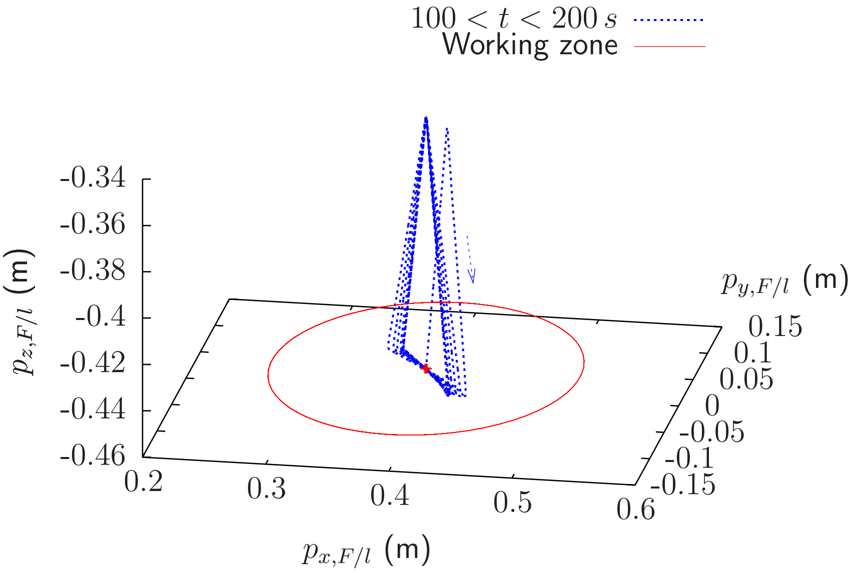


Figure 7.21: Foot zero stroke and swing adaptation for the spline motion

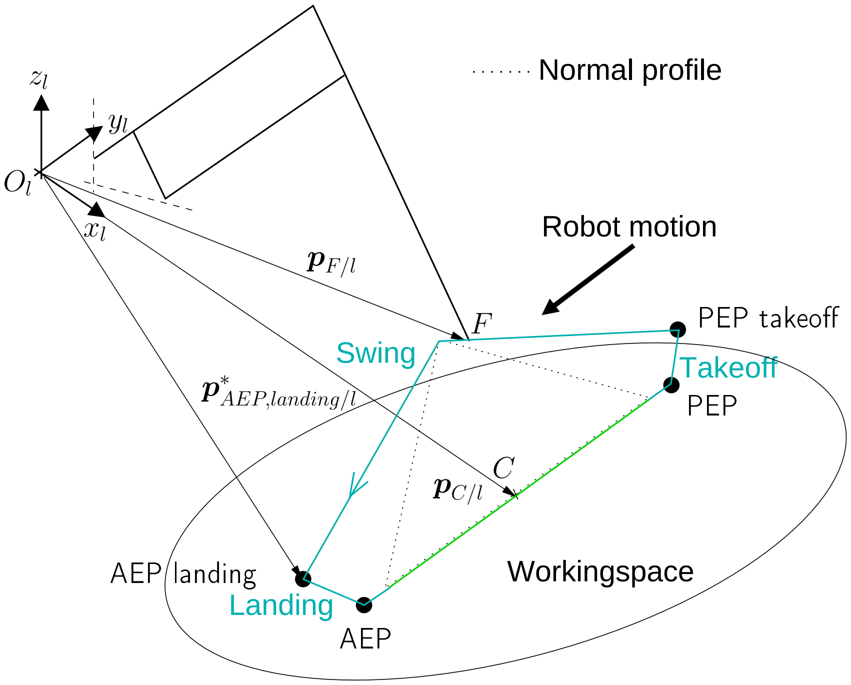


Figure 7.22: Enhanced cycle of the foot

7.6 Enhancement of the periodic gait

Up to now, we described how to generate a gait to produce a smooth motion of the robot, whatever the trajectory, provided that we stay in the framework of periodic gaits. However, such an implementation on the real robot is not recommended. The lack of rigidity of the horizontal joint makes that the foot is not immediately lifted up as the vertical joint is moved. Hence, if the foot forward motion is executed while it is still on the ground, antagonist forces arise. They disturb the gait and force the joint transmissions, and are consequently undesired.

Figure 7.22 shows the usual swing profile (dotted line), and an enhancement profile of the swing (blue line) [5]. The aim is to lift attain a certain high ΔZ sufficiently high to begin the forward motion to the AEP.

The support phase is similar in both cases. The swing phase is divided in four subphases. The differences between the normal and improved foot profiles are depicted qualitatively in Fig. 7.23, where trajectory of feet 2 and 3 are described in plane $O_l y_l z_l$.

- PEP (and AEP) overlap (equivalent to the double stance in human walking): the support phase is slightly extended to avoid a simultaneous landing and takeoff of the legs, which results in a better load transition. This is the case during $0.5 < \phi \leq 0.5 + \frac{\Delta\phi}{2}$ for foot 2;

- Takeoff: the foot keeps the same horizontal motion as in support phase but is lifted up with a constant velocity. Takeoff occurs during $0.5 + \frac{\Delta\phi}{2} < \phi \leq 0.5 + 2\Delta\phi$ for

[5] Note that this enhancement is the default implementation on AMRU5. A direct generation of the legs trajectory as presented before damages the sprocket chain transmission of the rotational joints.

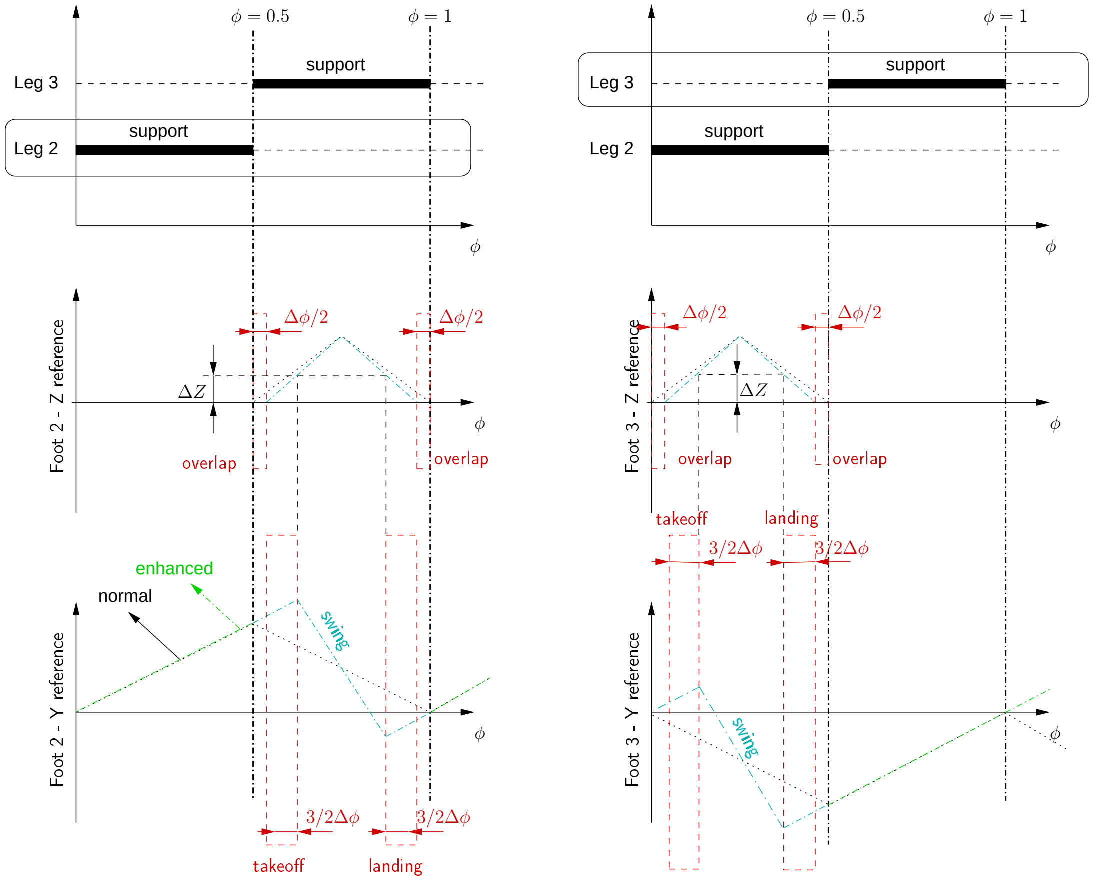


Figure 7.23: Detail of the enhanced leg cycle for legs 2 and 3

foot 2;

- Swing itself: the foot is brought to the AEP position, in such a way that the stroke is centered in the workspace. The (x,y) local coordinates of a virtual AEP at the end of the swing (or at the beginning of the landing) are computed once at the beginning of the swing:

$$\boldsymbol{p}^{*}_{AEP,landing/l} = \boldsymbol{p}_{C/l} + \boldsymbol{v}_{C/0} \cdot \frac{(\beta + 4\Delta\phi)T}{2} \tag{7.30}$$

  The horizontal velocity of the foot for a fixed swing time is:

$$\boldsymbol{v}_{swing} = \frac{\boldsymbol{p}^{*}_{AEP,landing/l} - \boldsymbol{p}^{*}_{PEP,takeoff/l}}{(1 - (\beta + 4\Delta\phi))T} \tag{7.31}$$

  where $\boldsymbol{p}^{*}_{PEP,takeoff/l}$ is the initial foot position at the end of the takeoff, and $(1 - (\beta + 4\Delta\phi))T$ is the swing time. The vertical motion of the foot is still lifted up with constant velocity until the half of the swing; then it goes down with the same velocity.

- Landing: similar to the takeoff, the horizontal motion corresponding to the support phase already begins while the foot is going down at constant velocity. A virtual AEP is computed once at the beginning of this phase, to allow the execution of the support phase algorithm. This is the case during $1 - 2\Delta\phi < \phi \leq 1 - \frac{\Delta\phi}{2}$ for foot 2.

Note that the idea of overlapping is not new [SHIH87], but it has not been formalized as in this work. The vertical compliance of the leg could be introduced in our simulations by changing the soil stiffness parameter. We chose a softer soil with K_{gnd}=1e4 N/m^{p_K}, which produces a penetration of 49 mm if the robot is standing on three legs. This seems important but it was actually observed before the replacement of the horizontal bearings. Under these conditions, we simulated a straight displacement of the robot with a velocity of 0.02 m/s, without navigation system, as it would be the case on the real robot.

Fig. 7.24 presents the evolution of the main body q_0 parameter, for a constant velocity V_x =2 cm/s. With the enhanced profile, the reference is correctly tracked, while with normal profile, antagonist forces disturb the motion of the robot, which quickly induces a drift in the reference tracking. The case without navigation is naturally considered (as for the real implementation).

Remark that this improved algorithm is interesting in the case of tripod gait, but becomes less relevant as duty cycle of the legs increases, because load transition is not so hard anymore. A tripod gait with this enhancement has been used for the validation of our dynamic model in Section 6.

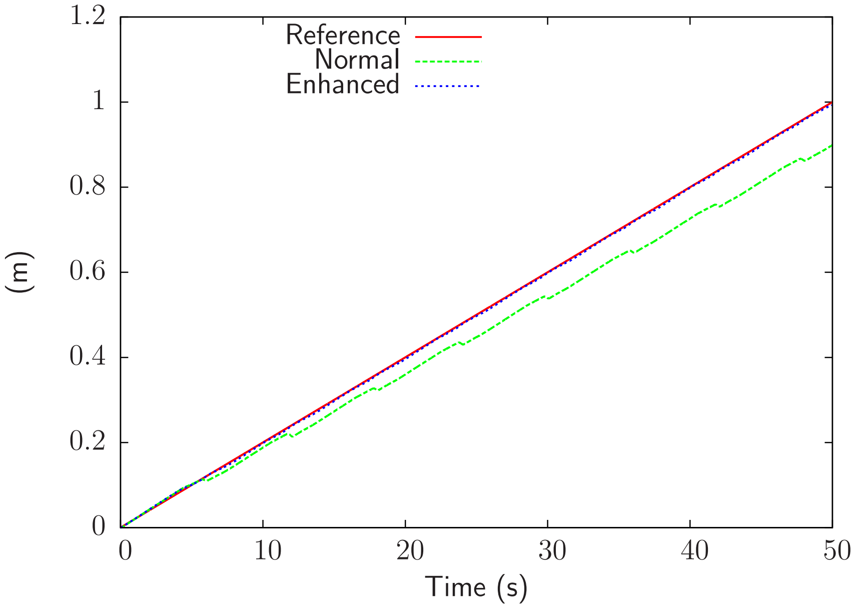


Figure 7.24: Tracking of a straight line on soft ground

7.7 Free gait

The last step to obtain an autonomous robot able to walk in unstructured environment is to deal with terrain variations. The algorithm developed here is adapted from those of Porta and Celaya [PORT04] presented before, with a major difference: our approach is top-down based, the leg motion is a consequence of the desired body motion. In [PORT04], the body motion is an emergent behavior of the legs positioning.

The free gait can be seen as a generalization of the omnidirectional periodic gait, where two points require special attention:

- The user imposes a trajectory in the horizontal plane (thus a global heading velocity). The robot body has to track it accurately while keeping a posture consistent with the relief. The posture algorithm of Section 7.7.1 and the global velocity correction of Section 7.7.2 manage this aspect.

- The ground touching is not known in advance anymore: hence the leg coordination is totally unpredictable and some extra conditions are specified to allow the lifting of the foot. This is detailed in Section 7.7.3.

Detection of the ground is different in simulation or on the real robot: normal force is known in simulation, and thus we consider that ground is detected when the normal force is higher than a threshold of 10 N. But on the real robot, method of ground detection presented in Section 5.6.3 is employed.

Complete algorithm is summarized in Section 7.7.4 and simulations results are presented in Section 7.7.5.

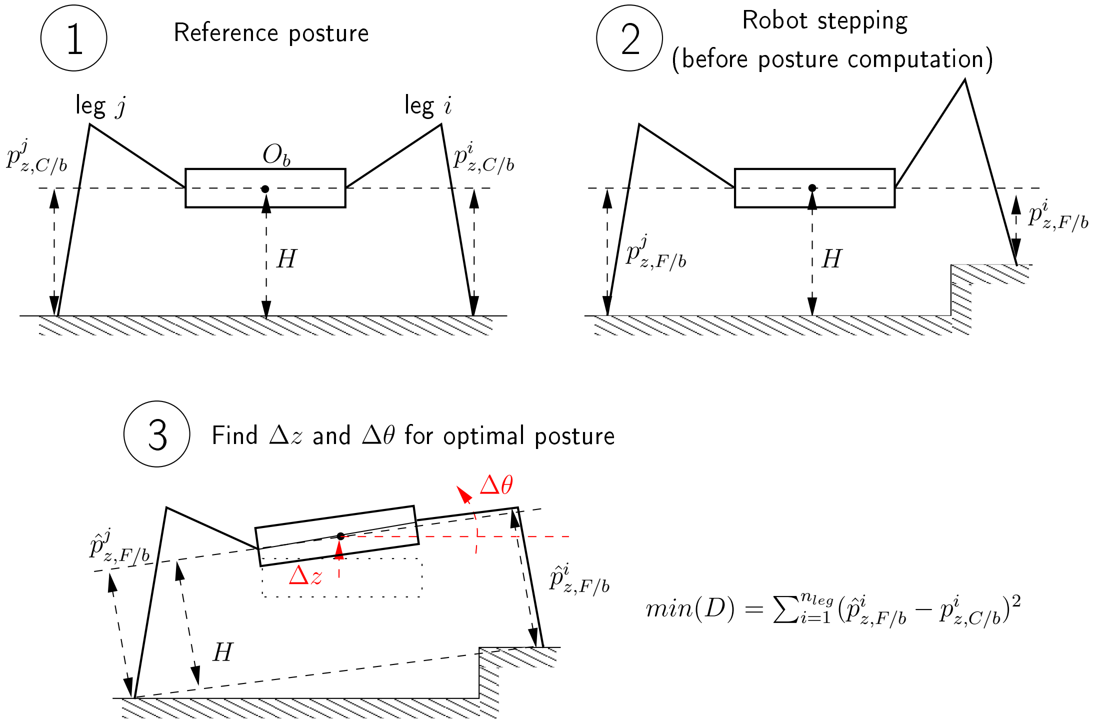


Figure 7.25: Posture adaptation : 2-D case

7.7.1 Posture control

The basic idea is to keep the algorithm managing the support phase intact, and to provide to it correct inputs so that the posture can be controlled. The posture control is a local phenomenon which only concerns $\dot{\Theta}$, $\dot{\Psi}$, the pitch and roll rates, and V_z the vertical velocity. The condition for posture calculation is called Rule 1 and is noted:

Rule 1

Compute the posture if a new leg touches the ground

Figure 7.25 depicts the posture control algorithm in 2-D. The initial case ① shows the reference posture. Assume that the hexapod encounters a step as in ②. The posture control consists in finding the adequate Δz and $\Delta\theta$ ③ to minimize Eq. (7.32).

