FACULTY OF ENGINEERING
AND ARCHITECTURE

Biologically Inspired Locomotion of Compliant Robots

Gabriel Urbain

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Computer Science Engineering

Supervisors

Prof. Francis wyffels, PhD - Prof. Joni Dambre, PhD

Department of Electronics and Information Systems
Faculty of Engineering and Architecture, Ghent University

September 2021

GHENT
UNIVERSITY

FACULTY OF ENGINEERING
AND ARCHITECTURE

Biologically Inspired Locomotion of Compliant Robots

Gabriel Urbain

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Computer Science Engineering

Supervisors
Prof. Francis wyffels, PhD - Prof. Joni Dambre, PhD

Department of Electronics and Information Systems
Faculty of Engineering and Architecture, Ghent University

September 2021

GHENT
UNIVERSITY

ISBN 978-94-6355-514-2
NUR 959, 958
Wettelijk depot: D/2021/10.500/62

Members of the Examination Board

Chair

Prof. Filip De Turck, PhD, Ghent University

Other members entitled to vote

Prof. Tony Belpaeme, PhD, Ghent University
Prof. Guillaume Crevecoeur, PhD, Ghent University
Prof. Auke Jan Ijspeert, PhD, Ecole Polytechnique Fédérale de Lausanne, Switzerland
Prof. Renaud Ronsse, PhD, Université catholique de Louvain

Supervisors

Prof. Francis wyffels, PhD, Ghent University
Prof. Joni Dambre, PhD, Ghent University

Acknowledgements

This book represents the achievements of four years of research, but it also concludes more than twenty years spent on the benches of different schools and universities, where I benefited from a priceless education and excellent advice. The list of people I would like to thank is so long that I can, unfortunately, cite only a few of them in a non-exhaustive list.

First of all, I would like to express my gratitude to my promotor, Prof. Francis wyffels, and my co-promotor, Prof. Joni Dambre for their constant follow-up during this Ph.D. thesis. I must admit that my methodology has evolved strongly over these last six years and they are the main authors of this change. My acknowledgment also goes to the Human Brain Project for supporting my Ph.D. financially and for all the opportunities that it offered me during the last six years. I would also like to thank deeply Victor Barasuol, Claudio Semini, and all the team at IIT DLS, who have welcomed me, guided me through the complex technical world of robotics, and supported me until late at night to help me conduct my experiments. My thanking also goes to Alexander Vandesompele, who co-authored some publications and would always take the time to carefully discuss new ideas. I would also like to thank Zoé Van Hoef, who wonderfully designed most illustrations in this manuscript, making it more accessible to the reader. Finally, my thoughts also go to my colleagues from AIRO, HBP Education, or the GUSO, who were entirely part of my Ph.D. life.

If the work conducted in this thesis would not be possible without the technical and scientific help gathered at university, neither would it be without the moral and human support of my friends. From the Royal Athenee of Mons, until today, through the University of Mons, Supaero, or the MIT, I had the priceless chance to always

be surrounded by happy, smiling, humble and passionate people, whose presence always helped me to tackle the various difficulties on my way. Last but not least, I do not know how to express my gratitude to my family. I would not have gone far without my mother, my father, and my sister, nor my grandparents, uncles, aunts, cousins,... Finally, a special thank goes to Nathalie, who has shared my life for more than four years and who, through her encouragement, her love, and her trust, let me progress every day in my passions.

Remerciements

La concrétisation de ce livre est l'aboutissement de quatres années de recherches, mais il conclut aussi plus de vingt années passées sur les bancs de différentes écoles et d'universités, où j'ai eu la chance de bénéficier d'une instruction et de conseils formidables et variés. La liste des gens que j'aimerais remercier est donc très longue et je ne peux me contenter que d'en citer seulement quelques-uns, de manière non-exaustive.

Tout d'abord, j'aimerais témoigner ma gratitude à mon promoteur, Prof. Francis wyffels, et mon co-promoteur, Prof. Joni Dambre pour leur suivi constant durant cette thèse de doctorat. Je dois reconnaître que ma méthodologie de travail a fortement évolué sur ces quatres dernières années et ils en sont les principaux artisans. Ma reconnaissance va également au Human Brain Project pour le soutien financier de mon doctorat et pour toutes les opportunités offertes au cours des six dernières années. Par ailleurs, je voudrais remercier profondément Victor Barasuol, Claudio Semini et toute l'équipe de l'IIT DLS, qui m'ont accueilli, guidé à travers le monde complexe de la robotique, et soutenu jusqu'à tard dans la nuit pour m'aider à mener mes expériences. Mes remerciements vont également à Alexander Vandesompele, qui a partagé mon bureau et mes papiers et a toujours pris le temps de discuter chaque nouvelle idée. J'aimerais aussi remercier Zoé Van Hoef, qui a merveilleusement réalisé la plupart des illustrations dans cette dissertation, la rendant de fait plus accessible au lecteur. Enfin, mes pensées vont également à mes collègues du laboratoire de recherche AIRO, de HBP Education ou du GUSO, qui ont fait partie intégrante de ma vie de doctorant.

Si ce travail de thèse n'eut pas été réalisable sans les conseils techniques et scientifiques recoltés dans le milieu universitaire, il

n'en serait pas moins possible sans le support moral et la contribution humaine de mes amis. Depuis les bancs de l'Athénée Royal de Mons, jusqu'à aujourd'hui, en passant par l'Université de Mons, Supaero, ou le MIT, j'ai eu la chance inestimable d'être toujours entouré de gens joyeux, souriants, humbles et passionnés, dont la présence me permet d'aborder de manière détendue les différentes difficultés sur le chemin. Last but not least, je ne sais comment témoigner ma reconnaissance envers ma famille. Je ne serais pas allé bien loin sans ma mère, qui a guidé mes premiers pas et m'a appris à avancer avec assurance, mon père, qui m'a enseigné à viser l'excellence, ma soeur, sur qui je peux compter à tout instant, dans les meilleurs moments comme dans les pires, mes grands-parents, mes oncles, tantes, cousins,...

Enfin, il y a bien sûr Nathalie, qui partage ma vie depuis plus de quatre ans et qui, par ses encouragements, son amour, et sa confiance dans mes choix, me laisse progresser quotidiennement dans mes passions, tout en me comblant de sa présence.

Summary

A better understanding of locomotion, and the processes that it involves, has potential benefits in our society. On one hand, it could help to design efficient legged robots, with better accessibility to the different environments on the globe, whereas wheeled platforms remain generally limited on even terrains without obstacles. This could be used to build social robots or to deal with exploration and rescue operations in hazardous environments. On the other hand, improving the understanding of the locomotion mechanisms can also contribute to biological sciences and, in particular, neurosciences. In this regard, studies on locomotion are a typical illustration of an embodied problem, which is often cited as a key concept to bridge the different scales in brain research, from the chemical and physical processes to the behavioral and psychological aspects.

To improve the agility and adaptability of robotic locomotion platforms, an appealing path is to use compliant structures and actuators rather than stiff elements. However, compliant and soft robots are not well suited for control with traditional computational architectures, generally directed towards centralized commands and exactness. New models, driven by data, or inspired by the broad locomotion abilities encountered within biological systems, create an opportunity to improve the state of the art in robotics. But they also raise questions on several fundamental aspects, limiting their maturity and their diffusion in the society. This dissertation tries to better investigate three of these research questions: how can we transfer knowledge from simulation to real robots, how does the mechanical compliance correlate with the locomotion performance and the controller complexity, and, thirdly, how can reflex-based control on compliant structures benefit from a stance correction mechanism taking its inspiration in the biological Cerebellum.

In the introduction and the state-of-the-art chapters, I identify these questions more precisely and provide an overview of the existing literature on the subject. The next chapter gives more details on the control methodology used in this dissertation. Chapter 4 presents three different robot platforms to target the research questions: a simulated network of masses and springs (MSD structure), a cheap passive compliant quadruped robot (Tigrillo), and a state-of-the-art quadruped robot with active compliant actuators (HyQ). The next four chapters expose the approach and discuss the results of the experimental trials conducted on these platforms, to formulate contributions in the domain. Finally, a conclusion is provided in the last chapter.

The contribution of this dissertation is manifold and five main topics are investigated throughout the manuscript. First, the recent progress in machine learning has led to impressive results for the locomotion of simulated creatures. However, mechanical compliance is not often considered in the work from this field and the difficulty to transfer trained parametric models from simulation to the real world has been highlighted in different research tracks. In this dissertation, I suggest an optimization procedure of the physics simulation model to reduce the difference between observations in simulation and the real world.

Secondly, an analysis of the state of the art in the field of robotics shows that compliance is not a straightforward defined concept, although it is generally linked to two physical parameters, damping and stiffness. I further analyze this relation through empirical analysis of non-linear robotic systems. They seem to indicate that the concept of resonance, only strictly defined in systems with second-order ordinary differential equations, could serve as a first approximation of the link between stiffness, damping, and optimal locomotion frequency. An investigation of this dependence is conducted on the MSD structures and the HyQ robot.

Thirdly, this dissertation presents a training method in two steps: a parameter optimization of open-loop biologically inspired models, followed by a supervised training of a feed-forward neural network, linking robot's sensors and actuators, to reproduce the targets obtained in the first step in closed-loop. This architecture demonstrates the ability to learn a reflex-based locomotion model without the need for centralized control. The properties of this closed-loop dynamical system are investigated on the three different robotic platforms and the robustness against external disturbance is also discussed.

Fourthly, to connect this reflex-based model with biological observations, I also compare two architectural hypotheses in a simple stability controller for a quadruped robot: one using an internal spatiotemporal representation of the body system, and the other based on afferent sensor signals from the lower limbs. I show that the first model performs better in maintaining a target locomotion frequency and resisting external disturbances, which corroborates biological observations conducted on the Cerebellum's functioning.

All these items put together establish an ideal framework to further investigate the potential exchange of computation capacity between physical body and controller during locomotion, as expected from the theory of morphological computation. A reflection on the matter constitutes the fifth contribution of this dissertation. The experiments on HyQ and the MSD networks promote another formulation of this phenomenon: although it is not possible to confirm a transfer of computation per se in my trials, we observe that increased structural complexity and larger mechanical compliance contribute to the simplification of computational requirements in the controller and promote a more stable locomotion process against external disturbance.

In conclusion, this work contributes to better understanding the impact of mechanical compliance on the design and the tuning of a locomotion controller. The experimental results advocate in favor of the use of compliance in robotics, not only to improve

performance but also to simplify the control, in association with generic and data-driven controllers. Furthermore, anchoring the architecture of these controllers into biological observations proves to be a source of inspiration for enhanced robots but also a way to test hypotheses and better understand the phenomena in action during human and animal locomotion.

Samenvatting

Een beter begrip van voortbeweging en de processen die daarbij een rol spelen heeft potentiële voordelen voor onze samenleving. Enerzijds zou het kunnen helpen bij het ontwerpen van efficiënte robots met poten, die beter toegankelijk zijn voor de verschillende omgevingen op aarde, terwijl platforms met wielen doorgaans beperkt blijven tot effen terreinen zonder obstakels. Dit zou kunnen worden gebruikt om sociale robots te bouwen of om verkennings- en reddingsoperaties in gevaarlijke omgevingen uit te voeren. Anderzijds kan een beter begrip van voortbewegingsmechanismen ook bijdragen tot de biologische wetenschappen en in het bijzonder de neurowetenschappen. In dit opzicht zijn studies over voortbeweging een typische illustratie van een belichaamd probleem, dat vaak wordt aangehaald als een sleutelconcept om de verschillende schalen in hersenonderzoek te overbruggen, van de chemische en fysische processen tot de gedragsmatige en psychologische aspecten.