Problem comes down to find $\Delta z, \Delta\psi, \Delta\theta$ which minimize the criterion:

$$D = \sum_{i=1}^{n_{leg}} (p^i_{z,F/b} - p^i_{z,C/b})^2 \tag{7.32}$$

where $p^i_{z,F/b}$ is the vertical position of foot i in the body frame, and $p^i_{z,C/b}$ a reference corresponding to a well-conditioned posture. The sum of Eq. (7.32) takes into account the six legs: for those in swing, the z coordinate of the last contact point is used. Equation (7.32) is a a simplification of Eq. (7.2). For the considered leg, the computation of

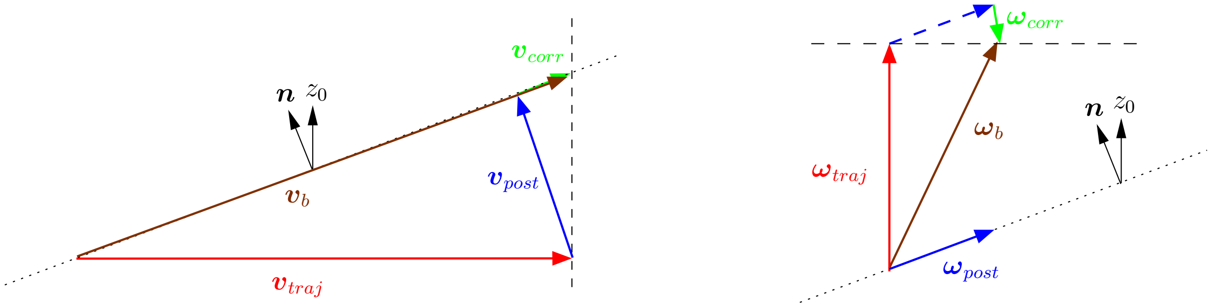


Figure 7.26: Correction of global velocities

$\boldsymbol{p}_{F/b}$ is made by:

$$\{\boldsymbol{p}_{F/b}\}_b = (\boldsymbol{T}_{posture})^{-1} \circ \{\boldsymbol{p}^0_{F/b}\}_b. \tag{7.33}$$

where $\boldsymbol{p}^0_{F/b}$ is the initial foot position before posture computation, and $\boldsymbol{T}_{posture}$ is a local kinematic model of the body, defined by:

$$\boldsymbol{T}_{posture} = \boldsymbol{T}^d(0,0,\Delta z) \cdot \boldsymbol{T}^{R_y}(\Delta\theta) \cdot \boldsymbol{T}^{R_x}(\Delta\psi). \tag{7.34}$$

Practically, the simplex minimization algorithm of the `GNU` scientific library is used to minimize Eq. (7.32).

Once the parameters $\Delta z, \Delta\psi, \Delta\theta$ minimizing criterion (7.32) are found, they are converted in body local velocities:

$$v_z = \Delta z / t_{post} \tag{7.35}$$

$$\omega_x = \Delta\psi / t_{post} \tag{7.36}$$

$$\omega_y = \Delta\theta / t_{post} \tag{7.37}$$

where t_{post} is a time interval long enough to attain the desired posture. In the particular case of AMRU5, this parameter has been set to 2 s.

7.7.2 Correction of global velocities

The posture correction must be transformed into global velocities. Consider Fig. 7.26, where $\boldsymbol{n}$ is a vector normal to the body plane, and z_0 is the global Z axis. The user imposes the heading velocities (V_x, V_y, Ω_z), which are now represented by $\boldsymbol{v}_{traj}, \boldsymbol{\omega}_{traj}$. The posture algorithm gives $\boldsymbol{v}_{post} = (0,0,v_z)$ and $\boldsymbol{\omega}_{post} = (\omega_x, \omega_y, 0)$ as outputs. For the linear velocities, the simple sum $\boldsymbol{v}_{traj} + \boldsymbol{v}_{post}$ would yield an error on the (X,Y) components of the global velocities. That's why a correction is added to the body velocity:

$$\boldsymbol{v}_b = \boldsymbol{v}_{traj} + \boldsymbol{v}_{post} + \boldsymbol{v}_{corr}. \tag{7.38}$$

The contribution $\boldsymbol{v}_{post} + \boldsymbol{v}_{corr}$ must be parallel to the Z global axis:

$$\boldsymbol{v}_{post} + \boldsymbol{v}_{corr} = V\boldsymbol{u}_z \tag{7.39}$$

Projection on the normal axis $\boldsymbol{n}$ allows to find the velocity V:

$$\begin{aligned}(\boldsymbol{v}_{post} + \boldsymbol{v}_{corr}) \cdot \boldsymbol{n} &= V \cdot \boldsymbol{u}_z \cdot \boldsymbol{n} && (7.40)\\ \boldsymbol{v}_{post} \cdot \boldsymbol{n} &= V \cdot \boldsymbol{u}_z \cdot \boldsymbol{n} && (7.41)\\ V &= \frac{\boldsymbol{v}_{post} \cdot \boldsymbol{n}}{\boldsymbol{u}_z \cdot \boldsymbol{n}} && (7.42)\end{aligned}$$

Finally, the corrected velocity vector which will be given to the support phase algorithm is

$$\boldsymbol{v}_b = \boldsymbol{v}_{traj} + \frac{\boldsymbol{v}_{post} \cdot \boldsymbol{n}}{\boldsymbol{u}_z \cdot \boldsymbol{n}} \cdot \boldsymbol{u}_z. \tag{7.43}$$

A similar reasoning can be made for the rotational velocities. The point is that the Z component of the rotational velocity remains unchanged. So, the simple sum of $\boldsymbol{\omega}_{traj}$ and $\boldsymbol{\omega}_{post}$ must also be corrected by adding $\boldsymbol{\omega}_{corr}$, a vector parallel to the normal direction $\boldsymbol{n}$. We have:

$$\boldsymbol{\omega}_b = \boldsymbol{\omega}_{traj} + \boldsymbol{\omega}_{post} + \boldsymbol{\omega}_{corr}. \tag{7.44}$$

We want to keep the Z components unchanged:

$$\begin{aligned}\boldsymbol{\omega}_{traj} \cdot \boldsymbol{u}_z + \boldsymbol{\omega}_{post} \cdot \boldsymbol{u}_z + \boldsymbol{\omega}_{corr} \cdot \boldsymbol{u}_z &= \boldsymbol{\omega}_{traj} \cdot \boldsymbol{u}_z && (7.45)\\ \boldsymbol{\omega}_{post} \cdot \boldsymbol{u}_z + (\omega_{corr} \cdot \boldsymbol{n}) \cdot \boldsymbol{u}_z &= 0 && (7.46)\\ \omega_{corr} = -\frac{\boldsymbol{\omega}_{post} \cdot \boldsymbol{u}_z}{\boldsymbol{n} \cdot \boldsymbol{u}_z} & && (7.47)\end{aligned}$$

Finally, the corrected rotational velocity vector which will be given to the support phase algorithm is:

$$\boldsymbol{\omega}_b = \boldsymbol{\omega}_{traj} + \boldsymbol{\omega}_{post} - \frac{\boldsymbol{\omega}_{post} \cdot \boldsymbol{u}_z}{\boldsymbol{n} \cdot \boldsymbol{u}_z} \cdot \boldsymbol{n}. \tag{7.48}$$

7.7.3 Swing phase for free gait

As mentioned before, the swing problem is twofold: first thing is to select the swing candidate. Second thing is to drive the foot to a convenient position for the next support phase. Here we use two criteria which are quite conservative, because the stability of the robot is more concerned than the cruise velocity.

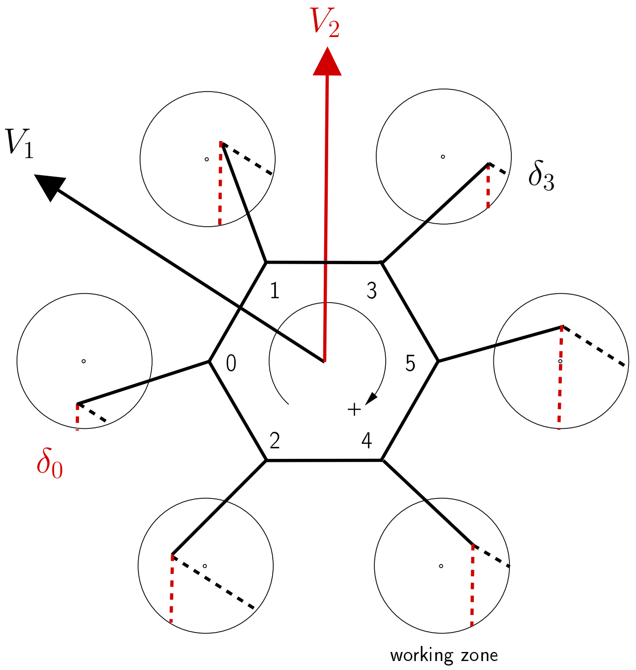
Selection of the swing candidate(s) Two rules have been used to select the best swing candidate. The first one is [PORT04]:

Rule 2

A leg can swing if and only if neighboring legs are in support

Table of Fig. 7.27 gives the numbering of the legs and their neighbors n_+, n_-.

The Rule 3 determines the leg to swing among the legs satisfying Rule 2:



leg	n_+	n_-
0	1	2
1	3	0
2	0	4
3	5	1
4	2	5
5	4	3

Figure 7.27: Illustration of rules 2 and 3

Rule 3

Lift up the leg if its kinematic margin is lower than the one of its two neighbors

where the kinematic margin [LEE88, HALM96] is the remaining distance to travel (in the direction of the robot motion) before reaching the boundary of the working zone. An illustration of the kinematic margin δ is found in Fig. 7.27 in the case of six legs on the ground. The interest of this criterion is that it confers to the vehicle omnidirectional ability, because δ is continuously computed and depends on the heading velocity of the robot body. Assume that initial velocity of the robot is V_1 in Fig. 7.27: then the candidate for swing is leg 3. If velocity changes abruptly to V_2, leg 0 becomes candidate.

The combination of these two criteria converges to a tripod gait on flat ground in most of the cases, and sometimes to a forward wave gait. But the convergence is not proved in this work: it would require extensive gait simulations with arbitrary positions of the six legs. However, in [PORT04], an extensive study proves that tripod convergence is reached with rule 2 and some priority assignments on neighboring legs.

Swing motion The AEP is selected to have a maximum kinematic margin, under the assumption that the body heading velocity is not changing. The takeoff and swing times are set constant, but not the landing because of uneven ground. Generation of feet motion for takeoff and landing is identical to the one of Section 7.5.2.

The swing itself is a bit different. Computation of the AEP landing point is made continuously with Eq. (7.24). The swing velocity of the foot is computed by Eq. (7.31), excepted that denominator is a constant time, fixed in our case at 3 s.

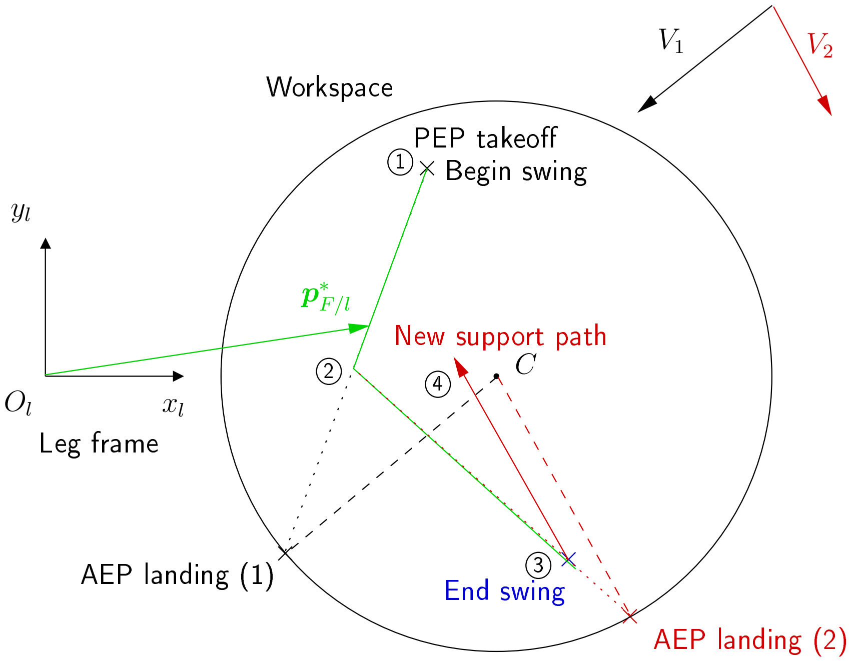


Figure 7.28: Research of target point in swing phase

The vertical position of the foot remains identical during the swing phase. This supposes that the takeoff has given a foot clearance high enough to avoid obstacles.

Figure 7.28 sketches a swing phase with changes in robot heading velocity. Beginning of the swing occurs in ①. The point AEP landing (1) is computed. Foot velocity is calculated with Eq.(7.31). At ②, the heading velocity changes from V_1 to V_2. The target point is now AEP landing (2). The swing ends at ③ because time for swing is elapsed. The leg goes down and a new support phase can begin, following the new support path ④. If velocity V_2 remains constant during several steps, the new stroke tends to align along the AEP landing-C direction so that the possible stroke is always maximal.

7.7.4 Free gait complete flowchart

Figure 7.29 depicts the flowchart of the free gait algorithm. The heading velocities (V_x, V_t, Ω_z) are read from the input stream. Then, if necessary (see **rule 1**), the posture is computed and gives the local velocities $(v_z, \omega_x, \omega_y)$ are merged together and corrected by the algorithm presented in Section 7.7.2, so that the global velocities $\boldsymbol{v}_{b/0}$ and $\boldsymbol{\omega}_{b/0}$ are available. The foot trajectory generation produce the next foot position $\boldsymbol{p}^*_{F/l}$ to reach, depending on the state of the leg. The inverse kinematic procedure provides the new joint references to the PI controllers. Then, **rule 2** and **rule 3** determines the swing candidate(s), and also manages the deadlocking condition of the robot, i.e. if one leg reaches the boundary of its workspace and can not be lifted up. In this case, the safest way to operate is to stop all the legs in support so that robot halts. Legs in takeoff, swing and landing are allowed to continue their motion. For landing and takeoff, only the vertical foot motion goes on. If the robot is not deadlocked, the loop begin again.

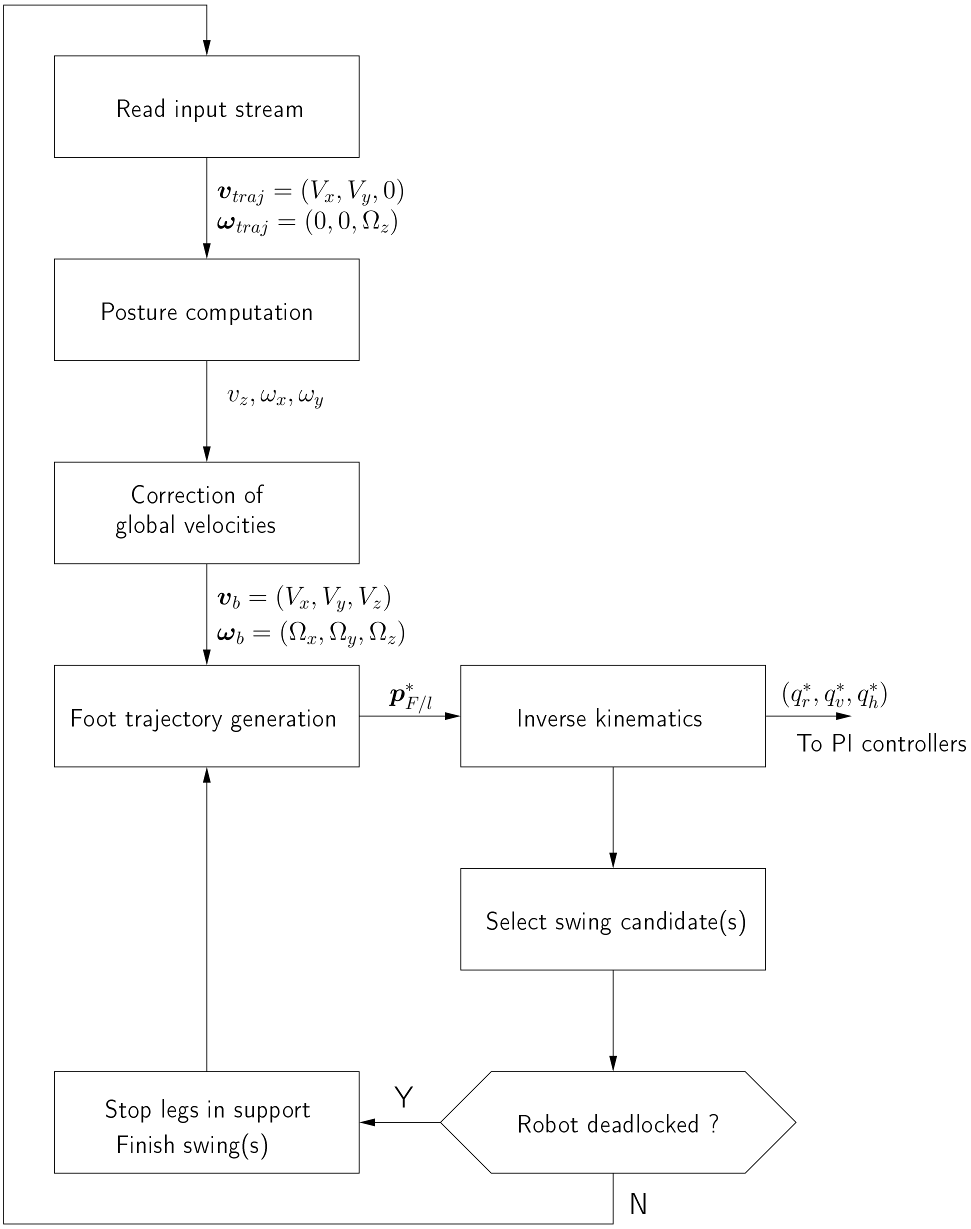

Read input stream
$\boldsymbol{v}_{traj} = (V_x, V_y, 0)$
$\boldsymbol{\omega}_{traj} = (0, 0, \Omega_z)$
Posture computation
v_z, ω_x, ω_y
Correction of global velocities
$\boldsymbol{v}_b = (V_x, V_y, V_z)$
$\boldsymbol{\omega}_b = (\Omega_x, \Omega_y, \Omega_z)$
Foot trajectory generation
$\boldsymbol{p}^*_{F/l}$
Inverse kinematics
(q^*_r, q^*_v, q^*_h)
To PI controllers
Select swing candidate(s)
Robot deadlocked ?
Y
N
Stop legs in support
Finish swing(s)

Figure 7.29: Free gait algorithm (starting with deadlock condition)

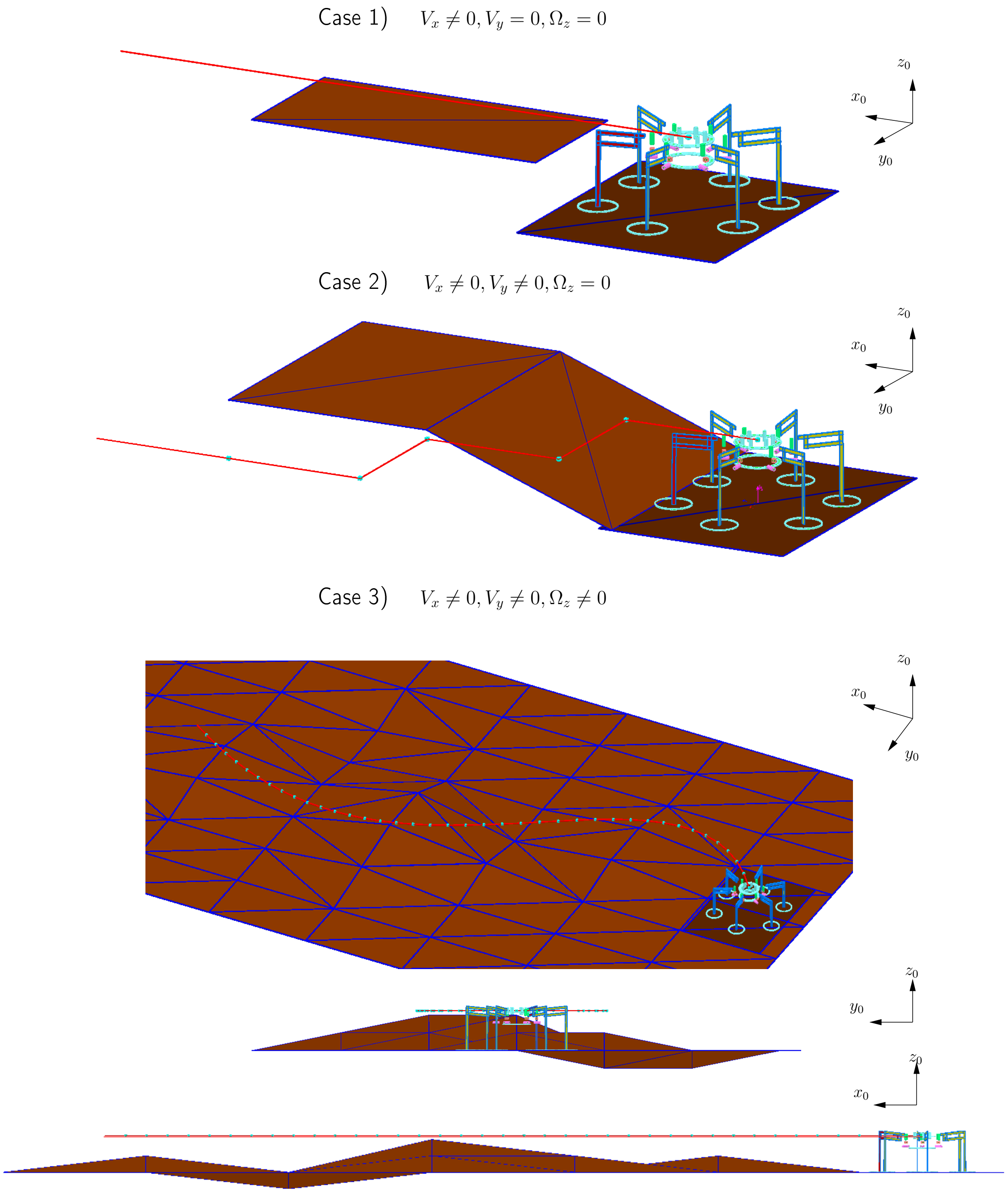
7.7.5 Results of simulations

As for Section 7.5.3, the dynamic model has been used to prove the correct working of the free gait algorithm. The friction coefficient at ground has been increased from 0.3 (Validation, see Chapter 6) to 0.7. In that way the robot is able to climb the different reliefs with a smaller slipping.

Three paths have been computed off-line to have an ASCII file as input stream for heading velocities. For each of them, a non flat terrain has been inserted in the model. The three cases considered are shown in Fig. 7.30:

- The robot follows a straight line during 400 s with a velocity V_x=0.01 m/s. There is a step as obstacle of 8 cm high on the right of the robot;
- The robot moves according to a succession of straight lines along X and Y axis. Each section of the trajectory lasts 100 s and is covered with V_x or V_y=0.01 m/s. The relief is a slope of 20 degrees;
- The robot follows a spline trajectory which lasts 1000 s. This spline has been built such that the tangential velocity of the robot never exceeds 0.02 m/s. The relief is a completely uneven ground.

Ground surfaces have been designed with `Blender` and exported under a *Polygon File Format* (or *the Stanford Triangle Format*) with `*.ply` file extension. The collision detection between the ground and the feet is managed by C++ routines which have been developed in the context of this thesis.



Case 1) $V_x \neq 0, V_y = 0, \Omega_z = 0$

Case 2) $V_x \neq 0, V_y \neq 0, \Omega_z = 0$

Case 3) $V_x \neq 0, V_y \neq 0, \Omega_z \neq 0$

Figure 7.30: Example of paths requiring a free gait

Case 1 The X motion of the robot remains at 1 cm/s along the X direction, during 500 s. Figure 7.31 shows the evolution of the robot when it climbs on the step and when it is completely on the step.

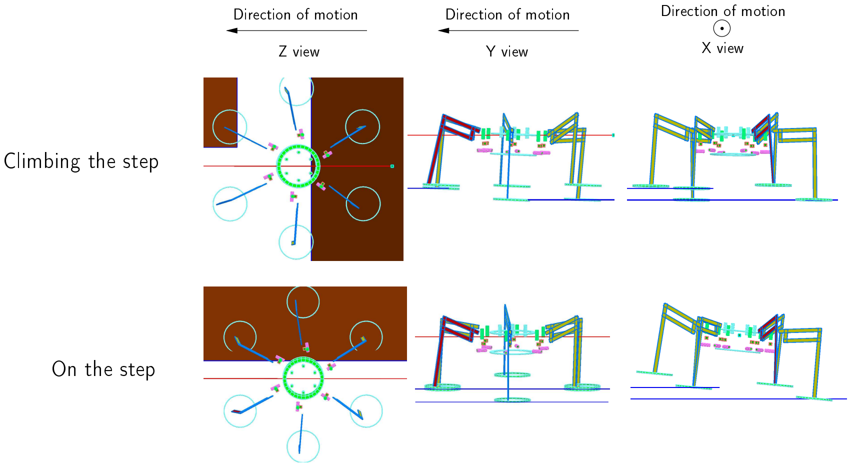


Figure 7.31: Screenshots of the animation (case 1)

Figure 7.32 shows the trajectory tracking errors. The obstacle does not disturb the tracking, which remains quite good even without navigation. The small deviation is due to the small slipping of the robot on the ground.

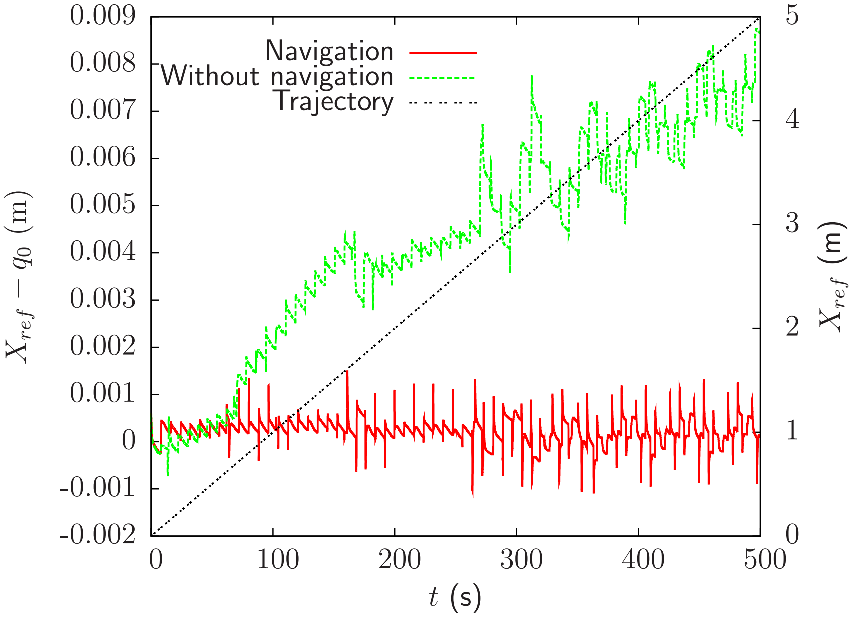


Figure 7.32: X reference and tracking error vs. time (case 1)

Figure 7.33 shows the evolution of the body altitude along the X position, for different robot clearances h_b. The clearance is defined as the height of the robot center of mass with respect to the estimated ground plane. The robot posture is just below the desired clearance due to the sinking of supporting legs in the ground (compare the robot heights with the dotted lines). The height varies logically when there are one, two or three steps on the step. The peaks are caused by the landing of the leg on high rigidity ground: this induces a light lifting of the main robot body, which is corrected immediately by the posture algorithm.

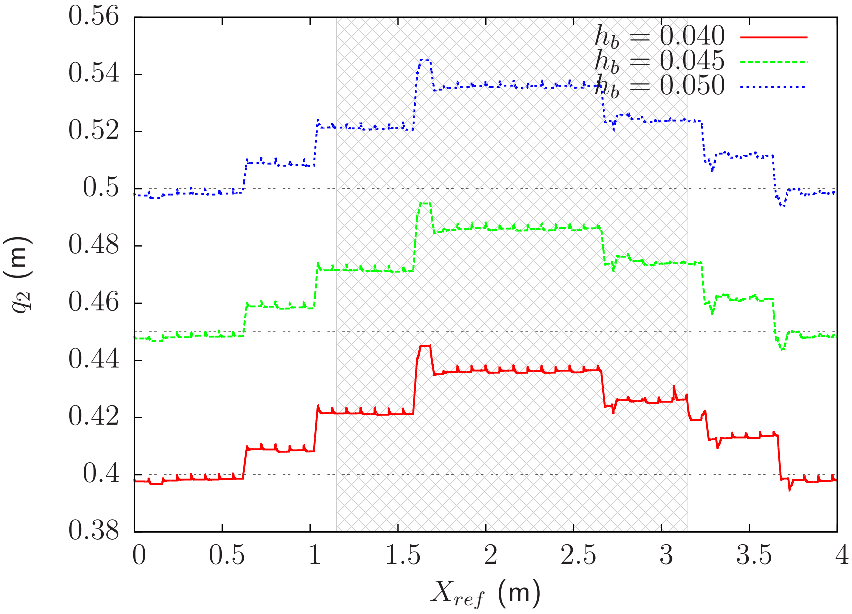


Figure 7.33: Body height vs. X coordinate, for different body clearances (case 1)

Figure 7.34 represents the roll and pitch of the robot for $h_b = 0.045$. The beginning and end of the trajectory are characterized by zero roll and pitch, because of flat ground. The pitch is negative during the rise of the obstacle, null on the step and positive at descent. The roll also goes down progressively during the obstacle climbing, stays constant during the crossing, and increases to reach a null value on flat ground. Because of the discrete nature of the posture correction, it is not surprising to have a discontinuous evolution for pitch and roll.

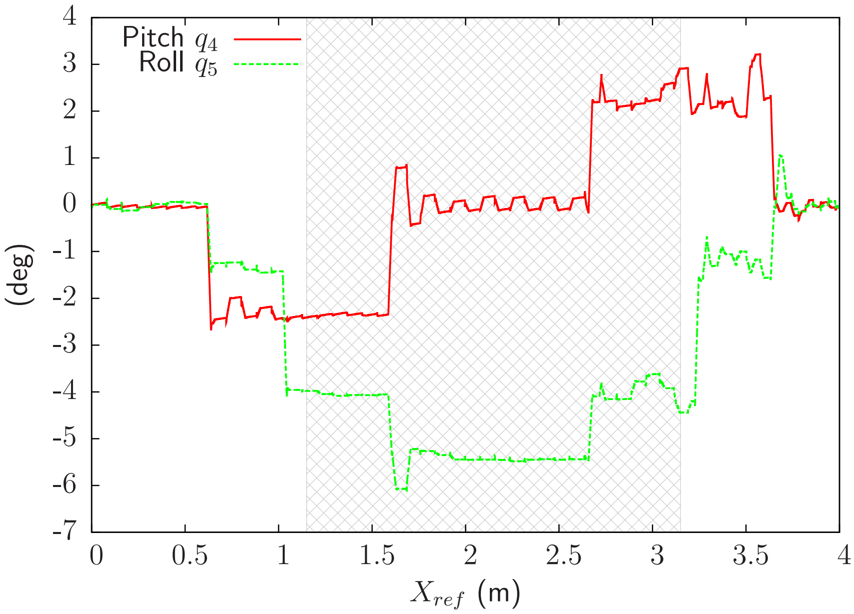


Figure 7.34: Evolution of the roll and pitch vs. X coordinate (case 1)

Case 2 Figure 7.35 shows the time history of X and Y references, and the magnitude of the XY tracking errors. The relief is correctly tracked, thanks to the posture correction.