Om de wendbaarheid en het aanpassingsvermogen van robotische voortbewegingsplatformen te verbeteren, is het gebruik van soepele structuren en actuatoren in plaats van stijve elementen een aantrekkelijke optie. Soepele en zachte robots zijn echter niet goed geschikt voor besturing door traditionele computerarchitecturen, over het algemeen gericht op gecentraliseerde commando's en exactheid. Nieuwe modellen, aangedreven door gegevens of geïnspireerd door de brede voortbewegingsmogelijkheden die in biologische systemen worden aangetroffen, bieden de kans om de stand van de techniek in robotica te verbeteren. Maar ze roepen ook vragen op over verschillende fundamentele aspecten, waardoor ze nog niet rijp genoeg zijn om op grote schaal in de samenleving te worden toegepast. Dit proefschrift probeert drie van deze onderzoeksvragen beter te onderzoeken: hoe kunnen we conclusies

verkregen uit simulaties toepassen op echte robots, hoe correleert de mechanische compliantie met de voortbewegingsprestatie en de complexiteit van de controller, en ten derde, hoe kan een reflexgebaseerde controle op verende structuren baat hebben bij een standcorrectiemechanisme dat zijn inspiratie vindt in het biologische cerebellum.

In de inleidende hoofdstukken, stel ik deze vragen nader vast en geef ik een overzicht van de literatuur over dit onderwerp. Het volgende hoofdstuk geeft meer details over de controlemethodologie die in dit proefschrift wordt toegepast. Hoofdstuk 4 stelt drie verschillende robotplatformen voor om de onderzoeksvragen te beantwoorden: een gesimuleerd netwerk van massas, dempers en veren (MSD structuur), een goedkope passieve verende vierpotige robot (Tigrillo) en een state-of-the-art vierpotige robot met actief verende actuatoren (HyQ). Experimenten op deze drie platforms worden gebruikt om bijdragen in het domein te formuleren. In de volgende vier hoofdstukken wordt de aanpak uiteen gezet en worden de resultaten besproken van de experimentele proeven die op deze platformen zijn uitgevoerd om bijdragen te formuleren op het gebied. Tenslotte wordt in het laatste hoofdstuk een conclusie gegeven.

De bijdrage van dit proefschrift is veelzijdig. Vijf belangrijke thema's worden onderzocht en besproken doorheen het manuscript. Ten eerste, heeft de recente vooruitgang in machinaal leren geleid tot indrukwekkende resultaten voor de voortbeweging van gesimuleerde wezens. Mechanische compliantie wordt in het werk op dit gebied echter niet vaak in overweging genomen. De moeilijkheid om getrainde parametrische controllers voor die structuren over te dragen van simulaties naar de reële wereld is in verschillende onderzoekstrajecten naar voren gekomen. In dit proefschrift stel ik een optimalisatieprocedure voor van het fysische simulatiemodel om het verschil tussen waarnemingen in de simulatie en in de echte wereld te verkleinen.

Ten tweede, blijkt uit een analyse van de stand van de techniek op het gebied van robotica dat compliantie geen eenduidig gedefinieerd concept is, hoewel het in het algemeen gekoppeld is aan twee fysieke parameters, demping en stijfheid. Ik analyseer dit verband verder aan de hand van een empirische analyse van niet-lineaire robotsystemen. Ze lijken erop te wijzen dat het concept van resonantie, dat alleen strikt gedefinieerd is in systemen met een tweede-orde gewone differentiaalvergelijkingen, zou kunnen dienen als een eerste benadering van het verband tussen stijfheid, demping en optimale voortbewegingsfrequentie. Een onderzoek naar deze afhankelijkheid wordt uitgevoerd op de MSD-structuren en de HyQ-robot.

Ten derde, presenteert dit proefschrift een trainingsmethode in twee stappen: een parameteroptimalisatie van open-loop biologisch geïnspireerde modellen, gevolgd door een gecontroleerd leren van een feedforward neuraal netwerk, dat sensoren en actuatoren van robots verbindt om de doelen verkregen in de eerste stap in closed-loopsysteem te reproduceren. Deze architectuur demonstreert de mogelijkheid om een reflexgebaseerde voortbewegingsmodel te leren zonder de noodzaak van een gecentraliseerde controle. De eigenschappen van dit closed-loop dynamisch systeem worden onderzocht op de drie verschillende robotplatformen en de robuustheid tegen externe verstoringen wordt ook besproken.

Ten vierde, om dit reflexgebaseerde model te verbinden met biologische waarnemingen, vergelijk ik ook twee architecturale hypothesen in een eenvoudige stabiliteitscontroller voor een vierpotige robot: één met een interne spatiotemporele weergave van het lichaamssysteem, en de andere gebaseerd op afferente sensorsignalen van de onderste ledematen. Ik toon aan dat het eerste model beter presteert in het behoud van een bepaalde voortbewegingsfrequentie en in de weerstand tegen externe verstoringen, wat biologische waarnemingen over de werking van het cerebellum bevestigt.

Al deze punten samen vormen een ideaal kader om de potentiële uitwisseling van rekencapaciteit tussen het fysiek lichaam en de controller tijdens voortbeweging verder te onderzoeken, zoals verwacht wordt uit de theorie van morfologische computatie. Een beschouwing hierover vormt de vijfde bijdrage van dit proefschrift. De experimenten op HyQ en de MSD-netwerken bevorderen een andere formulering van dit verschijnsel: hoewel het niet mogelijk is om een overdracht van computatie op zich in onze proeven te bevestigen, stellen wij vast dat een grotere structurele complexiteit en een grotere mechanische compliantie bijdragen tot de vereenvoudiging van computationele vereisten in de controller en een stabieler voortbewegingsproces bij externe verstoringen bevorderen.

Tot slot, draagt dit werk bij tot een beter begrip van de impact van mechanische compliantie op het ontwerp en de afstemming van een voortbewegingscontroller. De experimentele resultaten pleiten voor compliantie in de robotica, niet alleen om de prestaties te verbeteren maar ook om de controle te vereenvoudigen, in combinatie met generieke en datagestuurde controllers. Bovendien, blijkt de verankering van de architectuur van deze controllers in biologische waarnemingen een bron van inspiratie te zijn voor verbeterde robots, maar ook een manier om hypotheses te testen en de verschijnselen in actie tijdens menselijke en dierlijke voortbeweging beter te begrijpen.

List of Symbols

Abbreviations

AI	Artificial Intelligence
COG	Center Of Gravity
CPG	Central Pattern Generator
COT	Cost Of Transport
DRL	Deep Reinforcement Learning
DOF	Degree Of Freedom
DOA	Degree Of Actuation
DL	Delay Line
ESN	Echo State Networks
ELM	Extreme Learning Machine
EA	Evolution Algorithms
FMSE	Frequency Mean Square Error
GRF	Ground Reaction Force
HAA	Hip Adduction-Abduction
HFE	Hip Flexion-Extension
ID	Inverse Dynamics
IIT	Italian Institute of Technology
IK	Inverse Kinematics
IMU	Inertial Measurements Unit
KFE	Knee Flexion-Extension
LSM	Liquid State Machine
MAE	Mean Absolute Error
MAWE	Mean Absolute Windowed Error
MSE	Mean Square Error
MSD	Mass-Spring-Damper
MPC	Model Predictive Control

NRP	Neurorobotics Platform
NRMSE	Normal Root Mean Square Error
PCA	Principal Component Analysis
PRC	Physical Reservoir Computing
PSE	Predictive Stance Estimator
PID	Proportional Integral Derivative
RCF	Reactive Controller Framework
RSE	Reactive Stance Estimator
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
RL	Reinforcement Learning
RC	Reservoir Computing
ROS	Robot Operating System
SEA	Series Elastic Actuators
SNR	Signal-to-Noise Ratio
SNN	Spiking Neural Networks
SL	Supervised Learning
ZMP	Zero Moment Point

Mathematical Notations

Robots

Legged Robots

j	$\in [1, ..., J]$	Joint index
l	$\in [1, ..., L]$	Leg index
k	$\in [1, ..., K]$	Time step index
t_k		Discrete time step
m		Mass of the robot
$\mathbf{\Omega}$	$= (\phi, \theta, \psi)$	Attitude vector of the robot's trunk
$\mathbf{x}$	$= (x, y, z)$	Position vector of the robot's trunk
$\dot{\mathbf{x}}$		Velocity vector of the robot's trunk
$\ddot{\mathbf{x}}$		Acceleration vector of the robot's trunk
$\mathbf{q}$	$= \{q^j\}$	Vector of robot's joint position

$\dot{\mathbf{q}}$	$= \{\dot{q}^j\}$	Vector of robot's joint velocity
$\boldsymbol{\tau}$	$= \{\tau^j\}$	Vector of robot's joint torques
$\boldsymbol{\tau}^{\text{stab}}$	$= \{\tau^{\text{stab},\ j}\}$	Vector of robot's stabilization torques
$\boldsymbol{\tau}^{\text{mot}}$	$= \{\tau^{\text{mot},\ j}\}$	Vector of robot's motion torques
$\mathbf{f}^{\text{GRF}}$	$= \{f^{\text{GRF},l}\}$	Vector of Ground Reaction Forces on the feet
$\boldsymbol{\mu}$	$= \{\mu^l\}$	Vector of friction coefficients for the robot feet
$\boldsymbol{\epsilon}$	$= \{\epsilon^l\}$	Vector of simulation penetration coefficients for the robot feet
θ		Simulation model parameters
$\mathbf{a}_k$		Actuator signal (torque of position depending on the robot)
$\mathbf{s}_k$		Sensors signal
$\mathbf{A}_p$		Discrete-time Fourier transform of the actuators signal
$\mathbf{S}_p$		Discrete-time Fourier transform of the sensors signal
$\mathcal{F}^{\text{r}}$		Transfer function of the real robot
$\mathcal{F}^{\text{s}}$		Transfer function of the simulated robot

Mass-Spring-Damper Networks

i	$\in [1, ..., I]$	Node index
j	$\in [1, ..., J]$	Link index
k	$\in [1, ..., K]$	Time step index
t_k		Discrete time step
$\mathbf{m}$	$= \{m^i\}$	Nodes masses
$\mathbf{x}^i$	$= (x^i, y^i, z^i)$	Position vector of node i
$\mathbf{k}$	$= \{k^j\}$	Links stiffness coefficients
$\mathbf{d}$	$= \{d^j\}$	Links damping coefficients
$\mathbf{l}$	$= \{l^j\}$	Links lengths
$\mathbf{v}$	$= \{v^j\}$	Links extension velocity
C		Number of neighbor connections of a node
α^j, ω		Amplitude, frequency and phase offset of the link control signal
$l^{\text{ref},\ j}$		Equilibrium length of link j
β		Non-linearity coefficient of the springs
g		Universal gravitation constant
γ		Air friction coefficient