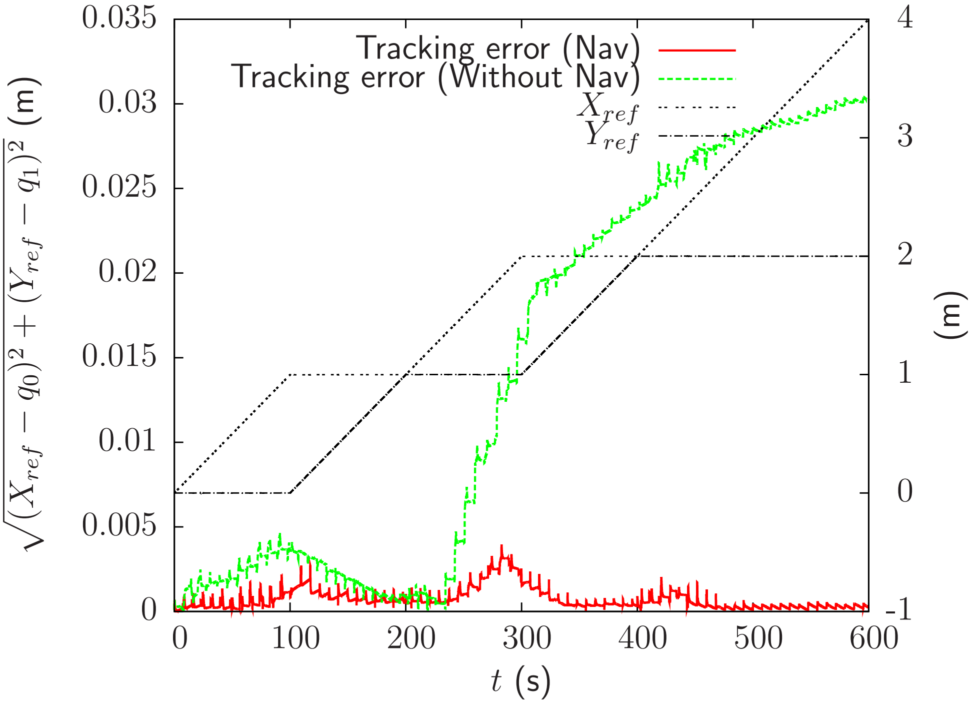


Figure 7.35: XY references and tracking error magnitudes vs. time (case 2)

Figure 7.36 represents the altitude evolution with the relief for different body clearances. The latter are conserved on the slope, according to an initial defined value. The velocity of the robot is never parallel to the slope because of the discrete nature of the correction. The initial clearance is correctly retrieved after the climbing of the slope.

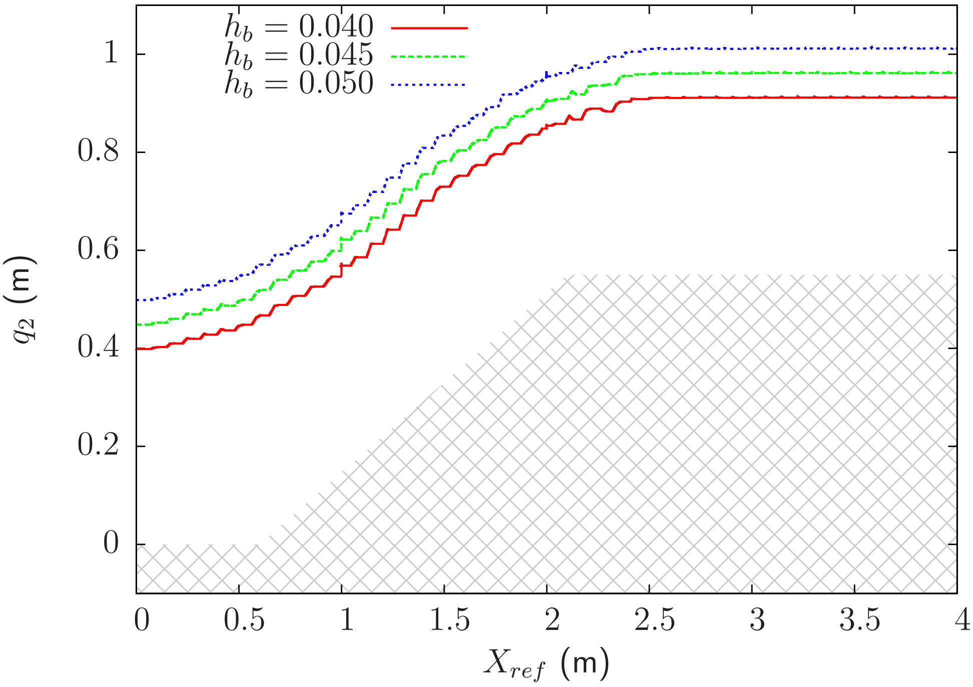


Figure 7.36: Evolution of the body height vs. X coordinate, for different body clearances (case 2)

Figure 7.37 presents roll and pitch for $h_b = 0.045$. The roll remains around zero, while the pitch adapts to 20 °. The pitch decreases afterwards because some legs attained the flat ground on the top of the slope, while others are still on it.

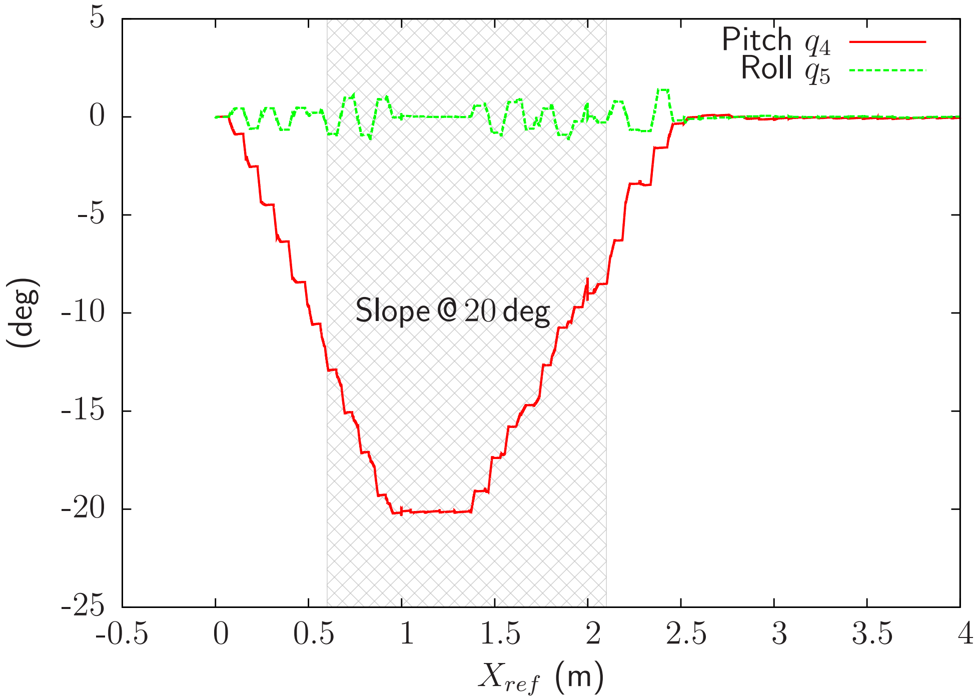


Figure 7.37: Evolution of the roll and pitch vs. X coordinate (case 2)

Case 3 The last case demonstrates that a curvilinear trajectory can be accurately tracked on real environment with the proposed algorithm. Figure 7.38 shows the X and Y references and the magnitudes of the X,Y tracking errors with and without navigation. Figure 7.39 illustrates the time history of the yaw reference, and the corresponding tracking error with/without navigation. From these two figures, we can say that even without navigation, the tracking is correct.

Figure 7.40 gives the robot altitude variation with respect to the X coordinate, and Fig. 7.41 depicts the roll and the pitch of the robot with respect to this same coordinate.

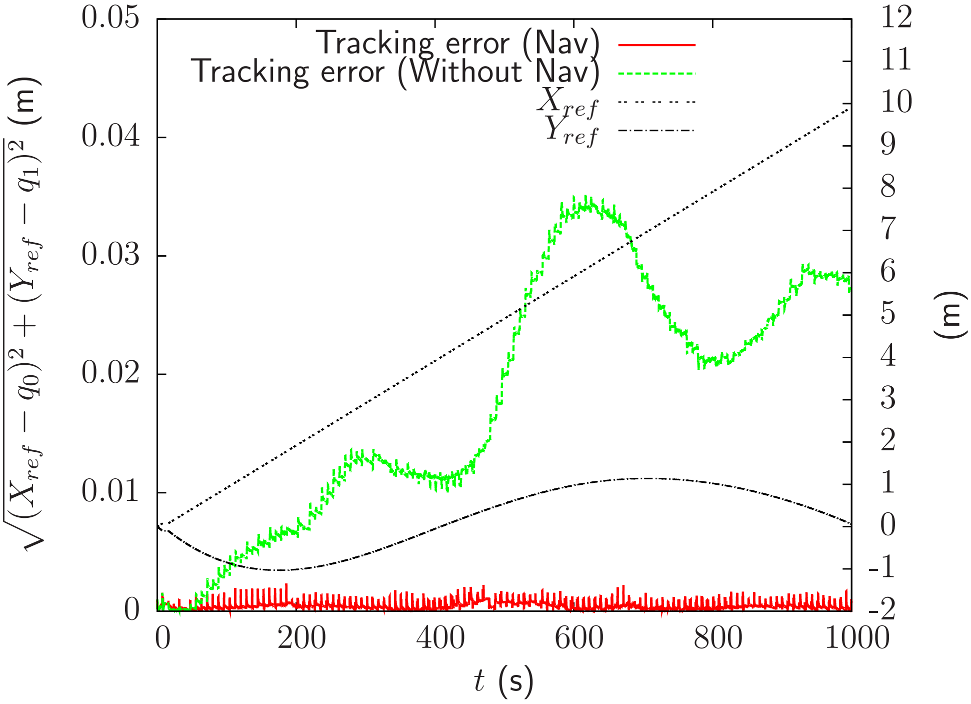


Figure 7.38: XY references and tracking error magnitudes vs. time (case 3)

A real example of walk on uneven ground exists on YouTube[6]. At the beginning of the video, the selection of the swing candidate takes some times to reach the tripod gait. Then, robot evolves on the uneven ground, and succeeds in passing the obstacle. The ground detection is made by the algorithm presented in Section 5.6. Note that this experiment has severely damaged the horizontal joints of the robot: for this reason we replace the ball bearings with linear bearings.

[6] http://www.youtube.com/watch?v=xYO59cPqId4

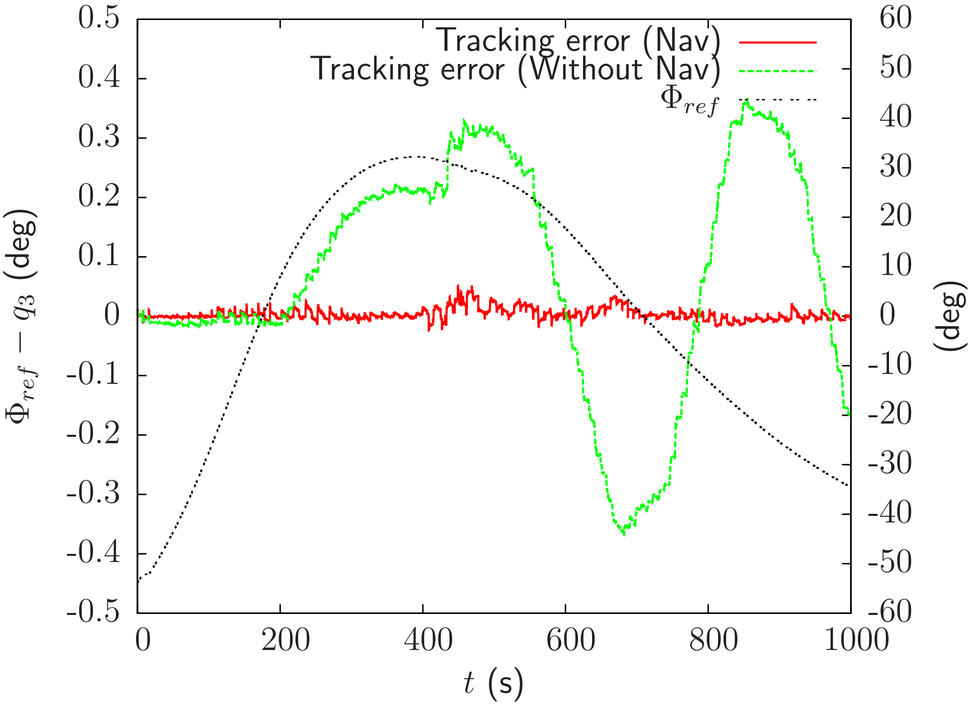


Figure 7.39: Yaw reference and tracking error vs. time (case 3)

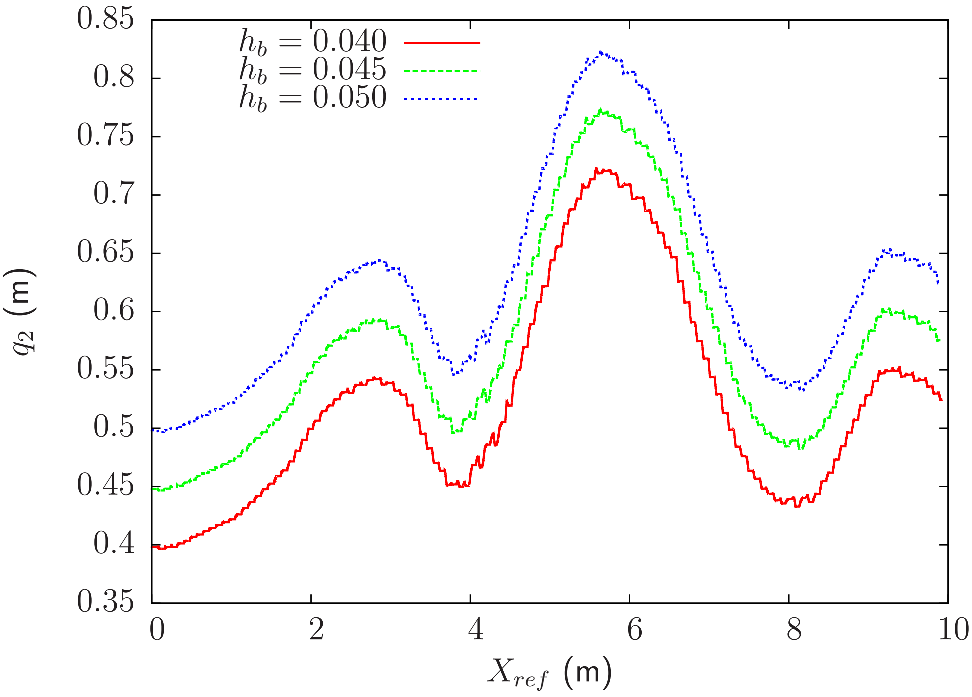


Figure 7.40: Evolution of the body height vs. X coordinate, for different body clearances (case 3)

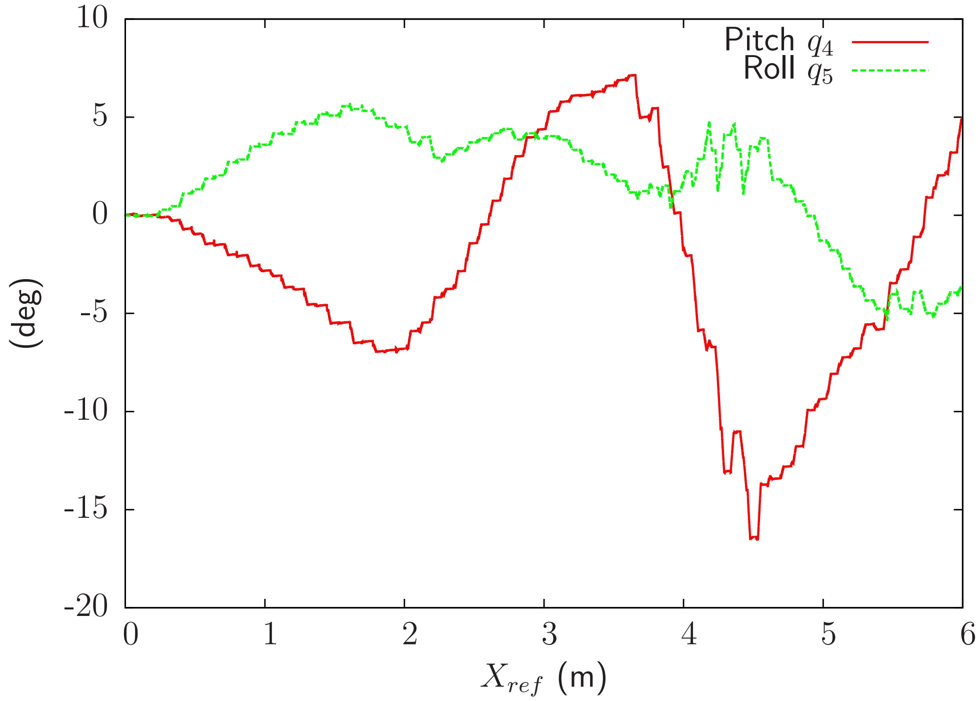


Figure 7.41: Evolution of the roll and pitch vs. X coordinate (case 3)

7.8 Summary

Three gait algorithms have been developed in this chapter, and successfully implemented on the real robot. The first one concerns classical periodic gaits. The second one develops the omnidirectional ability of the robot, in other words it allows the robot to follow any arbitrary trajectory, provided that it is on flat ground, and that velocities are not too important to keep the feet in their working zone. Stability issues have not been discussed, because the restricted workspace of the legs is such that a roll over of the robot can not occur. The third algorithm developed concerns free-gait on uneven ground. In addition to omnidirectional ability, the robot is able to walk on uneven ground by adapting to the relief, without force control as in most of walking robots. The ground profile tracking is ensured by a posture control, which tries to minimize the vertical distance of the feet in their actual position with a reference position.