Metrics

S	Stability
P	Power
V	Robot speed
A	Accuracy
COT	Cost Of Transport
$\mathrm{GRF}^{\max}$	Maximal Ground Reaction Force
MSE_k	Mean Square Error
NRMSE_k	Normalized Root Mean Square Error
FMSE	Frequency Mean Square Error
MAE_k	Mean Absolute Error
MAWE_k	Mean Absolute Windowed Error

Controllers

General Notations

t	Continuous time
t_k	Discrete time
$\mathbf{a}(t),\ \mathbf{a}_k$	Control signal (actuation)
$\mathbf{s}(t),\ \mathbf{s}_k$	Feedback signal (sensing)
$\mathbf{r}(t),\ \mathbf{r}_k$	Target signal
$\mathbf{e}(t),\ \mathbf{e}_k$	Error signal
$\boldsymbol{\pi}$	Control policy
$\boldsymbol{\pi}^\star$	Optimal control policy
$\boldsymbol{\theta}$	Policy parameters
$\hat{\boldsymbol{\theta}^\star}$	Estimation of optimal policy parameters

Classical Locomotion

κ	Linear system gain
$\phi(t)$	Disturbance noise on the system
$k^{\mathrm{p}}, k^{\mathrm{i}}, k^{\mathrm{d}}$	Gains of a PID controller
$\mathbf{x}_k$	Robot's state signal
$\hat{\mathbf{x}}_k$	Estimation of the robot state signal
$\mathbf{q}_k$	Robot's joints signal
$\boldsymbol{\tau}_k^{\mathrm{stab}}$	Stabilization torques of the robot's joints
$\boldsymbol{\tau}_k^{\mathrm{mot}}$	Motion torques of the robot's joints

Evolution Algorithms

i	Generation index
λ	Population size
G_i	Generation score

Supervised Learning

$\boldsymbol{f}(.)$	Non-linear function of the neuronal readout layer
J	Dimension of the robot's joint sensors vector
U	Dimension of the processed sensors vector
$\mathbf{q}_k$	Robot's joints vector signal
$\boldsymbol{u}_k$	Processed sensors vector signal
$\boldsymbol{r}_k$	Target signal
$\mathbf{e}_k$	Error signal
$\mathbf{W}_k^{\text{out}}$	Weight matrix of the readout layer
$\mathbf{P}_k$	Estimate of the inverse of the correlation matrix
$\mathbf{I}$	Identity matrix

Reinforcement Learning

$Pr(a \mid b)$	Probability of a random process a if b
$E[a]$	Expectation of the random process a
$\mathbf{q}_k$	Action signal
$\mathbf{u}_k$	State signal
r_k	Reward signal
$\boldsymbol{\pi}^{\boldsymbol{\theta}}(\mathbf{u}_k)$	Control policy with parameters $\boldsymbol{\theta}$
$V(\mathbf{u}_k)$	Value function starting from state $\mathbf{u}_k$
$Q(\mathbf{q}_k, \mathbf{u}_k)$	Q-value function
$\hat{Q}(\mathbf{q}_k, \mathbf{u}_k)$	Estimate of the Q-value of state-action
γ	Discount rate
η	Learning rate

Two-Steps Learning Approach

ρ	Parametric controller model in the first learning step
σ_i	Parameters of the controller at generation i
$\hat{\sigma^\star}$	Estimate of optimal controller parameters

Contents

List of Figures

CHAPTER 1

—

Introduction

Why building bio-inspired robot control architectures? What is it at stake? This chapter presents the different research questions investigated in this book, with their potential contributions to science and society.

Around five hundred years ago, Leonardo Da Vinci designed and built the *Automa Cavaliere* – automated knight – in his workshop in Milan. This man-sized armored automaton was capable of standing, sitting, raising its visor, and moving its arms independently using only cables and pulleys. In a sense, this automaton could be already called a robot. Even if the word 'robot' only exists since 1920, the concept of robotics itself is rather old. It also did not always relate to electronics and silicon semiconductors. Since the invention of the first tool, humans have tried to create machines and automate daily tasks. Over history, a series of technological advances made by scientists, engineers, and technicians, have pushed the limits of these achievements. The invention of the steam engine, electricity, electrical motors, telecommunications, electronics, and the computer have constituted, among others, the major ingredients in this thrilling enterprise. Today and with the recent advances in Artificial Intelligence (AI), the dream of a fully autonomous and intelligent machine, resembling and competing with humans, has been renewed in many people's minds. For others, this claim sounds utopian and even dangerous from a societal point of view. In practice, it is not easy to predict if such a robot will exist one day, nor when it would appear. **However, we can identify some paths to deal with the current limitations in robotics or to drive scientific progress toward an understanding of intelligence and I believe that mastering compliant locomotion is one of them.**

Learning Neuroscience Using Robots

In the creative process of developing automatons, humans have not only found a way to save time, avoid repetitive work, or develop crafting skills. **By modeling and reproducing their own bodies, or the ones of animals, and their interactions with the world, they also grew a fundamental understanding of the physiological and functional properties of biological systems. This process can be labeled as 'design science methodology' or 'constructive research' (Crnkovic, 2010).** In his paper 'Can a biologist

fix a radio', Lazebnik (2002) imagines an embarrassing scenario to highlight some limitations of the 'descriptive scientific method', and to promote the contribution of constructive research. To prove his point, he uses standard tools from experimental biology to analyze an old transistor radio. Dissecting and testing how the removal of each component affects the radio, eventually produces a catalog of the main components and the links between them. In the end, however, this knowledge would eventually fail to describe how to unequivocally repair the radio, whereas the engineering diagrams from the manual, created in a generative design process, would embed such information.

If this example is somewhat metaphoric, it emphasizes how synthesizing can be sometimes more insightful than dissecting a problem, especially for complex interconnected systems. Another historical illustration of this principle can be found in the ability to fly. For thousands of years, humans have described the movements of the birds in the sky, the material properties of their feathers, and how they articulate their wings in the air. Yet, with this meticulous description of birds, it was difficult to reproduce or even theorize a satisfying understanding of how they flew, up to recently. On the other hand, aerodynamics has developed hand-in-hand with a much more straightforward wing model, which has been the starting point of modern planes. If we can create, repair, and fly a plane, we have sufficient tools to understand it and the flying mechanism deeply.

Today, neuroscientists believe that the concept of constructive research should be used to complement experimental neuroscience in quest of understanding the brain, and more broadly, human intelligence. In Jonas & Kording (2017), the authors have transposed the example mentioned above, on analyzing a radio with tools from biology, to the study of a microprocessor with neuro-imagery. And they observed the same limitations with the 'descriptive scientific method'. The European initiative for large-scale research on the brain, called the Human Brain Project (HBP), has included in its initial goals the simulation of the entire human brain on

a High-Performance Computer (HPC) in the horizon 2024. This ambition may be excessive, but advances in realizing large-scale models have been made in the last years and their contribution has been largely discussed by the community (Einevoll et al., 2019). This path is also instructive about the new place of simulations in science. Nevertheless, building in simulation has two major drawbacks. Firstly, a simulator can never render the real world properly and can lead to an excessive difference in some specific cases. This pernicious effect is called the 'simulation-reality gap', or 'sim-to-real gap' in short (Lipson & Pollack, 2010). Although it can be attenuated using noise combined with evolution and learning algorithms, it can also lead to wrong or biased conclusions that are not applicable in the real world. Secondly, it is often argued that the time and resources spent on modeling and running realistic simulations could be more efficiently dedicated to building an experiment in the real world. This claim is not valid in all circumstances but invites us to consider the state of the art in robotics in the field.

In this context, real robots have contributed to the progress of neurosciences in the past decades (Floreano et al., 2014), and they will undoubtedly continue to be used in the future to validate hypotheses on neural architectures, cognitive mechanisms, or social interactions and development. An emblematic illustration is the work carried on Central Pattern Generators (CPGs) in robotic locomotion (Ijspeert, 2008), which demonstrated the validity of this fundamental mechanism in biological locomotion already described in Brown (1911). However, in this story, several factors are still to be clarified regarding, among others, the role of reflex feedback, the function of descending signals from the cortex, or the importance of muscle compliance. Building a robot locomotion controller that can adapt to different compliant bodies in various environments is, therefore, a convenient tool to derive essential laws about some brain functions.

Emerging Intelligence and Embodiment

It is often suggested that the ultimate goal of brain research is to understand human intelligence (Hawkins & Blakeslee, 2007). The same ambition is pursued in the field of Artificial Intelligence (AI), which aims at providing machines with an equivalent aptitude. **However, a prerequisite to this objective consists in defining correctly what is meant by 'intelligence' to avoid confusion when evaluating the current and future work.**

Influenced by our education system, we commonly define intelligence through the prism of knowledge, logic, and methodological thinking. Nevertheless, such a definition is not complete as it would imply that our laptop connected to the Internet is certainly more intelligent than we are. In Downing (2015), the authors review various definitions of intelligence, along with their deficiencies, and conclude that intelligence should be seen as an emerging behavior that includes adaptivity, robustness, and parallel evolutionary search in problem-solving. In the same perspective, seventy different definitions were gathered from various research fields in Legg & Hutter (2007). In this comprehensive review, the major features describing intelligence encountered across multiple literature references are summarized into three categories:

- The ability of an individual agent to interact with its environment,
- A relation between intelligence and the agent's ability to succeed or profit with respect to some goal or objective,
- The adaptation to different objectives and environments.

A compilation of these attributes allows to formulate a short informal definition:

> Intelligence measures an agent's ability to achieve goals in a wide range of environments.

This description can involve implicitly the ability to learn and adapt or to understand new external environments. While it remains imprecise regarding the type of goals and the variety of environments, the crucial element is to place the relationship between an agent and its environment as the core principle of intelligence. In robotics, we will generally restrict ourselves to embodied agents like robotic platforms, progressing in classical industrial or domestic environment and operating tasks such as recognition, classification, manipulation, or locomotion.

Adaptive interaction of the body with the environment is also the starting point of the work presented in Pfeifer et al. (2007). In this book, the authors deconstruct the Cartesian assumption '*Je pense, donc je suis*' and suggest embodiment as a key principle toward intelligent systems. **In this scope, it becomes clear that locomotion proposes an emblematic field to reach the goals of AI.** The large majority of animals display locomotion patterns to progress in their environments. In this sense, they achieve intelligent behavior. On the other hand, classical robotics has mostly employed a rational approach in the design of locomotion controllers. Despite recent successes that will be discussed later in this document, common techniques proved to be quite complex and limited to a restricted number of cases, in opposition to their biological counterparts.

In this trend, 'morphological computation' has been introduced as a theory to practically realize embodiment for a different set of tasks. Nonetheless, the concept of morphological computation does not have a clear definition, as discussed in Lizier et al. (2011). In Füchslin et al. (2013), the authors refer to the first *International Conference on Morphological Computation* in Venice in 2007, where it was first defined as:

> Any process that serves a computational purpose, has clearly assignable input and output states, is programmable (i.e., the behavior can be adapted by varying a set of parameters), and has a sort of teleological embedding.

This definition is, however, rather broad as it also includes every

traditional digital computing means. In this dissertation, I will restrict my definition to:

> Any way of increasing efficiency of computation in terms of energy, memory, time, etc by outsourcing computational tasks to analog physical systems.

This interpretation is in line with Pfeifer et al. (2007), where morphological computation is referred to as:

> Certain processes that are performed by the body that otherwise would have to be performed by the brain.

Practical Applications of Robot Locomotion

Beyond scientific and theoretical considerations, robotics has highly contributed to providing applications with a significant impact in real life. In this field, wheeled vehicles are most commonly used due to their simplicity and a low Cost Of Transport (COT) (i.e., their better energetic performance), as discussed in Tucker (1975). In contrast, robotic legged locomotion proved to be useful in many situations with unfriendly terrain covered with obstacles. This includes a broad range of sectors, stretching through agriculture, construction, mining operations, search and rescue, hazardous operations, health care, domestic help, or education (Siciliano & Khatib, 2017). This research domain has led to rapid improvements in the last decades due to miniaturization and increased performance of computational devices. More recently, advances in AI induced new perspectives, especially through the field of robot perception using vision or directly using control with Reinforcement Learning (RL) techniques.

However, despite the vast contribution in robotic locomotion, the field is yet far from approaching the incredible performance developed in living systems in terms of agility. Energetic performance, safety, cost, or complexity of the structures and the software are

some of the factors limiting the high potential of legged robots. Among the most promising paths for improvement, we can mention producing more flexible and robust materials, to mimic living systems (Calisti et al., 2017); improving the ratio between size, weight, power, and price of the actuators (Eschenauer et al., 2012); or the better miniaturization and integration of electronics, actuators, and sensors. Although these items have been identified for a long time, it remains challenging to tackle them in a comprehensive approach. Moreover, the quality of the controller remains the major disincentive to functional locomotion robots and explains mostly the difference in performance with their biological counterpart.

Research Problem and Goals

In the previous paragraphs, I have summarized three major motivations for studying robotic locomotion: **improving scientific knowledge in biomechanics and neural control, promoting a bottom-up conception of intelligence, and supporting technological advances in industrial and domestic systems.** As I have also mentioned, the progress in AI and the choice of softer compliant materials have inspired substantial expectations to the robotics community and the larger public.

However, performance in terms of robustness, energy consumption, speed, agility, or adaptivity is still far from biology. This claim is exemplified when we compare the capabilities of a cat, for instance, in an unknown environment against the most advanced state-of-the-art quadruped robots. The behavioral discrepancy that we perceive as humans is extremely large, and it becomes clear that there is an essential notion that is still not well captured in robotics.

To deal with this, locomotion has been investigated from various perspectives in multiple research domains, including robotics, biomechanics, neuroscience, AI, and morphological computation.

They all bring major contributions to a global understanding of locomotion, but the various approaches, the diversity of the vocabulary, and the different research frameworks make it hard to establish links between the different fields.

In this dissertation, I will discourse on a core question in neurorobotics: **how biology can inspire better controllers that exploit body properties, like mechanical compliance, during locomotion?** Using embodiment theory, and more precisely morphological computation as a starting point, I will further decompose this question into three specific research items:

1. How to **reduce the gap between simulation and real hardware** during the transfer of a controller from one to the other?
2. How is the mechanical **compliance** property linked with the **locomotion performance** and the **computational requirements** of the controller?
3. How can we inspire from **biology** to better investigate the interactions between **closed-loop reflex-based locomotion** and **open-loop position control** in robotics?

Research Outline

To position this research more accurately in the state of the art and provide the reader with the necessary background, **Chapter 2** presents a literature review on locomotion at the confluence of biology, robotics, machine learning, and embodiment. The relevant studies in the different fields are described and summarized in relation to the work conducted in this dissertation. This chapter is articulated in three sections, presenting references in robotic hardware, control software, and embodiment theory successively.

In **Chapter 3**, I define a common framework to describe different methods in locomotion control. In this regard, control architectures

from different fields are described with mathematical tools, and a comparison of their properties and advantages is also presented. A two-steps learning methodology is also formally introduced in this chapter. It will be subsequently applied in the rest of this dissertation to conduct all the different experiments.

The three robotic and simulation platforms which have been used in the experimental investigations are introduced in **Chapter 4**. Their features and benefits are also investigated to motivate how they fit the research conducted in this dissertation.

Concrete experimental investigations and discussions are presented in the next four chapters. I divide them into three parts, each of them tackling one of the research questions introduced above. To begin with, I discuss a calibration method in **Chapter 5**, to transfer a gait controller optimized in simulation to a real compliant robotic platform.

The second experimental part is dedicated to the relation between mechanical compliance and locomotion. In **Chapter 6**, I focus on the locomotion of Mass-Spring-Damper (MSD) structures in simulation. This study mainly investigates how compliance relates to energetic and speed performance and how the complexity of the body structure can affect the robustness and accuracy of an embodied gait with a generic controller. **Chapter 7** describes an embodied control framework for the actively compliant HyQ robot. The control principle is inspired by the experience with MSD structures and transposed to a large quadruped robot. Again, the compliance property is analyzed with reference to its effect on the robot performance in terms of gait stability, speed, and energy consumption but also in relation to the complexity of the controller.

The last experiments described in **Chapter 8** implements a specific hypothesis inspired by the biological evidence about the cerebellum functions or stability control. This work is also conducted on the HyQ robot and investigates the interactions between feedback-driven control and timed accurate position control. It also illus-

trates a concrete inspiration from brain functions to improve neural control on quadruped robots.

Finally, **Chapter 9** summarizes the contribution induced by the different works carried in this dissertation. A short overview of possible future work in this direction is also provided in this chapter.

Contributions

The contribution of this dissertation is manifold, and five main inputs are investigated and discussed throughout the manuscript. First, I suggest a new **optimization procedure of a physics simulation model to facilitate the transfer to the real world.** This approach has the benefit to reduce the amount of knowledge and hypotheses introduced in the design of the simulation model. Encouraging qualitative results have been observed on the Tigrillo robot.

Secondly, I provide an **empirical analysis of the relationship between stiffness, damping, and compliance, as well as their impact on the locomotion performance of non-linear systems.** This study promotes the concept of resonance of an equivalent system, although not generalizable to all non-linear systems, as a first approximation to determine the optimal compliance parameters on a robot.

The third contribution concerns a **training method in two steps**: an unsupervised parameter optimization of open-loop biologically inspired models, followed by a supervised training of a feed-forward neural network, linking robot's sensors and actuators, to reproduce the targets issued from the first step. This training procedure and this control architecture demonstrate the ability to learn a reflex-based locomotion model without the need for centralized control.

Stabilizing this reflex-based model can be obtained using an extra

controller for posture. The fourth contribution of this dissertation is to **anchor the architecture of this controller into biological observations on the mammal cerebellum**.

The combination of all these items provides an ideal framework to further investigate the possible exchange of computation capacity between physical body and controller during locomotion, as expected from the theory of 'morphological computation'. As the last contribution, I observe on HyQ and the MSD structures that **increased structural complexity and more substantial mechanical compliance** both lead to more coupling between the morphology and the controller. In turn, this is beneficial to **simplify the computational requirements in the controller and to provide increased robustness** against external disturbance.

List of publications

Journal publications

1. **Urbain G.**, Barasuol V., Semini C., Dambre J., and wyffels F. (2021). Effect of Compliance on Morphological Control of Dynamic Locomotion with HyQ. *Autonomous Robots*. (Accepted).

2. Vandesompele A., **Urbain G.**, wyffels F., and Dambre J. (2019). Populations of Spiking Neurons for Reservoir Computing: Closed-Loop Control of a Compliant Quadruped. *Cognitive Systems Research, 58, 317-323*.

3. Massi E., Vannucci L., Albanese U., Capolei M. C., Vandesompele A., **Urbain G.**, Sabatini A. M., Dambre J., Lschi C., Tolu S., and Falotico E. (2019). Combining evolutionary and adaptive control strategies for quadruped robotic locomotion. *Frontiers in Neurorobotics, 13, 71*.

4. Vandesompele A., **Urbain G.**, Mahmud H., wyffels F., and Dambre J. (2019). Body Randomization Reduces the Sim-to-Real Gap for Compliant Quadruped Locomotion. *Frontiers in Neurorobotics, 13, 9*.

5. **Urbain G.**, Degrave J., Carette B., wyffels F., and Dambre J. (2017). Morphological Properties of Mass-Spring Networks for Optimal Locomotion Learning. *Frontiers in Neurorobotics, 11, 16*.

Conference publications

1. **Urbain G.**, Barasuol V., Semini C., Dambre J., and wyffels F. (2020). Stance Control Inspired by Cerebellum Stabilizes Reflex-Based Locomotion on HyQ Robot. *Proceedings of the International Conference of Robotics and Automation, ICRA 2020*.

2. Vandesompele A., **Urbain G.**, wyffels F., and Dambre J. (2019). Closed-Loop Control of a Compliant Quadruped with Spiking Neural Networks. *Biologically Inspired Cognitive Architectures Meeting, 547-555*.

3. **Urbain G.**, Vandesompele A., wyffels F., and Dambre J. (2018) Design of a Bio-inspired Compliant Quadruped Robot for Research on Closed-Loop Locomotion Control. *2nd HBP Student Conference: Transdisciplinary Research Linking Neuroscience, Brain Medicine and Computer Science*.

4. **Urbain G.**, Vandesompele A., wyffels F., and Dambre J. (2018) Calibration Method to Improve Transfer from Simulation to Quadruped Robots. *International Conference on Simulation of Adaptive Behavior, 102-113*.

Abstracts, Demonstrations, Posters, Presentations

1. Scientific chair of the *4th HBP Student Conference*. January 2020. Pisa, Italy.

2. Chair and committee member of the *HBP Young Researcher Event*. July 2019. Belgrade, Serbia.

3. Host and scientific chair of the *3rd HBP Student Conference*. February 2019. Ghent, Belgium.

4. Technical demonstration and lightning talk at the *HBP SGA1 European Review*. May 2018. Stockholm, Sweden.

5. Presentation at the *Europe-Japan Neurorobotics Workshop*. April 2018. Tokyo, Japan.

6. Demonstration at the *European Robotics Week 2017*. November 2017. Brussels, Belgium.

7. Introduction speech and host of the *HBP SP10 Trimestrial Meeting*. September 2017. Ghent, Belgium.

8. Lightning talk and poster session at the *4th HBP School, Future of Artificial Intelligence*. June 2017. Obergurgl, Austria.

9. Lightning talk at the *1st Neurorobotics Platform User Workshop*. April 2017, Munich, Germany.

10. Lightning talk and poster session at the *3rd HBP School, Future of Neuroscience*. December 2016. Obergurgl, Austria.

11. Ph.D. vulgarization at *Let's Talk Science!, Communication Summer School*. July 2016. Ghent, Belgium.

12. Lightning talk and poster session at the *4th HBP Education Workshop, Future of Computing*. January 2016. Manchester, UK.

CHAPTER 2

—

State of the Art

In 2020, the two most famous conferences in robotics alone (namely ICRA and IROS) have totaled more than three thousand publications. Robotics progresses at a fast pace, and this chapter targets a few meaningful references to contextualize my work.

2.1 Compliant Legged Robots

In the early 1980s, influential work at the MIT Leg Lab demonstrated how dynamic locomotion gaits could be achieved using a few simple, decoupled control laws. Although robotic legged locomotion originated long before it, this accomplishment marked the beginning of a new era of extensive research in the field.