The implementation on the real robot has shown that takeoff, landing and overlap phases are important to have a smooth trajectory tracking. Effectively, the deformation of the horizontal transmissions makes that the foot is not immediately lifted up as the vertical joint is moved. Consequently, the swing motion can not be executed immediately after the support phase, because it creates antagonist forces on the ground. In the same way, the swing-support transition can not be direct, because foot has already landed on the ground while it ends its swing. These antagonist forces are responsible for severe damages on the rotational joints, and bad trajectory tracking.

CHAPTER 8

Conclusions and prospects

Walking machines are complex mechanical systems which have been extensively studied for more than thirty years. Their principal interest is their ability to move smoothly on uneven ground, and to cross obstacles that wheeled robots are not able to pass.

The first part of this work was the development of a complete dynamic model of the robot, with the help of the open-source and flexible C++ library `EasyDyn`. The basis of the model is a multibody description of the mechanical parts of the robot, by using the minimal coordinates approach. On top of this, actuators, controllers and friction models are inserted to create a deep dynamic model, whose time integration is performed as a whole. The **first contribution** of this thesis concerns the inclusion of the LuGre model in the eighteen joints, with a special care for the irreversibility of the vertical joint. Effectively, the irreversibility behavior of this joint is experimentally stated, because robot becomes self-supporting when power is removed. We identified the parameters of the model and succeeded in reproducing this irreversibility, either in the case of a non-powered robot and in the case of a tripod walking on a flat ground.

The **second contribution** of the modeling concerns the validation of the dynamic model. Usually, they are validated by comparing forces measured at ground, or current in motors for one leg moving in the air (similarly to a robotic manipulator). Here we tried to identify friction as accurately as possible, to reproduce the time history of currents and voltages in the actuators, and derive the real power expenditure of the robot. Extensive simulations have been performed by a Monte Carlo process, to quantify the error on the specific resistance estimation of the robot. Finally, friction appears to be the most contributive effect in terms of power consumption as it represents more than 80 % of the energy losses in the system. This has previously been stated by other authors in literature.

The second part of the work was dedicated to the generation of gait pattern. Secondly, we tried to developed a gait algorithm which is reliable, and simple to implement. Gaits on flat ground have been developed first, with omnidirectional ability which allow to follow any arbitrary trajectory defined in the ground plane, and in a continuous way.

This represents a **third contribution** of the work, because most of the time the path followed by a walking machine is defined by successions of straight lines or arcs of circle. Dynamic simulation and implementation on the real robot have been made in parallel. This allows to formalize an improved foot cycle, with takeoff, landing and overlap phases. Without this improvement, implementation on the real robot is very inefficient and can even damage the rotational joints.

Finally, a free gait algorithm for walking on uneven ground has been developed. The feet in support phase move the robot according to a trajectory defined in the ground plane, and adapt the posture of the robot to follow the terrain. The leg to swing is chosen from two simple criteria:

- if both neighbors of a leg are on the ground, then the leg can be lifted up;
- the leg can swing if it has a smaller kinematic margin than its two neighbors.

These rules are known for a while in walking machines. The **fourth contribution** rather lies in the navigation approach: the inputs of the system are heading velocities, which are corrected after the posture calculation. In that way, it is shown that the robot adapts its local velocities to satisfy the velocity imposed by the user in terms of horizontal motion. Always in the context of free gait, an original ground detection algorithm based on current measurement has been developed and implemented successfully on the real machine. The accuracy of ground contact force is not very good, principally because of the numerous elements existing in the transmission of the vertical joints.

Finally, it is worth noticing that the tools for foot reference computation have been developed in a very general way, so that the adaptation on any other leg mechanism classically encountered in walking machines is direct.

In parallel to modeling and simulation, the complete control architecture of the robot has been developed. It is based on a master-slave relationship. The developed codes are identical between simulations and experiments, and have been developed in C/C++ .

This work has shown that the design of a static walker is not easy, especially with regards to the transmissions which are high geared, and consequently deteriorate the global efficiency of the vehicle. It could be interesting to compare the same robot with different actuators: for example, consider an AMRU5 bis with artificial muscles, as in biped machines. This kind of actuator is naturally compliant and could avoid antagonist motions of the feet on the ground; it also requires no gearing and avoids undesirable effects like stick-slip. They could improve greatly the specific resistance of the static walkers. A second prospect is to mix the friction modeling approach developed in this thesis with an efficiency approach. Efficiency of transmission naturally accounts for the load acting on it. In this way, the tedious identification of friction with different load conditions could be avoided. A third research prospect would be the integration of stability issues in the choice of the swinging leg in the free gait algorithm. Indeed, the speed of the robot is linked to the mobility of the leg, itself linked to the size of the workspace. If the workspace becomes large, the risk of instability is more important, and hence it should be taken into account in the free gait criteria. Tip-over stability should also be considered, because when the robot is walking on uneven ground the stability issues developed for periodic gaits are no longer sufficient.

APPENDIX A

Definition of elementary homogeneous transformation matrices

A pure 3D displacement can be expressed as:

$$\boldsymbol{T}^d(x, y, z) = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{A.1}$$

A rotation θ about x axis is put under the form:

$$\boldsymbol{T}^{R_x}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{A.2}$$

A rotation θ about y axis is put under the form:

$$\boldsymbol{T}^{R_y}(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{A.3}$$

A rotation θ about z axis is put under the form:

$$\boldsymbol{T}^{R_z}(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{A.4}$$

APPENDIX B

Detailed description of AMRU5

B.1 Mechanics

AMRU5 is a six-legged robot with an hexagonal architecture. Its outer diameter varies between $1.2m$ and $1.6m$, and its weight is about 34 kg. Each leg is pantograph-mechanism based, and has three degrees of freedom. Each of them is actuated by one and only one DC motor.

Fig. B.1 gives the configuration of one pantograph mechanism, as well as the relative position of the leg with respect to the main body. Center of mass of the main body and of the leg chassis are determined with the help of CAD tools (*Solid Edge*). Table B.1 summarizes the relevant geometrical data.

The pantograph mechanism induces a reduction ratio between the motion of the foot and the degrees of freedom considered in the work (which are q_0, q_1 and q_2 on Fig. B.1). Reduction ratios are :

$$z_{foot} = -6q_1 \tag{B.1}$$

$$x_{foot} = 7q_2 \tag{B.2}$$

Note : to obtain the real length of each pantograph body, h must be added (except for the foot where $h/2$ has to be considered)

AMRU5 is made with duralumin, an aluminium alloy which benefits from the aluminium lightness an higher rigidity. Only some critical parts, like the rotational transmission shaft, are made with steel. Inertia and mass characteristics are listed in Table B.2. Inertia have been computed with the help of `Solid Edge`. For each body, data are given in the center of mass frame.

e	0.004	thickness of the bodies composing the leg
e_b	0.005	thickness of the plate composing the main body
h	0.025	distance from the rotational joints to the extremity of bodies
H	0.144	gap between the two rings of the main body
R_{rot}	0.175	distance from middle of robot to rotation axis of the leg
L_0	0.28	interaxial length of the superior part of the pantograph
L_1	0.07	interaxial length of the left part of the pantograph
L_2	0.24	interaxial length of the inferior part of the pantograph
L_3	0.492	length between upper joint and foot
c	0.04	distance used to compute the dependent configuration parameters
$x_{G_{leg}}$	0.2055	position of center of mass of the leg
L_{G0}	0.155	position of the center of mass 0 wrt the bottom of the “chassis”
L_{x1}	0.05155	rotational axis of the leg and its chassis CM X distance
L_{y1}	-0.0168	rotational axis of the leg and its chassis CM Y distance
L_{z1}	-0.084	rotational axis of the leg and its chassis CM Z distance
L_{xc}	0.06045	crossing point and leg chassis CM X distance
L_{yc}	0.0168	crossing point and leg chassis CM Y distance
L_{zc}	0.032	crossing point and leg chassis CM Z distance
L_{x6}	0.104	main body CM and rotational actuator modeling X distance
L_{x7}	-0.06	crossing point and vertical actuator modeling X distance
L_{y7}	-0.04	crossing point and vertical actuator modeling Y distance
L_{z7}	0.035	crossing point and vertical actuator modeling Z distance
L_{x8}	-0.057	crossing point and horizontal actuator modeling X distance
L_{y8}	-0.04	crossing point and horizontal actuator modeling Y distance
L_{z8}	-0.081	crossing point and horizontal actuator modeling Z distance

Table B.1: Geometrical data [m]

ρ	2710 $[kg/m^3]$	specific mass of duralumin
ρ_{steel}	7800 $[kg/m^3]$	specific mass of steel
m_{leg}	4.372	total mass of a leg
m_0	8.215	mass of the main body with 6 motors and transmissions
m_1	2.895	mass of body supporting leg ($m_{tot} - m_2 - m_3 - m_4 - m_5 - 3m_{rot}$)
m_2	0.157	mass of the upper part of the pantograph
m_3	0.055	mass of the left part of the pantograph
m_4	0.143	mass of the lower part of the pantograph
m_5	0.564	mass of the right part of the pantograph (leg)
m_{rot}	0.186	rotor mass
m_{tot}	34	total mass of the robot
I^0_{xx}	0.1063	XX inertia moment of the main body
I^0_{yy}	0.1198	YY inertia moment of the main body
I^0_{zz}	0.0843	ZZ inertia moment of the main body
I^1_{xx}	0.016337	XX inertia moment of the body supporting the leg
I^1_{yy}	0.019609	YY inertia moment of the body supporting the leg
I^1_{zz}	0.006668	ZZ inertia moment of the body supporting the leg
I^2_{xx}	0.000019	XX inertia moment of the upper part of the pantograph
I^2_{yy}	0.001142	YY inertia moment of the upper part of the pantograph
I^2_{zz}	0.000047	ZZ inertia moment of the upper part of the pantograph
I^3_{xx}	0.00005	XX inertia moment of the left part of the pantograph
I^3_{yy}	0.000047	YY inertia moment of the left part of the pantograph
I^3_{zz}	0.00047	ZZ inertia moment of the left part of the pantograph
I^4_{xx}	0.000017	XX inertia moment of the lower part of the pantograph
I^4_{yy}	0.000825	YY inertia moment of the lower part of the pantograph
I^4_{zz}	0.000828	ZZ inertia moment of the lower part of the pantograph
I^5_{xx}	0.000074	XX inertia moment of the leg
I^5_{yy}	0.012151	YY inertia moment of the leg
I^5_{zz}	0.012167	ZZ inertia moment of the leg
I^{rot}_a	6.55e-6	axial inertia moment of the rotor
I^{rot}_e	1.91e-4	equatorial inertia moment of the rotor

Table B.2: Inertial data (mass in [kg] and inertia in [kg.m^2])

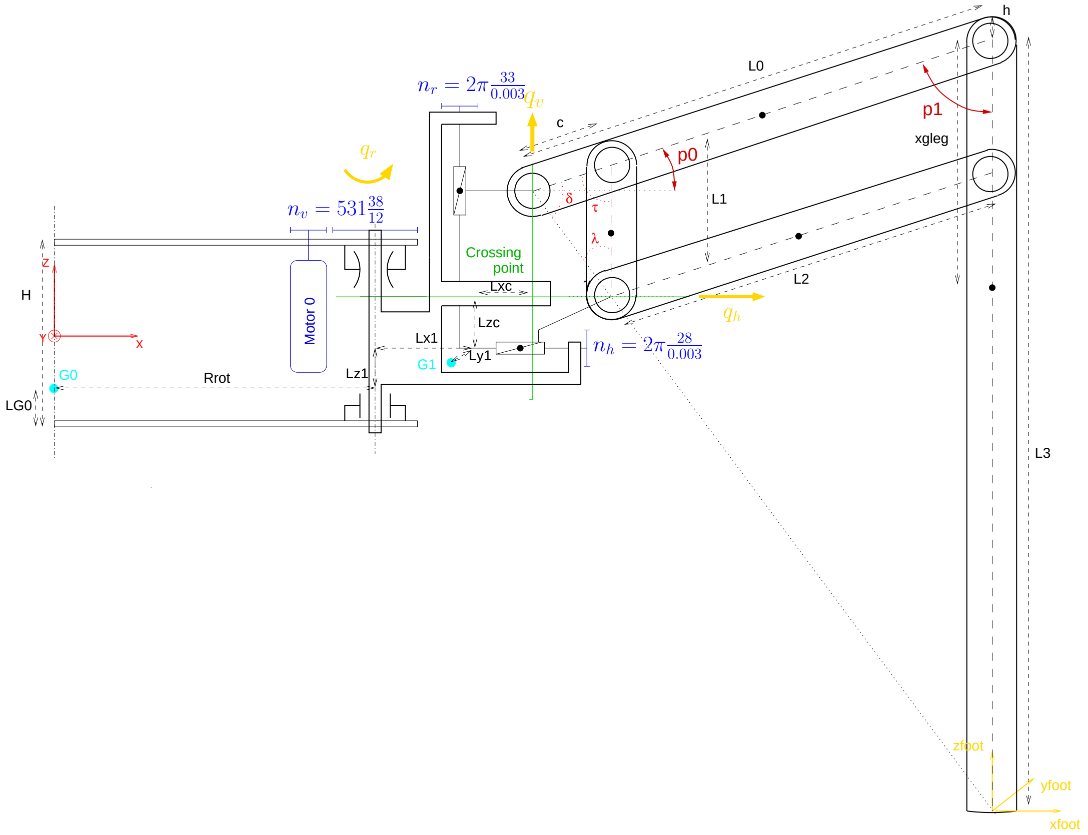


Figure B.1: Geometrical configuration of the leg

B.2 Electronics

Figure B.2 presents the slave board organization, and Fig. B.3 focuses on the motor current acquisition schematics.