Through their agility and complexity, legged robots are suited to operate in situations where today's wheeled robots are still ineffective like exploration, rescue operations, and social interactions. This need for a new kind of mobile platform has proven especially true during recent hazards like nuclear failures, earthquakes, or fires. These robots have, and unfortunately according to me, also stimulated the interest in various military applications. Part of worldwide research is (directly or indirectly) funded through governmental, military budgets, as illustrated by the *DARPA Robotics Challenge* for instance. The most emblematic architecture used in legged locomotion is certainly the quadruped robot. Implicit identification with cats or dogs and the large diffusion of videos to the larger public on the internet have certainly contributed to this trend. However, the possibilities offered by quadruped locomotion should not be overlooked. The inherent stability provided by the use of four legs, and the possibility to switch between different gait regimes generally makes it a better compromise than two- or six-legged robots in real-world situations. This explains the larger emphasis on this type of legged robot. A second and no less important motivation in designing and operating quadruped robots is that they provide an excellent tool to test and validate hypotheses from bio-mechanics or neuroscience.

Despite the intensive research in legged robots, biological systems still present a large advance in terms of systemic integration, energy consumption, or adaptability in complex environments. To bridge this gap, mechanical compliance and softness have been identified as key concepts for improved performance. But the difficulties of finding the right materials to implement com-

pliant actuators, and the implications on the controller's complexity and efficiency are still raising many questions and challenges for the researchers.

In this section, I provide a short review of the most relevant work related to the design of compliant legged robots and more especially quadrupeds. I start from a biological perspective by providing some recent models in biomechanics, which are used to understand and reproduce locomotion patterns. A discussion on mechanical compliance is later carried and leads to the classification and the definition of different levels of compliance in robotics. Finally, some recent hardware platforms in the field of compliant quadruped robotics are presented. I group them into two categories according to their research goals: the robots aiming at practical application with a focus on the engineering part, and the robots for fundamental research in all domains of science.

2.1.1 Biological Inspiration

Biology is, without doubt, the principal source of inspiration for robotic locomotion (Dickinson et al., 2000). **On the one hand, biology helped in improving the state of the art in mechanical design, robotic control (Pfeifer et al., 2007), and AI (Marblestone et al., 2016; Hassabis et al., 2017). On the other hand, neuroscience has regularly benefited from robotics to conduct in vitro experiments (Floreano et al., 2014).** Literature in biology is full of physiological and functional studies describing the musculoskeletal structure and the low-level neural control of various mammals. With the increasing power available for computation, this data has been exploited to create simulated models for locomotion and explore the different hypotheses characterized in vivo. A series of musculoskeletal models have been suggested to reflect the locomotion properties of humans (Geyer & Herr, 2010), cats (Prilutsky et al., 2016), felines (Y. Kim et al., 2014), rats (Johnson et al., 2008), etc.

A correct representation of actuation and sensing is a fundamental element in each of these models. In terms of actuation, a

significant reference is the Hill's muscle model (Hill, 1938). Represented initially as a network of springs and compressors placed in series and parallel, it can reproduce both passive and active components of the muscle's behavior. It has been thoroughly studied and refined to enable efficient implementation or accurate responses (see Tsianos et al. (2012) or Haeufle et al. (2014) for recent examples). In most biomechanical models, sensory feedback also plays an essential role in obtaining accurate behaviors, as discussed in Rossignol et al. (2005). Beyond the action of vestibular, auditive, and visual feedback to modify a gait, simulation studies on cat locomotion have indicated that the locomotion process depends heavily on sensing Ground Reaction Forces (GRFs) in each separate leg (Ekeberg & Pearson, 2005), as well as the mechanical coupling between the legs, and the propagation delays between in the model (Murai et al., 2010). Hill's muscle and GRF feedback have been implemented in various forms in literature, and several adaptations have been built upon. These features are exploited as a source of inspiration in this dissertation to implement reflex-based locomotion on compliant robots, as detailed in Chapter 3.

2.1.2 Mechanical Compliance

In classical robotics and mechanical engineering, the vast majority of the systems are modeled using rigid-body dynamics. A rigid body structure consists of non-deformable body segments, with a single point mass, articulated together using joints that can have one or several Degrees Of Freedom (DOFs). Such systems can be described with the laws of kinematics combined with Newton's second law (dynamics) or with their derivative form (Lagrangian mechanics). This theoretical framework allows representing any system as a set of second-order ordinary differential equations, for which the analytical solutions have been thoroughly described in mathematics and physics. It makes also it possible to model forces due to stiffness and damping at the joint level.

In physics, the mechanical 'compliance' property (sometimes also called flexibility) of a body is defined as the inverse of stiff-

ness and typically measured in units of meters per Newtons (m/N). Compliance is generally linked to the elasticity modulus of a body. However, the modulus is an intensive property of the material whereas compliance, on the other hand, is an extensive property of the solid body.

In the last decades, an increasing interest has been brought to compliance in robotics. One reason can be found in the sharp limitations presented by a rigid body when it comes to complex tasks in a cluttered or unknown environment. Another motivation is the growing ability to predict the behavior of flexible parts efficiently using finite element methods in computer simulations. Depending on the scope of the studies, there are different definitions of a 'compliant mechanism' or a 'compliant robot', in opposition to a 'rigid' or 'stiff' robot. According to Edwards (2002):

> A compliant mechanism transfers or transforms motion, force or energy. Unlike rigid-link mechanisms, compliant mechanisms gain at least some of their mobility from the deflection of flexible members rather than from movable joints only.

Another definition is given in Kota & Ananthasuresh (1995):

> Compliant mechanisms are flexible structures that deliver the desired motion by undergoing elastic deformation, as opposed to the rigid body motions of conventional mechanisms.

According to this source, the definition encompasses the designs where elastic elements are used merely as springs, as well as those in which the primary source of motion is due to elastic deformation to achieve a wide variety of force-deflection characteristics with the least mechanical complexity. In other words, **the authors already distinguish two categories of 'compliant mechanisms': one where the systems are inspired by rigid-link mechanisms with extra springs and another where the different body segments are themselves subjects to deformation**. It is essential to note that, by computing equivalent rotational springs at the joint

level, the first category can be described using rigid-body dynamics theory whereas deformation of the bodies necessarily implies the use of numerical methods for solving partial differential equations in continuous space, such as the finite element method.

Another major concept in the field of compliant mechanics originates from control theory and the idea of 'impedance control' first detailed in Hogan (1984). This paper demonstrates the ability to simulate the effects of compliance, purely at a software level. The control software responsible for this achievement relies on two configuration parameters with the same dimensions of an equivalent rotational spring and damper. This process is named 'active compliance' in opposition to 'passive compliance' systems because it requires motor actuation and an external source of power. This drawback is counter-balanced by the aptitude to tune stiffness and damping parameters in software without having to modify the robot hardware.

Another notion that has emerged in the last decades is the 'soft robot' and 'soft robotics'. According to Trivedi et al. (2008):

> Soft robots have distributed deformation with theoretically an infinite number of DOFs. This leads to a hyper-redundant configuration space wherein the robot tip can attain every point in the three-dimensional workspace with an infinite number of robot shapes or configurations. Soft robots have an additional advantage over hard hyper-redundant robots: they generate little resistance to compressive forces and can conform to obstacles.

This definition presents the soft robots as an extreme configuration where the under-actuation and deformation abilities tend to the infinite. It also introduces the concept of redundancy or 'under-actuated mechanisms' that adds up to the compliance property. With such a definition, it is interesting to note that a soft robot cannot be modeled with traditional rigid body mechanics but require an approach such as finite element methods for an accurate

representation. Based on this understanding, I came up with a shorter description of soft robotics:

> Soft robots are mechanical systems with a continuous structure and featuring a hyper-redundant configuration space.

In this dissertation, I will use the following terminology to characterize a robot or a mechanical system:

- **Rigid robot**, for a robot including only non-deformable rigid parts with perfectly rigid joints (where all springs or dampers effects can be neglected) and with the same degree of actuation as freedom.
- **Active compliant robot**, for a rigid robot in which some motors are controlled in impedance and capable of rendering compliance properties using the appropriate software.
- **Passive compliant robot**, for a robot made of rigid and non-deformable parts only, but where the joints can be under-actuated and dissipate energy through spring and damper constraints.
- **Soft robot**, for a robot that has continuous deformable parts and cannot be modeled by the rigid body theory.

Hereafter, I present some of the most recent and relevant works in the field of locomotion focusing on the design, the implementation and the performance of passive and active compliance, as well as soft robots.

2.1.2.1 Active Compliance

Impedance control (Hogan, 1984) is the most common implementation of active compliance. The resulting accuracy and the capacity to deal with external impacts make it particularly **suitable for big-size quadrupeds**. Another useful property is the **ability to modify flexibility parameters during operation**. It allows reproducing more accurately natural observations, like the fact that the muscle stiffness of animals changes if the leg is in contact with the ground

(stance phase) or not (swing phase). However, impedance control assumes the use of **actuators that can tolerate high-torques** during a short amount of time. On the Big Dog (Raibert et al., 2008) and HyQ (Semini et al., 2011) robots, this has been handled using hydraulic actuators (Boaventura, Semini, et al., 2012). The control frequency is also determinant to establish the range where stable stiffness and damping can be accurately reproduced (Boaventura et al., 2013). Electric motors can also be used for active compliance if they remain limited to a restricted range of torques (Focchi et al., 2012). To handle this, SEAs (G. A. Pratt & Williamson, 1995) have also become a standard choice of actuators to mitigate large torques with elastic components coupled in series with electrical motors. This approach has proven very successful for legged systems (J. E. Pratt & Krupp, 2004) and has been recently used on the StarlETH (Hutter et al., 2012) and the ANYmal (Hutter et al., 2016) robots.

A different approach has also been suggested on the robots Cheetah-2 and Cheetah-3 robots (Seok et al., 2012; Bledt et al., 2018). To reduce high torques, these robots use a custom actuator and mechanical design based on proprioceptive control. Proprioception describes the ability to create an internal body representation from sensing reaction forces. In this scenario, the control primarily minimizes the GRF applied on the foot, therefore keeping the torque in a desirable range. Other solutions for compliant actuators using passive spring elements and active control to vary stiffness have been suggested in the MACCEPA (Van Ham et al., 2007), MACCEPA 2.0 (Vanderborght, Tsagarakis, et al., 2009), AMASC (Hurst & Rizzi, 2008), VSA (Bicchi & Tonietti, 2004), UT-SEA (Paine et al., 2014), SA-SEA (Isik et al., 2017), or DLR VS-Joint (Wolf & Hirzinger, 2008) designs. Recently, cheap designs have also emerged to enhance standard servo-motors with series elastic components and allow impedance control (Martins et al., 2015). In another direction, low-melting-point-alloy has also received some interest as a hybrid solution to control the stiffness properties during operation (Nakai et al., 2003; Schubert & Floreano, 2013).

2.1.2.2 Passive Compliance

The two major motivations in using a passive compliance system are the **safer interactions with humans and the environment** (Conde et al., 2004) and their **increased energetic performance** (Vanderborght, Van Ham, et al., 2009). With these considerations in mind, different designs of passive legs have been suggested to drive progress in robotic locomotion. The ability of a small quadruped with passive, compliant legs to produce robust bounding gaits was already demonstrated in Papadopoulos & Buehler (2000). The relation between the passive leg stiffness and the optimal locomotion speed of quadruped robots has been explored in Galloway et al. (2011). This corroborates the work of McMahon & Cheng (1990), where the authors show, using a simple mass-spring model cross-validated with experimental data from humans and kangaroos, that there exists an almost linear function between the forward locomotion velocity and the leg stiffness of a biped. Adding compliant elements to a mechanical system has another interesting consequence: reducing the Degrees Of Actuation (DOAs) whilst keeping the same DOFs. This effect has been exploited in robotic modular legs, that can track and modulate a bio-inspired locomotive cycle using only a single actuation DOF per leg (Saab et al., 2017). In Eckert et al. (2015), a comparison of different passive leg designs also confirms this result and discusses the influence on the robot's locomotion performance.