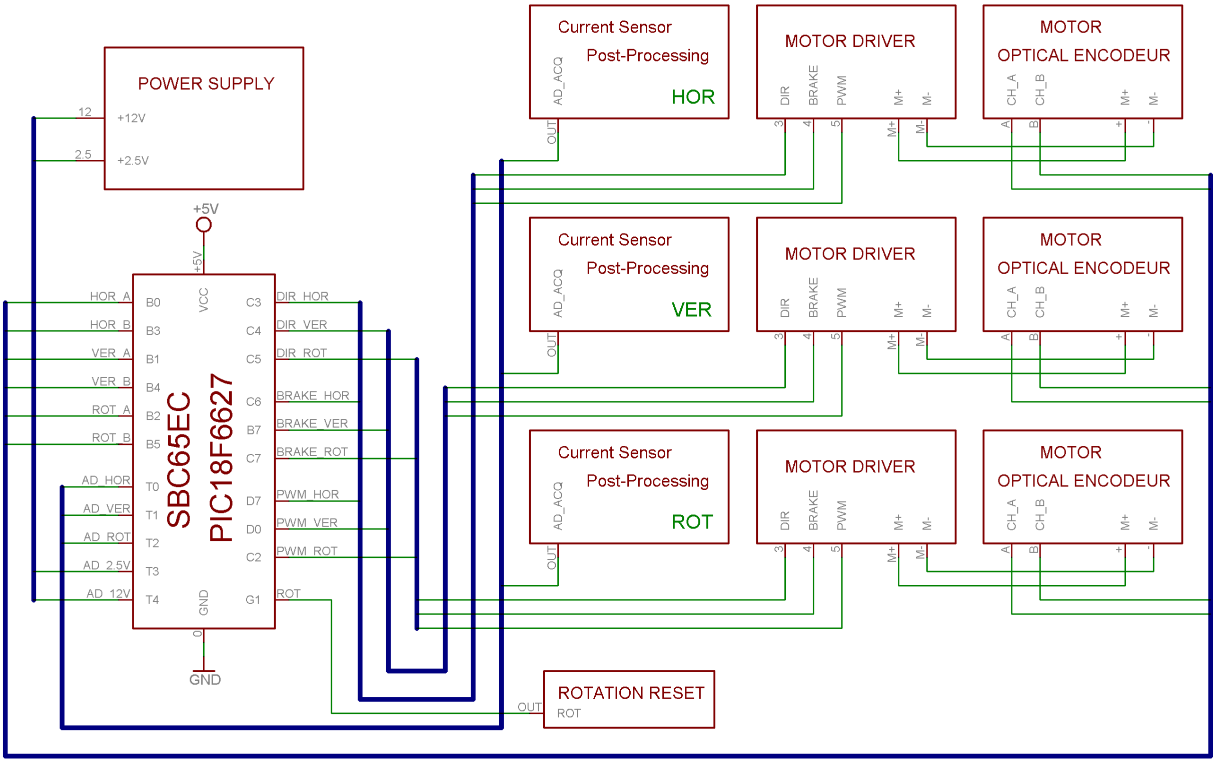


Figure B.2: Slave board organization

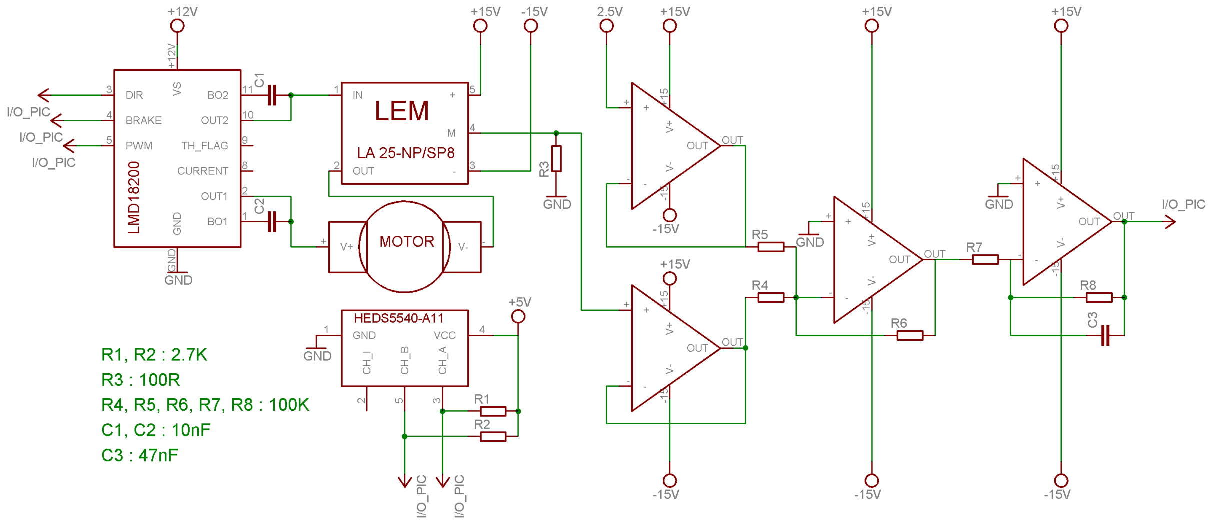
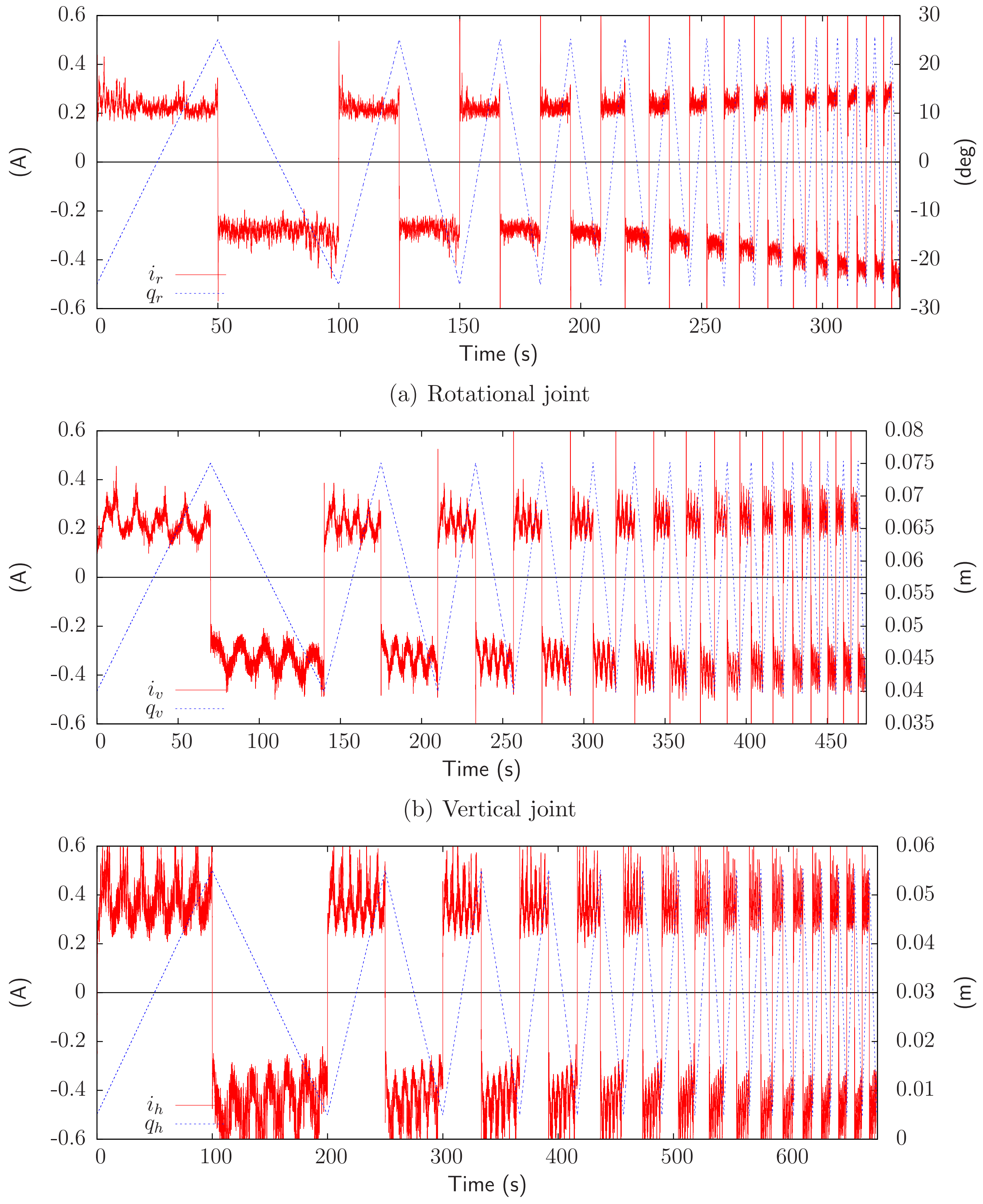


Figure B.3: LEM acquisition schematics

APPENDIX C

Friction measurements for unloaded legs

Current measurements are represented in Figures C.1 to C.6, for unloaded legs, and with constant velocity profiles imposed to each joint separately. The ordinate has been voluntary scaled from -0.6 to 0.6 A, so that the reader can easily compares the differences between actuators.

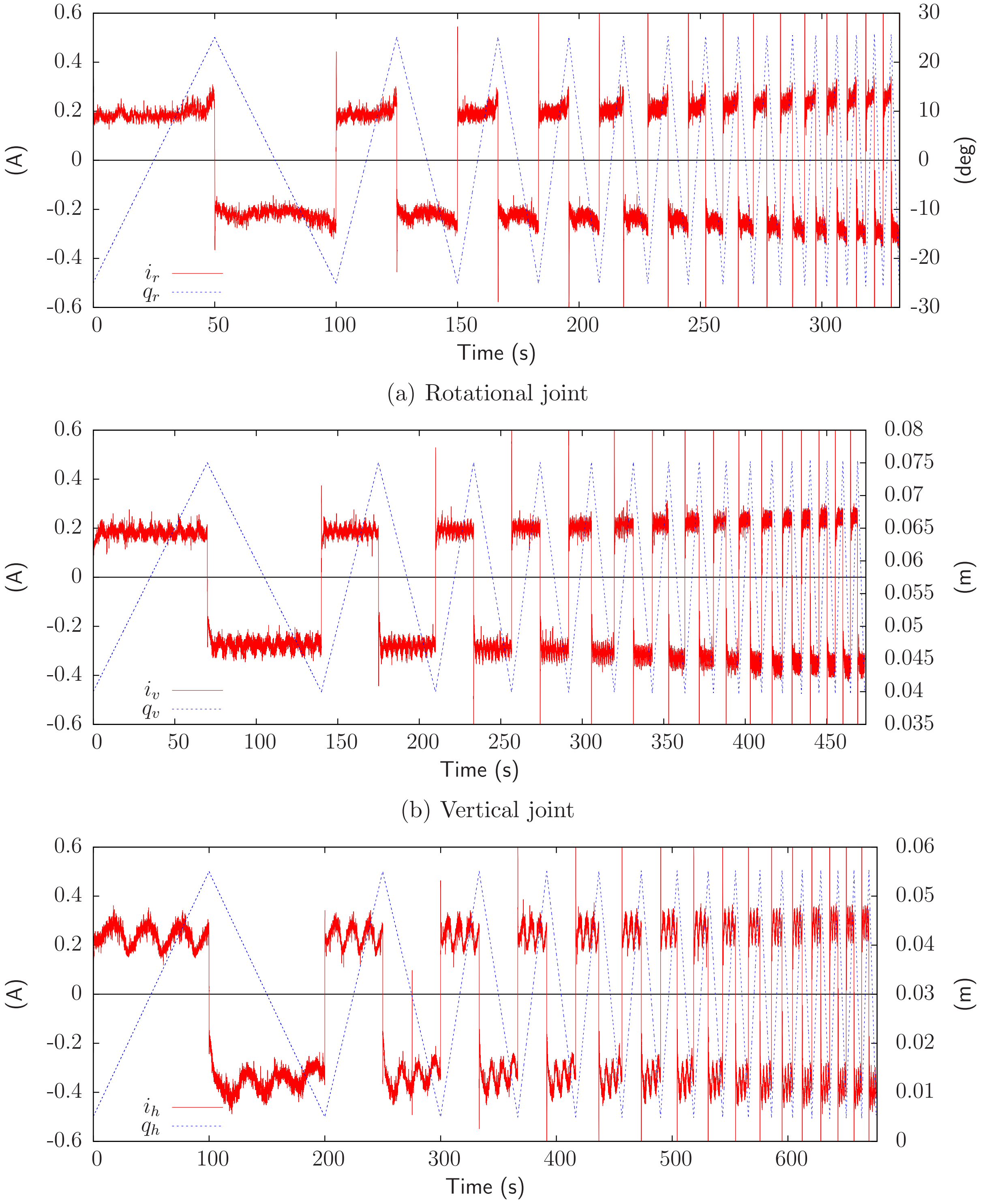


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.1: Current measurements for leg 0

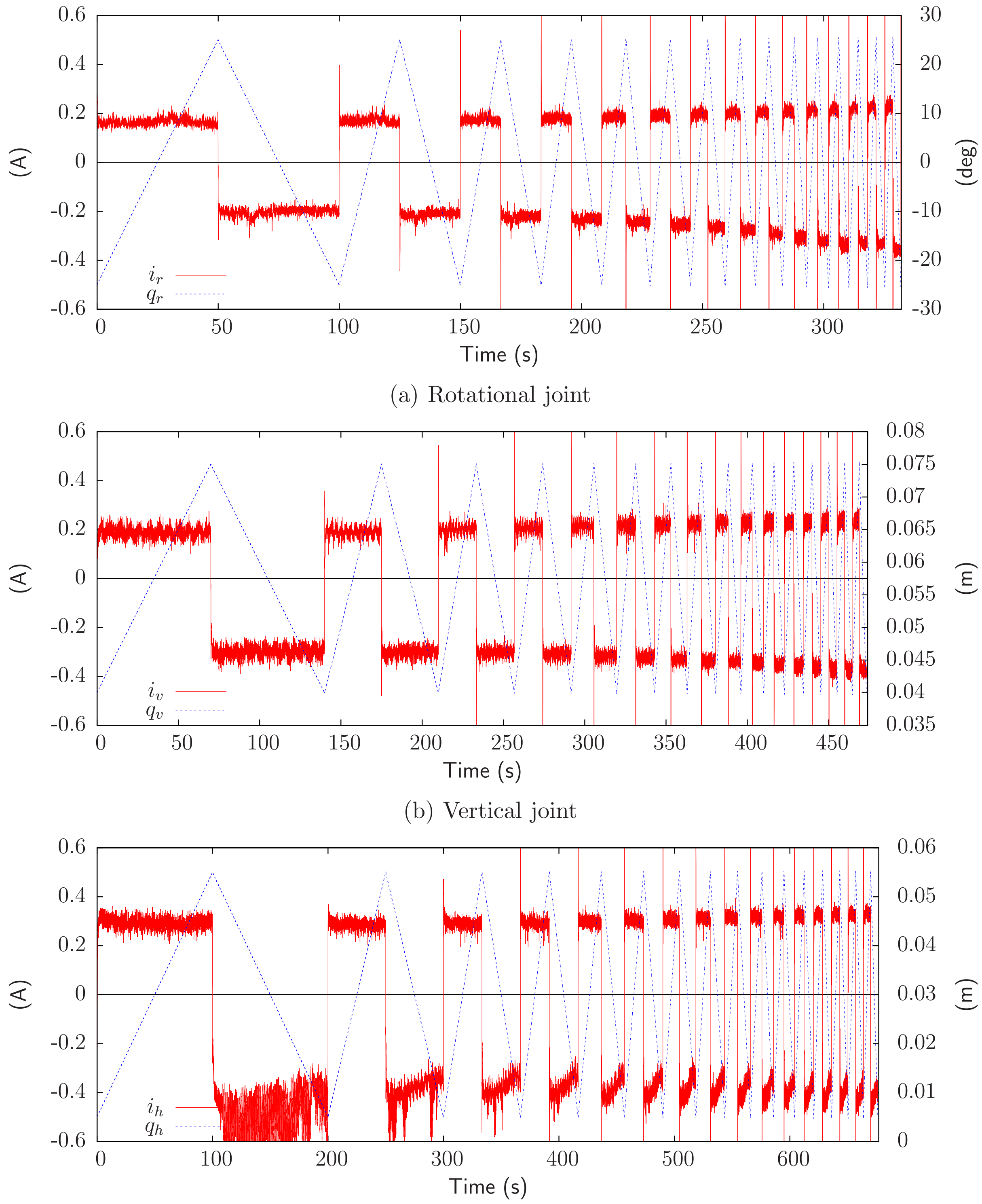


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.2: Current measurements for leg 1

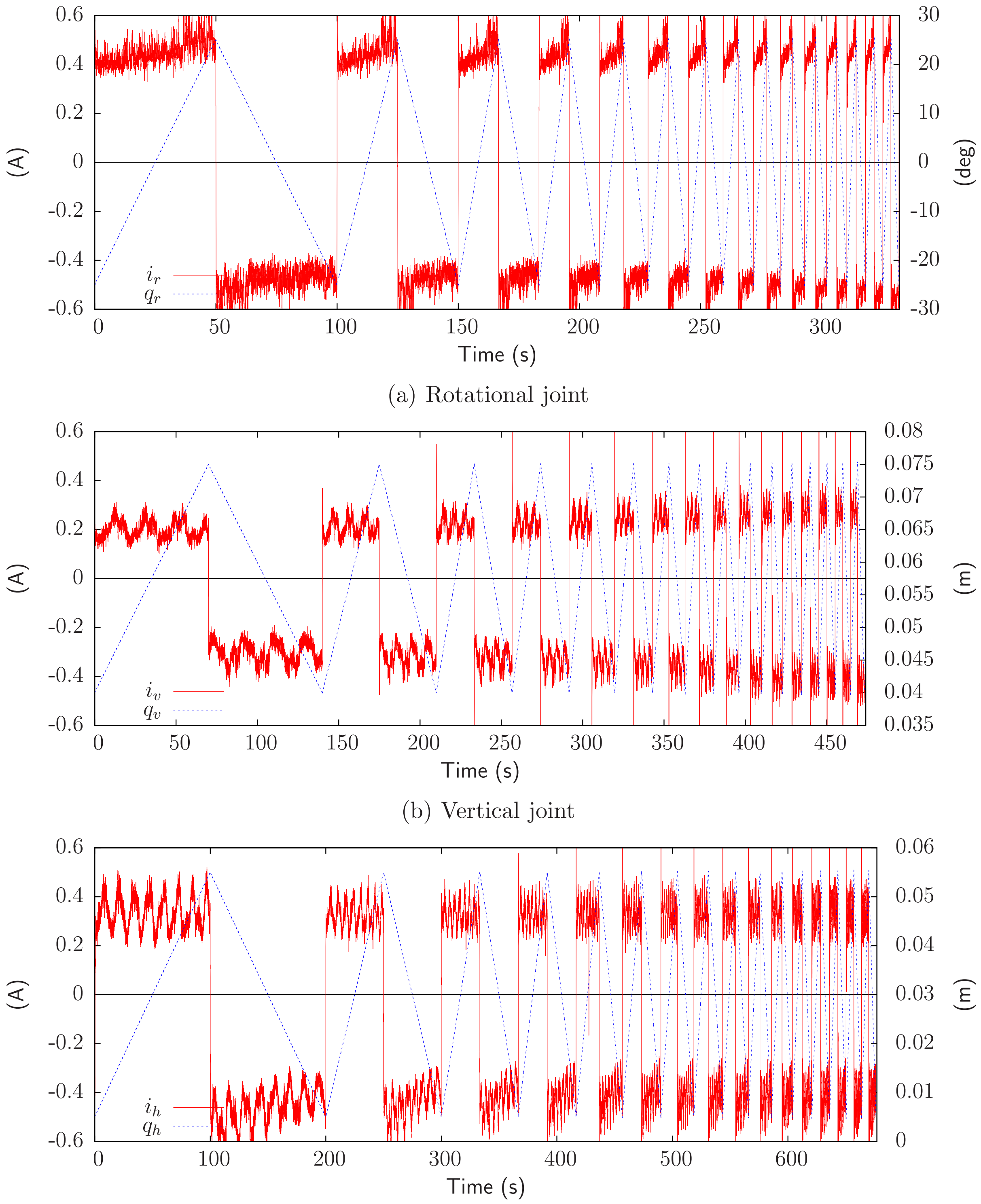


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.3: Current measurements for leg 2

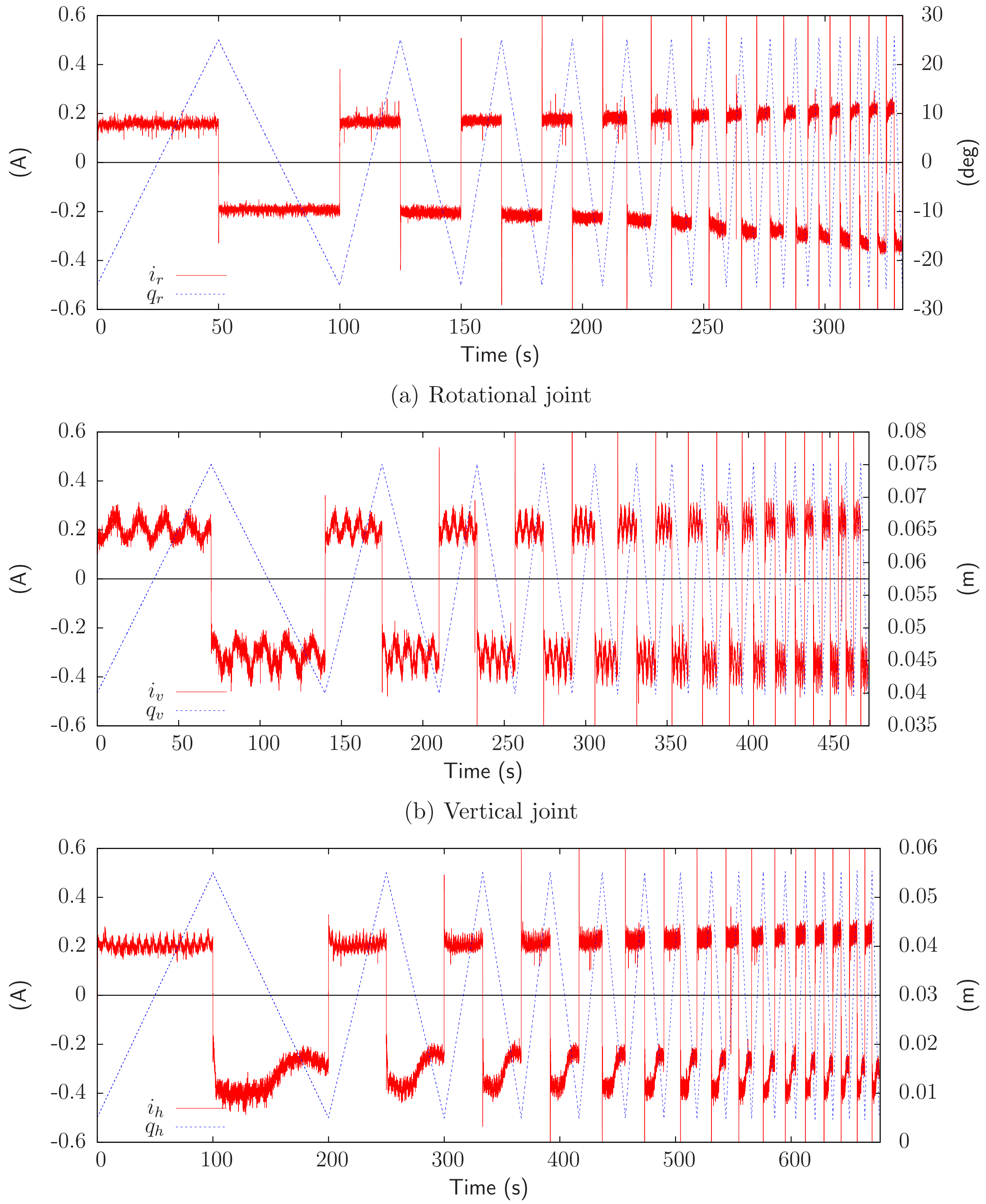


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.4: Current measurements for leg 3

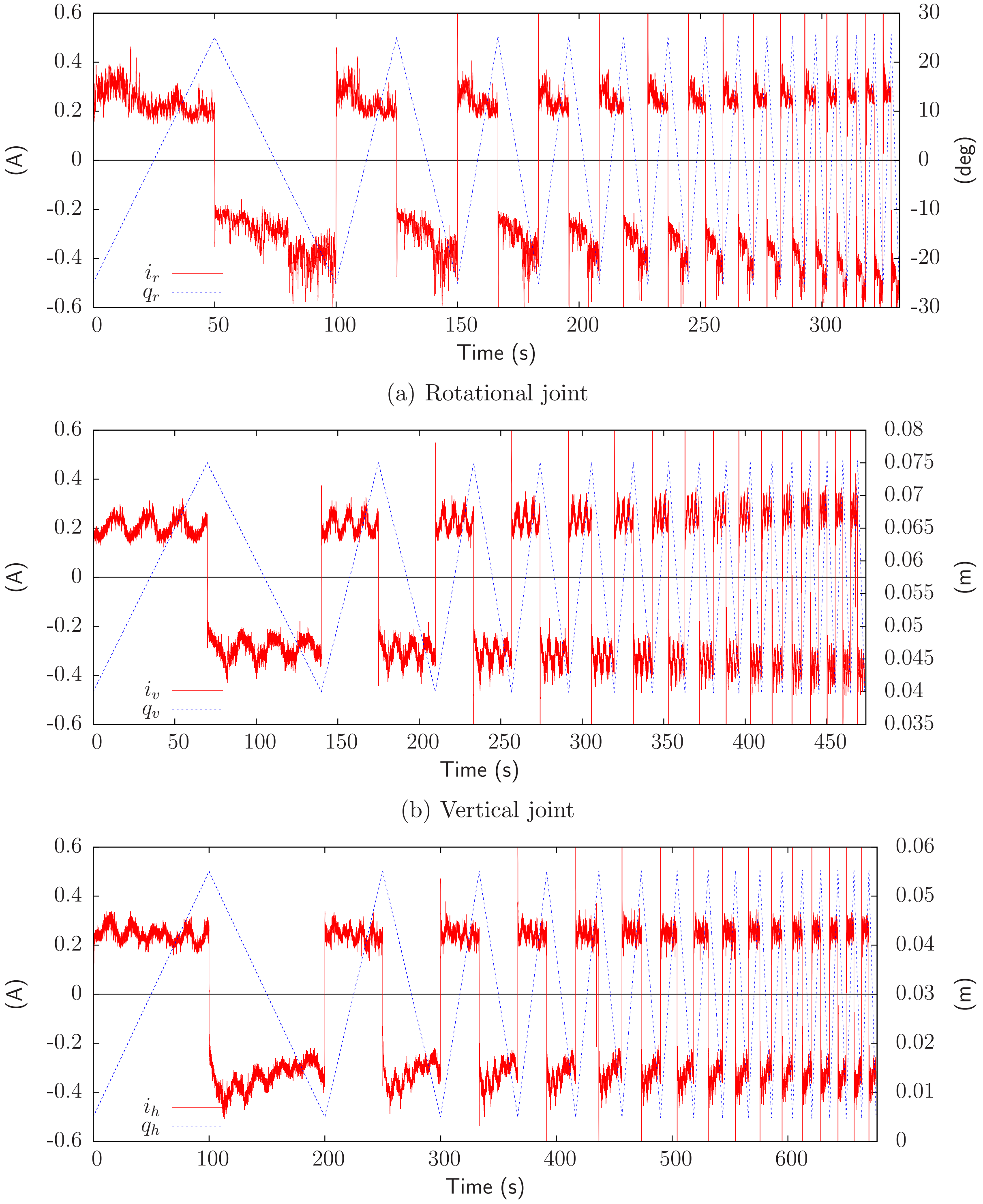


(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.5: Current measurements for leg 4



(a) Rotational joint

(b) Vertical joint

(c) Horizontal joint

Figure C.6: Current measurements for leg 5

Bibliography

[ACAR08] V. Acary and B. Brogliato. *Numerical methods for nonsmooth dynamical systems - Applications in Mechanics and Electronics.* Springer-Verlag, 2008.

[AL-BE04] F. Al-Bender, W. Symens, J. Swevers, and H. Van Brussel. Theoretical analysis of the dynamic behavior of hysteresis elements in mechanical systems. *International journal of non-linear mechanics*, 39:1721–1735, 2004.

[AL-BE05] F. Al-Bender, V. Lampaert, and J. Swevers. The generalized maxwell-slip model : A novel model for friction simulation and compensation. *IEEE Transactions on Automatic Control*, 50(11):1883–1887, November 2005.

[ALBI03a] J.C. Albiez, T. Luksch, K. Berns, and R. Dillmann. An activation-based behavior control architecture for walking machines. *The International Journal of Robotics Research*, 22:203–211, 2003.

[ALBI03b] J.C. Albiez, T. Luksch, K. Berns, and R. Dillmann. Reactive reflex-based control for a four-legged walking machine. *Robotics and Autonomous Systems*, 44:181–189, 2003.

[ALEX96] P. Alexandre and A. Preumont. On the gait control of a six-legged walking machine. *International Journal of Systems Science*, 27(8):713–721, 1996.

[ALEX97] P. Alexandre. *Le Contrôle Hiérarchisé d'un Robot Marcheur Hexapode.* PhD thesis, Université Libre de Bruxelles, 1997.

[ALTP99] F. Altpeter. *Friction Modeling, Identification and Compensation.* PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 1999.

[ANAN91] M. Anantharam and M. Hiller. Numerical simulation of mechanical systems using methods for differential-algebraic equations. *Int. J. Num. Meth. Eng.*, 21:1531–1542, 1991.

[ARMS91] B. Armstrong-Hélouvry. *Control of Machines with Friction.* Kluwer Academic, 1991.

[ARMS94] B. Armstrong-Hélouvry, P. Dupont, and C. Canudas de Wit. A survey of models, analysis tools and compensation methods for the control of machines with friction. *Automatica*, 30(7):1083–1138, 1994.

[ARMS94b] B. Armstrong-Helouvry. Frictional memory in servo control. In *Proceeding of the 1994 American Control Conference*, pages 1786 – 1790, Baltimore, MD, 1994.

[ASTR06] H.J. Aström and T. Hagglünd. *Advanced PID Control.* ISA, 2006.

[AUSL02] D.M. Auslander, J.R. Ridgely, and J.D. Ringgenberg. *Control Software for Mechanical Systems.* Prentice Hall, 2002.

[BARR98] J. Barreto, A. Trigo, P. Menezes, J. Dias, and A.T. de Almeida. Kinematic and dynamic modeling of a six legged robot. In *IEEE International Workshop on Advanced Motion Control*, Coimbra - Portugal, 1998.

[BERN99] K. Berns, W. Ilg, M. Deck, J. Albiez, and R. Dillman. Mechanical construction and computer architecture of the four-legged walking machine bisam. *IEEE/ASME Transactions on Mechatronics*, 4(1):32–38, March 1999.

[BONA05] B. Bona and M. Indri. Friction compensation in robotics : an overview. In *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, December 2005. IEEE.

[BONA06] B. Bona, M. Indri, and N. Smaldone. Friction identification and model-based digital control of a direct-drive manipulator. In L. Menini, L. Zaccarian, and C.T. Adballah, editors, *Current Trends in Nonlinear Systems and Control*, chapter 3, pages 231–251. Birkhäuser, 2007.

[BOST] http://www.bostondynamics.com.

[BOWL06] A. P. Bowling. Dynamic performance, mobility, and agility of multi-legged robots. *ASME Journal of Dynamic Systems, Measurement and Control*, 128(4):765–777, 2006.

[BREM] http://robotik.dfki-bremen.de/en.

[BROO91] R. A. Brooks. New approaches to robotics. *Science*, 253:1227–1232, 1991.

[BRUN96] F. Bruni, F. Caccavale, C. Natale, and L. Villani. Experiments of impedance control on an industrial robot manipulator with joint friction. In *IEEE Internation Conference on Control Applications*, 1996.

[CANU95] C. Canudas de Wit, H. Olsson, K. Åström, and P. Lischinsky. A new model for control systems with friction. *IEEE Transactions on Automatic Control*, 40:419–425, 1995.

[CAPP07] Maki K. Habib, editor. *Bioinspiration and Robotics : Walking and Climbing Robots*. I-Tech, 2007.

[CELA98] E. Celaya and J.M. Porta. A control structure for the locomotion of a legged robot on difficult terrain. *IEEE Robotics and Automation Magazine, Special Issue on Walking Robots*, 5(2):43–51, 1998.

[CMU] http://www.ri.cmu.edu.

[COBA08] J.A. Cobano, J. Estremera, and P. Gonzalez de Santos. Location of legged robots in outdoor environments. *Robotics and Autonomous Systems*, 56:751–761, 2008.

[COBA09] J.A. Cobano, J. Estremera, and P. Gonzales de Santos. Omnidirectional hexapod walking and efficient gaits using restrictedness. *Accurate tracking of legged robots on natural terrain*, 28:231–244, 2009.

[COLL94] J. J. Collins and S. A. Richmond. Hard-wired contral pattern generators for quadrupedal locomotion. *Biol. Cybern.*, 71:375–385, 1994.

[CORK96] P.I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, mar 1996.

[CRUS06] H. Cruse. *Adaptative Motion of Animals and Machines*, chapter Control of Hexapod Walking in Biological Systems, pages 17–29. Springer Tokyo, 2006.

[CRUS09] H. Cruse, V. Dürr, M. Schilling, and J. Schmitz. Principles of insect locomotion. In Rüdiger Dillmann, David Vernon, Yoshihiko Nakamura, Stefan Schaal, Paolo Arena, and Luca Patanè, editors, *Spatial Temporal Patterns for Action-Oriented Perception in Roving Robots*, volume 1 of *Cognitive Systems Monographs*, pages 43–96. Springer Berlin Heidelberg, 2009.

[CRUS95] Holk Cruse, Christian Bartling, and Thomas Kindermann. High-pass filtered positive feedback for decentralized control of cooperation. In F. Moran, A. Moreno, J. Merelo, and P. Chacon, editors, *Advances in Artificial Life*, volume 929 of *Lecture Notes in Computer Science*, pages 668–678. Springer Berlin / Heidelberg, 1995.

[CRWU] http://biorobots.cwru.edu.

[CSIC] http://www.iai.csic.es.

[CYBE] http://cyberneticzoo.com.

[DEBE05] P. Debenest, E. F. Fukushima, Y. Tojo, and S. Hirose. A new approach to humanitarian demining - part 1: Mobile platform or operation on unstructured terrain. *Autonomous Robots*, 18:303–321, 2005.

[DELC04] F. Delcomyn. Insect walking and robotics. *Annu. Rev. Entomol.*, 49:51–70, 2004.

[DEVJ83] E.A. Devjanin, V.S. Grufinkel, E.V. Gurfinkel, V.A. Kartashev, A.V. Lensky, A. Yu. Schneider, and L.G. Shtilman. The six-legged walking robot capable of terrain adaptation. *Mechanism and Machine Theory*, 18(4):257–260, 1983.

[DUPO02] P. Dupont, V. Hayward, B. Armstrong, and F. Altpeter. Single state elastoplastic friction models. *IEEE Transactions on Automatic Control*, 47(5):787–792, May 2002.

[ERDE08] M.S. Erden and K. Leblebicioğlu. Free gait generation with reinforcement learning for a six-legged robot. *Robotics and Autonomous Systems*, 56:199–212, 2008.

[ESPE96] K.S. Espenschied, R.D. Quinn, R.D. Beer, and H.J. Chiel. Biologically based distributed control and local reflexes improve rough terrain locomotion in a hexapod robot. *Robotics and Autonomous Systems*, 18:59–64, 1996.