Besides the focus on the legs, the spine is another major element for mechanical passivity in locomotion. Its critical importance has been demonstrated in quadruped mammals (Y. Kim et al., 2014), and numerous robotic experiments have examined the effect of passive spines through their properties and control methods. These studies have investigated, among others, the mechanical design to produce a bounding gait (Duperret & Koditschek, 2017; Kawasaki et al., 2016), the energy efficiency (Chen et al., 2017; Khoramshahi et al., 2013), the influence on locomotion stability (C. Wang et al., 2017), or the role as a computational resource, using the morphological computation framework (Zhao et al., 2013). Numerous studies have also been carried in simulation to analyze the exact role of

the morphological parameters of the spine on the bounding gait (Culha & Saranli, 2011; Pouya et al., 2017; C. Wang & Wang, 2016; Deng et al., 2012) or gallop (Wei et al., 2015).

2.1.2.3 Soft Robots

Soft robotics is a growing field that encompasses a large number of different methods and applications. The core idea in soft robotics is to **inspire from biological structures and materials and encourage high deformability in robotic systems to achieve better results**, like reduced mass and energy consumption, simplified computation, natural movements, safe interactions, or adaptive behavior (S. Kim et al., 2013). Major applications in soft robotics involve locomotion (Calisti et al., 2017) and object grasping and manipulation (Hughes et al., 2016). In general, soft robotics can be achieved by using either soft material, soft sensors, soft actuators, or a combination of the three. In C. Lee et al. (2017), soft actuation methods are classified into three categories, including electroactive polymers (like, for instance, artificial muscles described in Miriyev et al. (2017)), fluidic actuation (including pneumatic and hydraulic systems), and length variable tendons (which includes tension cables and shape-memory alloys). It is important to note that the last two categories also overlap with the definitions of active and passive compliant actuators as stated above since there is no strong border between the different concepts. The control of soft robots have been achieved using embodiment and learning (Li et al., 2012), or by computing inverse kinematics through finite element methods (Bieze et al., 2018)

2.1.3 Quadruped Robots

Various quadruped robots featuring different compliance strategies or leg mechanisms have been designed in the last decades. Hereafter, I discuss some of the existing platforms in the literature to understand how they fit the research in bio-inspired compliant locomotion. I have tried to summarize them in Table 2.1 and Table

2.2, in relation with some crucial parameters: robots mass, length, leg mechanisms, DOFs, DOAs, and compliance. Besides, the scatter plot in Figure 2.1 provides a visual projection of this data based on four criteria: compliance level, length, weight, and DOAs/DOFs ratio.

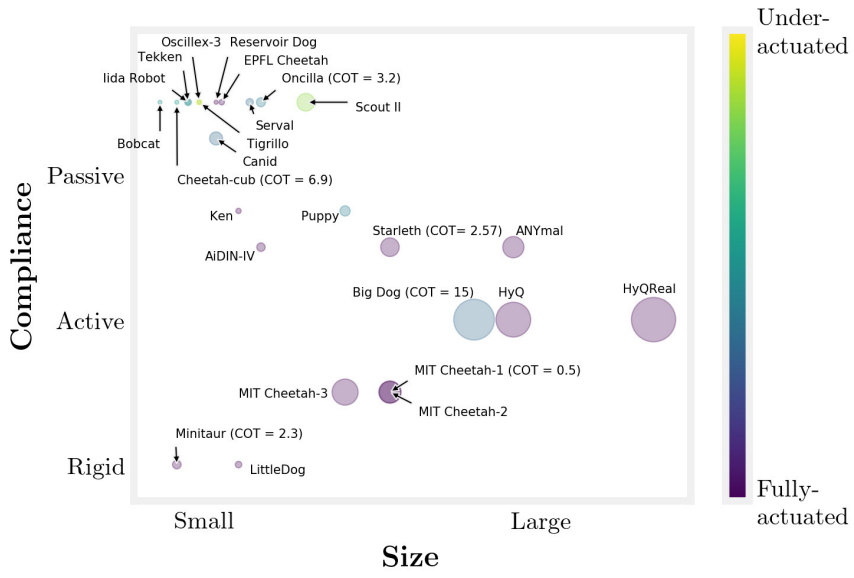


Figure 2.1: This figure presents the quadruped robots mentioned in Tables 2.1 and 2.2 in structured manner. The X-axis corresponds to the robot length and the Y-axis, the compliance degree, from rigid to passive systems. The size of the patches corresponds to the robot weight, and their color describes the ratio between DOAs and DOFs. In this review, large robots tend to make more use of active compliance strategies whereas small robots generally rely on passive, compliant systems, or simply include rigid segments (Minitaur, LittleDog).

2.1.3.1 Large Sizes

Although it has not been thoroughly documented in scientific journals, the Big Dog robot (Raibert et al., 2008) released in 2008 by *Boston Dynamics* has demonstrated the feasibility of implementing and controlling a large quadruped robot in real use cases. Big

Dog has a length of 91 cm for a height of 76 cm, it weighs 110 kg and is capable of trotting up to 6.4 km/h, lifting charges of 150 kg, and climbing slopes of 35.3 degrees. The use of hydraulic actuation provides sufficient torque and tolerates the high impacts encountered in dynamic locomotion, for a body of that size and weight. Its success encouraged new research projects in the same field across the world. Based on the same actuation principle, the robot HyQ was developed at the *Italian Institute of Technology (IIT)* (Semini et al., 2011). Eight of the twelve degrees of freedom of the robots are actuated using hydraulic cylinders and the last four with electrical motors. HyQ has roughly the same dimensions as Big Dog, with a length of 1 m, a width of 50 cm, and a height of 98 cm. It weighs 80 kg but does not embed the hydraulic pump in contrast to a newer and heavier version, called HyQReal, which is completely autonomous.

Series Elastic Actuators (SEA) have been introduced in G. A. Pratt & Williamson (1995) as a means to achieve actuation at high torques using passive, compliant elements embedded with the motor for efficient energy storage and release. This principle has been employed in the StarlETH robot, from *ETH Zürich* (Hutter et al., 2012). With its length of 71 cm length and width of 64 cm for a weight of 23 kg, this robot is slightly smaller than the two previously mentioned. It has been the main source of inspiration for a larger robot of 30 kg dedicated to rescuing situations called ANYMal (Hutter et al., 2016). In parallel, the *Massachusetts Institute of Technology* has grown some experience in a different type of motor proprioceptive control on a leg level to mitigate the compliance issue. Cheetah-1 (Seok et al., 2013), Cheetah-2 (Park et al., 2017), and Cheetah-3 (Bledt et al., 2018) are successive versions of a robot using proprioceptive actuation. With a respective weight of 33 kg, 31 kg, and 45 kg, they can achieve good energetic performance in dynamic gaits with a low Cost Of Transport (COT), down to 0.45 for Cheetah-3. More detail about the COT is given in Section 3.6.2.4.

	Quadruped Robot	l (m)	m (kg)
1	**Big Dog** (Raibert et al., 2008)	0.91	110.00
2	**HyQ** (Semini et al., 2011)	1.00	80.00
3	**HyQ Real** (Villarreal, Barasuol, Wensing, & Semini, 2019)	1.33	130.00
4	**Starleth** (Hutter et al., 2012)	0.71	23.00
5	**ANYMal** (Hutter et al., 2016)	1.00	30.00
6	**MIT Cheetah-1** (Seok et al., 2013)	0.70	33.00
7	**MIT Cheetah-2** (Park et al., 2017)	0.70	31.00
8	**MIT Cheetah-3** (Bledt et al., 2018)	0.60	45.00
9	**Puppy** (Aschenbeck et al., 2006)	0.60	7.00
10	**Ken** (Narioka et al., 2012)	0.35	1.90
11	**Reservoir Dog** (wyffels et al., 2010)	0.29	1.10
12	**Scout II** (Poulakakis et al., 2005)	0.50	20.80
13	**Oscillex-3** (Owaki & Ishiguro, 2017)	0.25	2.00
14	**Tigrillo** (Willems et al., 2017)	0.25	1.00
15	**Canid** (Pusey et al., 2013)	0.29	11.30
16	**EPFL Cheetah** (Tuleu et al., 2011)	0.30	2.00
17	**Cheetah-cub** (A. Spröwitz et al., 2013)	0.21	1.10
18	**Oncilla** (A. T. Spröwitz et al., 2018)	0.40	5.50
19	**Bobcat** (Sprowitzt et al., 2013)	0.17	1.03
20	**Serval** (Eckert et al., 2018)	0.38	3.56
21	**LittleDog** (Murphy et al., 2011)	0.34	2.85
22	**Minitaur** (Kenneally et al., 2016)	0.20	5.00
23	**Tekken** (Fukuoka & Kimura, 2009)	0.23	3.10
24	**AiDIN-IV** (Y. H. Lee et al., 2017)	0.40	4.80
25	**Iida Robot** (Iida & Pfeifer, 2004)	0.17	1.5

Table 2.1: This table presents the different quadruped robots reviewed in this dissertation along with their size (in meters) and weight (in kilograms). Data from this table is organized on a 2D plane in Figure 2.1.

	Spine	Leg	Seg	DOF	DOA	Source of Compliance
1	No	Open	3	20	16	Active (hydraulic)
2	No	Open	2	12	12	Active (hydraulic)
3	No	Open	2	12	12	Active (hydraulic)
4	No	Open	2	12	12	Active (SEA)
5	No	Open	2	12	12	Active (SEA)
6	Yes	Pantograph	3	9	9	Active (proprioceptive)
7	No	Pantograph	3	12	12	Active (proprioceptive)
8	No	Open	2	12	12	Active (proprioceptive)
9	No	Open	3	16	12	Active (pneumatic)
10	Yes	Pantograph	4	13	13	Active (pneumatic)
11	No	Pantograph	3	8	8	Passive (spring-damper)
12	No	Prismatic	1	8	4	Passive (spring-damper)
13	No	Prismatic	1	10	4	Passive (spring-damper)
14	No	Open	2	8	4	Passive (spring-damper)
15	No	Prismatic	1	5	4	Passive (segment)
16	No	Pantograph	3	8	8	Passive (spring-damper)
17	No	Pantograph	4	12	8	Passive (spring-damper)
18	No	Pantograph	4	16	12	Passive (spring-damper)
19	Yes	Open	3	13	9	Passive (spring-damper)
20	No	Pantograph	4	19	15	Passive (spring-damper)
21	No	Open	2	12	12	Rigid
22	No	Closed	5	8	8	Rigid
23	No	Open	3	16	12	Passive (spring-damper)
24	No	Open	2	8	8	Active (SEA)
25	No	Open	3	12	8	Passive (spring-damper)

Table 2.2: This table is the follow-up of Table 2.1. The first column indicates whether the robots include a spine joint. The second characterizes the leg mechanism used on the robot. The third column gives the number of segments in each leg. The next two columns define the total number of Degrees of Freedom (DOFs) and Actuation (DOAs) for the robot. The last column gives some insights into the main source of compliance used on the robot.