[FERR03] G. Ferretti, G. Magnani, G. Martucci, P. Rocco, and V. Stampacchia. Friction model validation in sliding and presliding regimes with high resolution encoders. In B. Siciliano and B. Dario, editors, *Experimental Robotics*, volume 8 of *Springer Tracts in Advanced Robotics*, pages 328–337. Springer, 2003.

[FERR95] C. Ferrel. A comparison of three insect-inspired locomotion controllers. *Robotics and Autonomous Systems*, 16:135–159, 1995.

[FIEL04] M.R. Fielding and G.R. Dunlop. Omnidirectional hexapod walking and efficient gaits using restrictedness. *The International Journal of Robotics Research*, 23(10-11):1105–1110, 2004.

[FIGL07] G. Figliolini and P. Rea. *Climbing and Walking Robots Towards New Applications*, chapter Mechanics and Simulation of Six-Legged Walking Robots, pages 546–568. 2007.

[FREE91] P.S. Freeman and D.E. Orin. Efficient dynamic simulation of a quadruped using a decoupled tree-structure approach. *The International Journal of Robotics Research*, 6:619–627, 1991.

[FZI] http://en.wikipedia.org/wiki/LAURON.

[GALV03] J. A. Galvez, J. Estremera, and P. Gonzalez de Santos. A new legged-robot configuration for research in force distribution. *Mechatronics*, 13:907–932, 2003.

[GARC00] E. Garcia, J.A. Galvez, and P. Gonzalez de Santos. A mathematical model for real-time control of the silo4 leg. In *Proceedings of the 3rd International Conference CLAWAR*, Madrid - Spain 2000.

[GARC02] E. Garcia, P. Gonzalez de Santos, and C. Canudas de Wit. Velocity dependence in the cyclic friction arising with gears. *The International Journal of Robotics Research*, 21(9):761–771, 2002.

[GARC03] E. Garcia, J.A. Galvez, and P. Gonzalez de Santos. On finding the relevant dynamics for model based controlling walking robots. *Journal of Intelligent and Robotic Systems*, 37(4):375–398, 2003.

[GASS01] B. Gaßmann, K.-U. Scholl, and K. Berns. Locomotion of LAURON III in rough terrain. In *IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, volume 2, pages 959–964, Como, Italy, 2001. IEEE/ASME.

[GONZ05] P. Gonzalez de Santos, E. Garcia, J. Estremera, and M.A. Armada. DYLEMA : Using walking robots for landmine detection and location. *International Journal of Systems Science*, 36(9), 2005.

[GONZ06] P. Gonzalez de Santos, E. Garcia, and J. Estremera. *Quadrupedal Locomotion: An introduction to the Control of Four-Legged Robots.* Springer, 2006.

[GONZ07] P. Gonzalez de Santos, E. Garcia, and J. Estremera. Improving walking-robot performances by optimizing leg distribution. *Autonomous Robots*, 23(4):247–258, 2007.

[GONZ09] P.Gonzalez de Santos, E. Garcia, R. Ponticelli, and M.A. Armada. Minimizing energy consumption in hexapod robots. *Advanced Robotics*, 23:681–704, 2009.

[GORI90] D. M. Gorinevsky and A. Yu. Schneider. Force control in locomotion of legged vehicles over rogid and soft surfaces. *The International Journal of Robotics Research*, 9(2), 1990. que photocopie dispo.

[GREG97] P. Gregorio, M. Ahmadi, and M. Buehler. Design, control, and energetics of an electrically actuated legged robot. *IEEE Trans. Syst. Man, Cybern*, 27(4):626–634, 1997.

[GROT02] M. Grotjahn and B. Heimann. Model-based feedforward control in industrial robotics. *The International Journal of Robotics Research*, 21(45), 2002.

[GROT04] M. Grotjahn, B. Heimann, and H. Abdellatif. Identification of friction and rigid-body dynamics of parallel kinematic structures for model-based control. *Multibody System Dynamics*, 11:273–294, 2004.

[GUAR03] T.A. Guardabrazo Pedroche, M.A. Jimenez Ruiz, and P. Gonzalez de Santos. A detailed power consumption model for walking robots. In *Proceedings of the 5th International Conference CLAWAR*, 2003.

[HABU04] J.C. Habumuremyi. *Adaptative Neuro-Fuzzy Control for a Walking Robot.* PhD thesis, Vrije Universiteit Brussel, 2004.

[HAIR91] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II.* Springer-Verlag, 1991.

[HALM94] A. Halme, K. Hartikkainen, and K. Kärkkäinen. Terrain adaptive motion and free gait of a six-legged walking machine. *Control Eng. Practice*, 2(2):273–279, 1994.

[HALM96] S. Salmi and A. Halme. Implementing and testing a reasoning-based free gait algorithm in the six-legged walking machine "MECANT". *Control Eng. Practice*, 4(4):487–492, 1996.

[HIRO84] S. Hirose. A study of design and control of a quadruped walking vehicle. *The International Journal of Robotics Research*, 3(2):113–133, 1984.

[HUAN02] Q.-J. Huang and K. Nonami. Humanitarian mine detecting six-legged walking robot and hybrid neuro walking control with position/force control. *Mechatronics*, 13:773–790, 2003.

[IFF] http://www.uni-magdeburg.de.

[INAG03] S. Inagaki, H. Yuasa, and T. Arai. Cpg model for autonomous decentralized multi-legged robot system – generation and transition of oscillation patterns and dynacmis of oscillators. *Robotics and Autonomous Systems*, 44:171–179, 2003.

[JIME96] M.A. Jimenez and P. Gonzalez de Santos. Attitude and position control method for realistic legged vehicles. *Robotics and Autonomous Systems*, 18:345–354, 1996.

[KLEI80] C. A. Klein and R. L. Briggs. Use of active complance in the control of legged vehicles. *IEEE Trans. Syst. Man, Cybern*, 10:393–400, 1980.

[KRAU97] P.R. Kraus, A. Fredriksson, and V. Kumar. Modeling of frictional contact models for dynamic simulation. In *IEEE/RSJ Internation Conference on Intelligent Robots and Systems*, 1997.

[KURO08] J. F. Kurose and K. W. Ross. *Computer Networking: International Version: A Top-Down Approach.* Pearson Education, 2008.

[LAMP02] V. Lampaert, J. Swevers, and F. Al-Bender. Modification of the leuven integrated friction model structure. *IEEE Transactions on Automatic Control*, 47(4), April 2002.

[LAMP03] V. Lampaert, F. Al-Bender, and J. Swevers. A generalized maxwell-slip friction model appropriate for control purposes. In *PhysCon 2003*, St. Petersburg, Russia, 2003.

[LAMP04] V. Lampaert, J. Swevers, and F. Al-Bender. Comparison of model and non-model based friction compensation techniques in the neighbourhood of pre-sliding friction. In *Proceeding of the 2004 American Control Conference*, pages 1121–1126, Boston, Massachussetts, 2004.

[LEE88] W. Lee and D. E. Orin. Omnidirectional supervisory control of a multilegged vehicle using periodic gaits. *IEEE Journal of Robotics and Automation*, 4(6):79–91, 1988.

[MAHA09] A. Mahapatra and S.S. Roy. Computer aided dynamic simulation of six-legged robot. *Int. J. of Recent Trends in Engineering*, 2(2), 2009.

[MANK92] D. J. Manko. *A General Model of Legged Locomotion on Natural Terrain.* Kluwer Academic Publishers, 1992.

[MARH97] D. W. Marhefka and D. E. Orin. Gait planning for energy efficiency in walking machines. In *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, 1997.

[MAYE00] P. Mayé. *Moteurs électriques pour la robotique.* Dunod, 2000.

[MCGH79] R.B. McGhee and G.I. Iswandhi. Adaptive locomotion of a multilegged robot over rough terrain. *IEEE transactions on systems, man, and cybernetics*, 9(4):176–182, 1979.

[MIT] M. Dahleh, M. A. Dahleh, and G. Verghese. Lectures on dynamics systems and control. Massachussets Institute of Technology.

[MIT2] http://www.ai.mit.edu/projects/humanoid-robotics-group.

[MOLN10] I. Molnar. Real-time preempt, February 2010.

URL `http://people.redhat.com/mingo/realtime-preempt/`

[MULL09] P. Müllhaupt. *Introduction à l'analyse et à la commande des systèmes non-linéaires.* Presses Polytechniques et Universitaires Romandes, 2009.

[NAIR92] S.S. Nair, R. Singh, K.J. Waldron, and V.J. Vohnout. Power system of a multi-legged walking robot. *Robotics and Autonomous Systems*, 9:149–163, 1992.

[OLSS96] H. Olsson. *Control Systems with Friction.* PhD thesis, Lund Institute of Tehcnology, 1996.

[OLSS98] H. Olsson, K. J. Aström, C. Canudas de Wit, M. Gafwert, and P. Lischinsky. Friction models and friction compensation. *European J. of Control*, 4(3):176–195, 1998.

[OMIC] http://www.oricomtech.com.

[ORIN82] D. E. Orin. Supervisory control of a multilegged robot. *The International Journal of Robotics Research*, 1:79–91, 1982.

[OUEZ98] F. B. Ouezdou, O. Bruneau, and J. C. Guinot. Dynamic analysis tool for legged robots. *Multibody System Dynamics*, 2:369–391, 398.

[PAL91] P. K. Pal and K. Jayarajan. Generation of free gait - a graph search approach. *IEEE trnasactions on robotics and automation*, 7(3):299–305, June 1991.

[PAPA04] E.G. Papadopoulos and G.C. Chasparis. Analysis and model-based control of servomechanisms with friction. *ASME Journal of Dynamic Systems, Measurement and Control*, 126(4):911–915, 2004.

[PARI10] Bernard Parisse and René De Graeve. *Giac/Xcas*, 2010.

URL `http://www-fourier.ujf-grenoble.fr/parisse/giac_fr.html`

[PARK01] Gary B. Parker. The incremental evolution of gaits for hexapod robots. In *Proceedings of the Genetic and Evolutionary Computation Conference.* GECCO 2001, July 2001.

[PAUL81] R.P. Paul. *Robot manipulators: Mathematics, Programming, and Control.* MIT Press, Cambridge, MA, 1980.

[PEAR76] K. Pearson. The control of walking. *Scientific American*, 235:72–86, 1976.

[PFEI00] F. Pfeiffer and Th. Roßmann. About friction in walking machines. In *Int. Conf. on Robotics and Automation*, San Francisco - USA, April 2000.

[PFEI04] F. Pfeiffer and C. Glocker. *Multibody dynamics with unilateral contacts.* Wiley-VCH, 2004.

[PORT04] J.M. Porta and E. Celaya. Reactive free-gait generation to follow arbitrary trajectories with a hexapod robot. *Robotics and Autonomous Systems*, 47:187–201, 2004.

[PREU91] A. Preumont, D. Ghuys, and Y. Baudoin. Etude cinématique et de stabilité des hexapodes. *European Journal Mech. Eng.*, 37(2):109–115, 1991.

[PREU94] A. Preumont. An investigation on the kinematic control of a six-legged walking robot. *Mechatronics*, 4(8):821–829, 1994.

[PREU97] A. Preumont, P. Alexandre, I. Doroftei, and F. Goffin. A conceptual walking vehicle for planetary exploration. *Mechatronics*, 7(3):287–296, 1997.

[RABI58] E. Rabinowicz. The intrinsic variables affecting the stick-slip process. *Proc. Physical Society of London*, pages 668–675, 1958.

[REIC00] J. A. Reichler and F. Delcomyn. Dynamics simulation and controller interfacing for legged robots. *The International Journal of Robotics Research*, 19:41–57, 2000.

[ROBE97] B. Robertson. A preview of the european commission teleman programme for telerobotics research. *IEEE Robotics and Automation Magazine*, 4(4), December 1997.

[ROBU] http://www.clarity-support.co.uk/products.

[SALM96] S. Salmi and A. Halme. Implementing and testing a reasoning-based free gait algorithm in the six-legged walking machine MECANT. *Control Eng. Practice*, 4(4):487–492, 1996.

[SAMI03] J.C. Samin and P. Fisette. *Symbolic Modelling of Multibody Systems*. Kluwer Academic Publisher, 2003.

[SCHM03] U. Schmucker, A. Schneider, and V. Rusin. Interactive virtual simulator (ivs) of six-legged robot "Katharina". In *Proceedings of the 6th International Conference CLAWAR*, Catania - Italy, 2003.

[SCHM96] U. Schmucker, A. Schneider, and T. Ihme. Hexagonal walking vehicle with force sensing capability. In *International Symposium on Measurement and Control in Robotics*, Brussels, 1996.

[SCHN06] A. Scheider and U. Schmucker. Force sensing for multi-legged walking robots. In J. Buchli, editor, *Mobile Robot, Moving Intelligence*, pages 447–470. ARS/plV, 2006.

[SHEN08] Jingjin Shen, Chenggang Li, and Hongtao Wu. A research on dynamics of a hexapod with closed-loop legs. In Caihua Xiong, Honghai Liu, Yongan Huang, and Youlun Xiong, editors, *Intelligent Robotics and Applications*, volume 5314 of *Lecture Notes in Computer Science*, pages 1060–1069. Springer Berlin / Heidelberg, 2008.

[SHIH87] L. Shih, A.A. Frank, and B. Ravani. Dynamic simulation of legged machines using a compliant joint model. *The International Journal of Robotics Research*, 6:33–46, 1987.

[SICI09] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics - Modelling, planning and Control*. Springer-Verlag London Limited, 2009.

[SILO4] Industrial Automation Institute - Madrid. *The SILO4 Walking Robot*, July 2000.

URL `http://www.iai.csic.es`

[SILV05] M.F. Silva, J. A. Tenreiro Machado, and A. M. Lopes. Modelling and simulation of artificial locomotion systems. *Robotica*, 23:595–606, 2005.

[SILV08] M.F. Silva and J. A. Tenreiro Machado. Kinematic and dynamic performance analysis of artificial legged systems. *Robotica*, 26:19–39, 2008.

[SONG89] S. Song and K. J. Waldron. *Machines That Walk - The Adaptive Suspension Vehicle*. The MIT Press, 1989.

[SPIN97] G. Spinnler. *Conception des Machines - Principes et Applications*, volume Tome 1 and 2. Presses Polytechniques et Universitaires Romandes, 1997.

[SWEV00] J. Swevers, F. Al-Bender, C. Ganseman, and T. Prajogo. An integrated friction model structure with an improved presliding behavior for accurate friction compensation. *IEEE Transactions on Automatic Control*, 45:675–686, April 2000.

[TIT] http://www-robot.mes.titech.ac.jp.

[TKK] http://automation.tkk.fi/MECANT.

[TSUC06] K. Tsuchiya, S. Aoi, and K. Tsujita, editors. *Adaptative Motion of Animals and Machines*, chapter A Turning Strategy of a Multi-legged Locomotion Robot, pages 227–236. Springer Tokyo, 2006.

[UCHI00] H. Uchida, N. Shimoi, H.Q. Jiu, D. Komizo, and K. Nonami. Force/attitude control of mine detectinf six-legged locomotion robot. In *Proc. Of the 2000 IEEE/RSJ International Conference on Intellignet Robot and Systems*, 2000.

[ULB] http://www.ulb.ac.be/scmero/.

[VANB96] H. M.J. Van Brussel. Mechatronics – a powerful concurrent engineering framework. *IEEE/ASME Transactions on Mechatronics*, 1(2), June 1996.

[VERL01] P. Verlinde, M. Acheroy, and Y. Baudoin. The belgian humanitarian demining project and the european research context. *European Journal Mech. and Env. Eng.*, 46(2):69–98, 2001.

[VERL05b] O. Verlinden, K. Kouroussis, and C. Conti. Easydyn: A framework based on free symbolic and numerical tools for teaching multibody systems. In *Eccomas Thematic Conference*, Madrid, Spain, June 2005.

[VERL07] O. Verlinden and G. Kouroussis. *EasyDyn 1.2.4 — C++ library for the easy simulation of dynamic problems.* Faculté Polytechnique de Mons, April 2007.

[WANG08] Peng fei Wang, Man tian Li, and Li ning Sun. Body posture control of wheeled foot quadruped robot based on virtual suspension model. In Caihua Xiong, Honghai Liu, Yongan Huang, and Youlun Xiong, editors, *Intelligent Robotics and Applications*, volume 5314 of *Lecture Notes in Computer Science*, pages 834–843. Springer Berlin / Heidelberg, 2008.

[ZENN04] Y. Zennir. *Apprentissage par renforcement et systèmes distribués : application à l'apprentissage de la marche d'un robot marcheur hexapode.* PhD thesis, Institut national des sciences appliquées de Lyon, 2004.

[ZLAJ08] Leon Zlajpah. Simulation in robotics. *Math. Comput. Simul.*, 79(4):879–897, 2008.