2.1.3.2 Small Sizes

With a lower price and a reduced need for tools and components to build and maintain the hardware, small quadruped robots have been mainly employed for fundamental research on locomotion. It is impossible to provide an exhaustive list of the different designs realized in the last decade, but I give here a small overview of the most influential works regarding this dissertation. On one hand, small quadruped robots have been used to study new types of bio-inspired controllers. The puppy (Aschenbeck et al., 2006) and the Ken (Narioka et al., 2012) robots are both good illustrations in this field. They have been mainly used to study the use of air-driven muscles and actuators to implement compliant locomotion control. Inspired by the concept of morphological computation, the passively compliant robot Reservoir Dog (wyffels et al., 2010) has been developed to test Physical Reservoir Computing (PRC) in locomotion tasks.

On the other hand, small quadruped robots can be used to test new mechanical solutions to test and improve leg designs in terms of stability, speed, passivity, etc. A recent review on the subject can be found in Zhong et al. (2019). The Scout II robot is a typical example of a robot with prismatic compliant legs (Poulakakis et al., 2005). Using a single actuator in the hip and a linear spring-damper system in the knee, it has been used to study a bounding gait. In a more recent study, the robot Oscillex-3 was able to demonstrate different gaits and gait transitions, with a compliant prismatic mechanism using a single actuator achieving realistic foot cycles (Owaki & Ishiguro, 2017). Leg actuation using only one motor has also been achieved on the passive, compliant robot Tigrillo (Willems et al., 2017), by including under-actuated joints or in Canid (Pusey et al., 2013), a robot with a compliant spine and where each leg is physically implemented with a single flexible segment. Tigrillo is itself inspired by a quadruped robot described in (Iida & Pfeifer, 2004). In more complex designs, the benefits of compliant pantograph legs have been investigated in the *EPFL Biorobotics Laboratory* with the conception of the small quadruped robots Cheetah (Tuleu et al., 2011) and Cheetah-cub (A. Spröwitz

et al., 2013). The Oncilla platform (A. T. Spröwitz et al., 2018) suggested improvements by adding an extra actuated Hip Adduction and Abduction (HAA) joint to test efficient turning strategies.

Whilst these robots are making use of passive, compliant elements, the design of active compliant legs has also been studied on middle-size robots with the Tekken (Fukuoka & Kimura, 2009) and the AiDIN-IV (Y. H. Lee et al., 2017) robots. The implementation of an active spinal joint was also considered on the Bobcat robot (Sprowitzt et al., 2013) or, more recently, the Serval robot (Eckert et al., 2018). Finally, despite their lack of compliant elements, it is worth mentioning the LittleDog robot (Murphy et al., 2011), which has been a major platform to study machine learning algorithms in the real world. More recently, the Minitaur robot has also played a role in this field (Kenneally et al., 2016).

2.2 Control Architectures and Methods

This section provides a review of the most common methods in quadruped robot locomotion control. I classify them into five categories: (1) classical approaches built upon automation theory, (2) decentralized control methods which are generally used for dynamic gaits like trot and gallop, (3) centralized control, exemplified by the use of central pattern generators, (4) data-driven methods in which I discuss two machine learning techniques, and (5) evolutionary robotics where the co-optimization of morphology and control is taken under consideration. In each category, the biological inspiration is also introduced, except for the first one which is strongly based on engineering methods. A summary of the different control methods is presented in Figure 2.2. The control methods are organized in this Figure against three criteria: the system complexity that can be handled, the type of model, and the biological plausibility. This illustration is of course symbolic since it is difficult to draw a clear boundary between the different control methods.

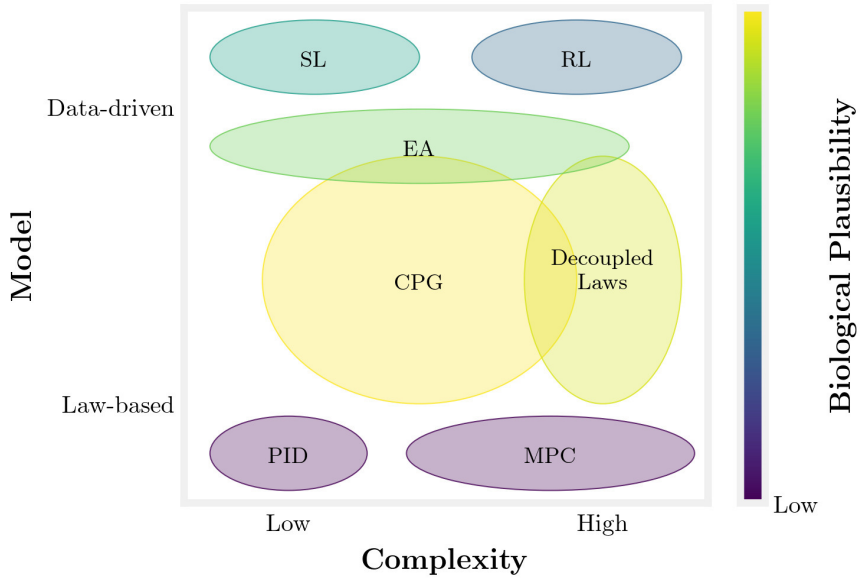


Figure 2.2: This figure gives an approximation of the domain of application for different control methods. The X-axis represents the complexity of the system under control. The Y-axis shows if the inherent model of the method is rather based on mechanical laws or training data. The color estimates biological plausibility. As they cover different aspects of the control, the combination of different methods is also common in literature, which partly explains the overlap.

2.2.1 Classical Rigid Control

2.2.1.1 Proportional Integral Derivative Controller

Since its very early formulation, control theory has made considerable improvements and generalized in practically every automated system (Fernández Cara & Zuazua Iriondo, 2003). **However, the main building block of control theory, the Proportional Integral Derivative (PID) controller, is not well suited for complex systems with many DOFs requiring coordination** (for a theoretical explanation, see Sections 3.1 and 3.2). Also, tuning the settings of automatic controllers is not an easy task and is highly sensitive to the system under consideration (J. G. Ziegler & Nichols, 1942). Two

approaches built upon control theory in the 1970s and 1980s have been widely used in robotic locomotion ever since: Model Predictive Control (MPC) (Richalet et al., 1978) which is specifically suited to static gaits, and decoupled stability and motion laws, which are mainly used with dynamic gaits.

2.2.1.2 Model Predictive Control

MPC is a popular method to control complex multivariate processes for which we have a detailed analytical model. It has been successfully used to control legged robots, helicopters, drones, or all sorts of wheeled vehicles. **MPC relies on an internal representation of the robot model to predict its evolution in time and space and determine the optimal actuation vector for a certain motion goal**. It can be an essential element to solve whole-body control in robotics, i.e. solving the full dynamics of all joints and links to determine the optimal control strategy. Illustrative works on quadruped robots include the crawling gait of the HyQ robot (Focchi et al., 2020) and parameter optimization for multiple gaits on the LittleDog robot (Kalakrishnan et al., 2011). There are two major drawbacks to this technique. Firstly, an optimization process needs to run at each timestep, which requires large computational resources. To run in real-time, the problem is generally constrained (Farshidian et al., 2017). This is generally solved by equilibrating the Zero Moment Point (ZMP) (Vukobratovi & Borovac, 2004) and keeping the projection of the Center Of Gravity (COG) inside the polygon formed by the feet in contact with the ground. Nonetheless, recent examples have pushed the limits of the method to dynamic gait control (D. Kim et al., 2019; Grandia et al., 2019). Secondly, the need for an internal model causes this approach to generalize badly to soft and compliant robots although different tracks for improvements have been studied. For instance, in Fahmi et al. (2019), the authors demonstrate locomotion of rigid robots on soft terrain by adding soft constraints in the equations modeling the interaction of the robot with the ground. In Villarreal, Barasuol, Wensing, & Semini (2019), the authors combine a convolutional neural network to MPC to enhance the model of the environment with visual perception.

2.2.2 Decentralized Dynamic Control

2.2.2.1 Biological Inspiration

Despite its success in robotics, the practical implementation of MPC is not directly linked to the real locomotion process observed in biology. Within the vertebrates, **the control of locomotion generally happens both at the musculoskeletal level and in the central nervous system, especially in the spinal cord and the cerebellum.**

The fundamental role played by the compliant musculoskeletal system in control itself has been emphasized in Geyer & Herr (2010), where an accurate model of the human legs coupled with local muscle reflexes was able to produce realistic patterns of walking without any form of global control. It has been shown that the model could also handle disturbances and small obstacles by naturally generating recovery foot trajectories (Murai & Yamane, 2011). Further demonstrations have shown that **pure-reflex control without centralized generation of rhythmic signals, could reproduce realistic and stable gait patterns in bipeds and quadrupeds using accurate musculoskeletal models** (Geijtenbeek et al., 2013; Song & Geyer, 2015).

On top of this, the localization of a descending pathway from the brainstem to the spinal cord and its functional role during locomotion have been documented in cats since 1980 (Steeves & Jordan, 1980) and confirmed in various studies (Jahn et al., 2008). The role of the cerebellum is complex and diverse, and many models have been suggested to capture its essence (Ito, 2006). During locomotion, it has been advanced that this organ implements different functions, including the initiation of locomotion patterns (Jordan et al., 2008), the modification of gait and posture on uneven terrain (Drew et al., 2004), or the regulation of interlimb coordination and gait transitions (Danner et al., 2016). Cerebellar lesions on walking cats have demonstrated the active role of the cerebellum by showing abnormal timing of relative limb movements, reduced amplitude in different joints, and decreased stride lengths (Yu &

Eidelberg, 1983). In Morton & Bastian (2004), the authors indicate that the medial zone and the flocculonodular lobe in the human cerebellum influence the control of extensor muscles to maintain correct balance and a proper stance, and modulate the rhythm of locomotion patterns. Experiments with patients affected by cerebellar ataxia also demonstrated a decreased stability of the trunk's center of mass due to a deteriorated stance in the presence of lateral and backward disturbance during locomotion compared to healthy subjects (Bakker et al., 2006). Similar results have pointed out the role of the vestibular feedback provided by the inner ear in this process (Borel et al., 2004). More details on the adaptive role of the cerebellum in locomotion have been provided in Morton & Bastian (2006). In this paper, experiments on patients with ataxia walking on a split-belt treadmill demonstrated that cerebellar impairment did not decrease reactive feedback-driven adjustments, but significantly damaged predictive feedforward motor adaptations. This evidence supports the hypothesis that **the cerebellum helps during locomotion in predicting the limb movements using a stored internal representation with spatial and temporal components.**

2.2.2.2 In Robotics

Based the biological observations stated in the previous section, robotic control strategies have suggested the use of multiple decoupled laws to control different aspects of the gait simultaneously. The core idea is to **dissociate the generation of the foot trajectory, ideally periodic with a constant amplitude, the control of the speed, requested by the user, and the balance, ensured using inertial corrections** (Raibert, 1986). The sharp separation between posture control, dynamically corrected through stabilization torques, and foot trajectory, governed by kinematics equations to achieve a constant locomotive cycle, has also been employed in other state-of-the-art quadruped robots (Papadopoulos & Buehler, 2000; Barasuol et al., 2013; Park et al., 2017). Many improvements have been added to make this technique more robust to external disturbance and on rough terrains. Among others, dealing

with small obstacles has been overcome using local proprioceptive reflexes (Focchi et al., 2013) or vision-based corrections of foot trajectory (Barasuol et al., 2015; Villarreal, Barasuol, Camurri, et al., 2019).

2.2.3 Centralized Control

2.2.3.1 Biological Inspiration

Notwithstanding the ability to realize biological musculoskeletal models controlled by reflexes only, **the production of centralized rhythmic patterns, called Central Pattern Generators (CPGs), that do not require sensory information was clearly identified in the spinal cord of quadruped mammals** more than a century ago (Brown, 1911). Grillner & Wallen (1985) have extended this claim to most of the vertebrate family. Detailed physiological and functional studies have also asserted the presence of CPG in fish and amphibians like the lamprey (Sigvardt & Williams, 1992) or the salamander (Delvolvé et al., 1999). However, the existence of CPGs and their implication in locomotion has yet not be proven in humans (Minassian et al., 2017). A reasonable hypothesis claims that, despite a physical presence, their role has decreased with the evolution toward biped walking, which relies more on reflexes (i.e., sensory feedback from the hind limbs) and stability control (i.e., vestibular feedback from the brainstem). The interaction of CPGs with a reflex-based neuromuscular locomotion model has been emphasized in Rossignol et al. (2005) and extensions of the model implemented in Geyer & Herr (2010) have demonstrated the ability to combine CPGs with sensory feedback (Dzeladini et al., 2014). On the other hand, the connections between the CPG and vestibulospinal neurons have been described in various studies and simulations (Orlovsky, 1972; Danner et al., 2016; Fukui et al., 2019).

Various approaches with different levels of granularity and accuracy exist to represent biological CPGs. The first model has been introduced in Brown (1911) and consists of two coupled populations of neurons, linked with inhibitory connections. Matsuoka oscillatory models have refined this approach and developed the

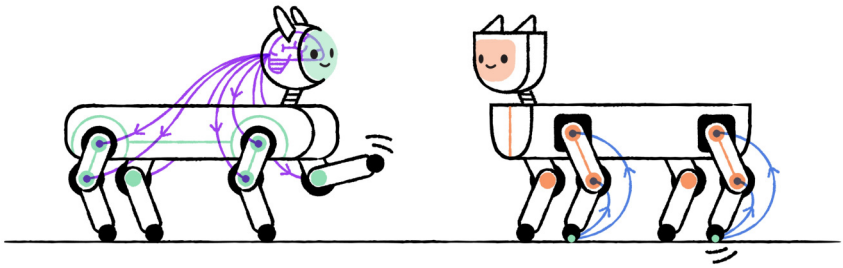


Figure 2.3: The principles and plausibility of centralized (left) versus reflex-based (right) locomotion have been thoroughly discussed in biology (Grillner, 1975). However, in-silico applications are still rare in the domain and raise questions on how to integrate the feedback to centralized control or how to efficiently coordinate and stabilize reflex-based control.

mathematical aspects in Matsuoka (1985). More recently, wyffels & Schrauwen (2009) discussed how artificial Recurrent Neural Network (RNN) could be organized using Reservoir Computing (RC) and trained to produce CPG trajectories. In the last generation of CPG, this approach has been transferred to Spiking Neural Networks (SNN) and successfully applied in locomotion tasks with quadrupeds (Vandesompele, Urbain, wyffels, & Dambre, 2019) and hexapods robots (Espinal et al., 2016).

2.2.3.2 In Robotics

Based on these observations, work on salamander robots showed practical applications of CPGs to produce natural locomotion gaits on the ground and in the water (Ijspeert, 2008). This has been a fundamental point for the generalization of the CPGs in robotics. **CPGs have been successfully used to control bio-inspired robots** such as quadrupeds (Fukuoka et al., 2003; Tuleu et al., 2011), hexapods (Dasgupta et al., 2015), bipeds (Righetti & Ijspeert, 2006; Nassour et al., 2014), amphibians with ability to walk and swim (Ijspeert et al., 2005) or marine turtles (Seo et al., 2010).

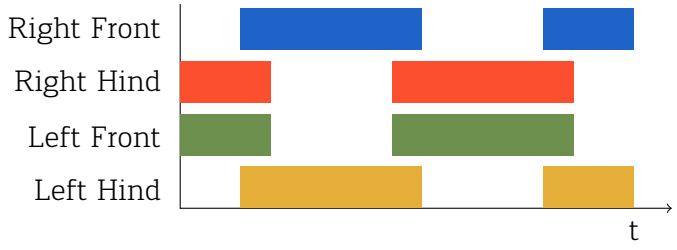
The success of CPGs is also due to the strong mathematical framework employed to describe their dynamics. Buchli, Righetti, &

Ijspeert (2006) provide a review of equations for different CPG types according to their dynamical properties. In Ajallooeian et al. (2013), it is discussed how CPGs can be generalized to produce arbitrary rhythmic trajectories and Degallier & Ijspeert (2010) review how CPG motor primitives can combine with force fields to generate both discrete and rhythmic patterns. Adding feedback on CPGs for closed-loop control has also exhibited results to regulate the gait or adapt it to rough environments (Righetti & Ijspeert, 2008). In Gay et al. (2013), optimization methods are used to compute this feedback for efficiently walking on a slope and in the presence of external disturbance.

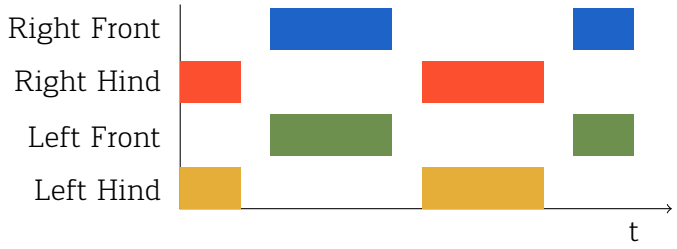
2.2.3.3 Interlimb Coordination

Legged robots generally possess a large number of degrees of actuation, and efficient interlimb coordination is, therefore, a key element in a good locomotion system. A review of different studies concerning legged robot coordination has been discussed in Aoi et al. (2017). In Figure 2.4, I represented the most common patterns for walking, trotting, and galloping in quadruped mammals. Walking is generally named a 'static gait'. This motion guarantees a stable posture at all times during the locomotion process, i.e. the COG of the body is always included in the support polygon of the contacts between the feet and the ground. On the other hand, trotting and galloping are described as 'dynamic gaits', because they rely on the position of the COG in front of the contact points to create torque and move the body forward. The timing is, therefore, extremely important to avoid falling. Incidentally, different studies have demonstrated that realistic coordination modes at various regimes can emerge spontaneously from the interaction between the body and the environment with no prior knowledge on the gait patterns (Owaki et al., 2012; Owaki & Ishiguro, 2017). This effect is strongly linked to the speed and energy consumption, as demonstrated in Xi et al. (2016) and Fukui et al. (2019). The same conclusions have also been observed in simulation and on real quadrupeds (Danner et al., 2016).

(a) Walking gait



(b) Trotting gait



(c) Gallopping gait

Figure 2.4: Typical foot patterns of the cat during locomotion as described in Pearson (1976). The plain bars represent the stance phases and the gaps, the swing phases. During walking, each foot is lifting consecutively while there are always at least three supports in contact with the ground. During trotting, the two diagonally opposite feet are supporting the body weight alternatively. Gallop involves a rocking movement between the front and the hind legs.

2.2.4 Learning-Based and Data-Driven Control

Since the 2010s, machine learning, and more especially the subfield of deep learning, have gained a growing interest in many applications, among which neuroscience and robotics are no exception (Hassabis et al., 2017). **In robotics, machine learning has created new perspectives for more efficient and agile controllers, but this approach also presents serious limitations, in particular, because of a lack of interpretability after training** (Sünderhauf et al., 2018). The most widespread technique used for machine learning is the Artificial Neural Network (ANN). In this section, I will quickly introduce the ANNs before presenting some influential work of the last years concerning quadruped robotic locomotion in Supervised Learning (SL) and Reinforcement Learning (RL). Subsequently, I will define the sim-to-real gap and introduce some research efforts carried in this direction. A few recent works toward more biologically plausible networks, like the Spiking Neural Networks (SNN), will be presented at the end of this section.

2.2.4.1 Biological Inspiration

ANNs are one of the most popular machine learning models as they can process complex, nonlinear relationships between a set of inputs and outputs. They are loosely inspired by the brain structure and consist of inter-connected networks of simple computational entities, called neurons. In the most generic form, an individual artificial neuron, as presented in Figure 2.5, is a simple mathematical model that takes a series of input signals and multiplies them by the corresponding connection weights. This weighted sum is subsequently processed through an 'activation function'. Historically, the sigmoid has been a popular choice for the activation function, as it combines a simple mathematical expression with a certain biological plausibility. Electroencephalographic experiments demonstrated that this function can be understood in terms of the variance or dispersion of states in neuronal populations (Marreiros et al., 2008). However, the Rectified Linear Unit (ReLU) function has generalized rapidly for its even higher simplicity and

its ability to avoid vanishing gradient in deep networks (i.e. with numerous layers). Its increased plausibility has been also claimed in Glorot et al. (2011). The mathematical framework used to model ANNs is mainly based on the concept of 'perceptron' developed in the 1950s, and more broadly the kernel methods, widely used in machine learning.

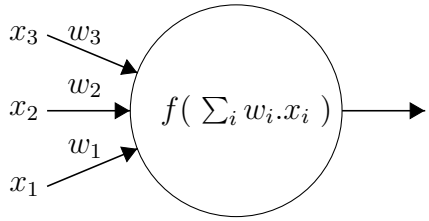


Figure 2.5: An artificial neuron performs a simple mathematical function. On this diagram, w_i represent the weights of the input connections and x_i the input states. f is called the activation function.

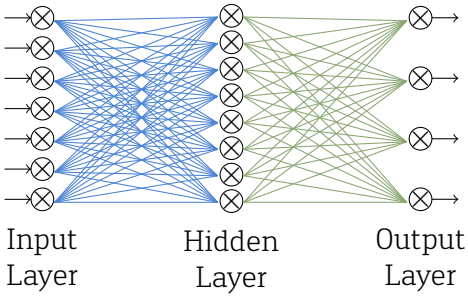


Figure 2.6: A feed-forward ANN is an assembly of successive neural layers which are sparsely or fully connected together, inspired from the biological visual cortex (Van Essen & Felleman, 1991). It has become a popular computational unit to solve complex problems in the last decade.

The architectures of ANNs can be divided into two types: feed-forward and recurrent networks. In the first category, there is no feedback loop in the connection pattern between the neurons. This spatial organization can be related to the brain connections in the

visual and inferior temporal cortex and has strongly contributed to inspire the design of Deep Learning (DL) networks (Y. A. LeCun et al., 2012). This efficient architecture, together with the development of the back-propagation algorithms for efficient learning (Y. LeCun et al., 2015), established this framework as a leading technique in machine learning. The second category of ANN includes feedback connections between neurons, hence their name: Recurrent Neural Networks (RNN). Despite the high plausibility of their spatial distribution, these networks have found less practical applications in robotics due to their training complexity. Nevertheless, the memory instantiated by the connection loops makes them particularly suited to process temporal sequences.

2.2.4.2 Supervised Learning

Supervised Learning (SL) embraces the **learning tasks where the goal is to infer a mapping function between a set of inputs and outputs by relying on example pairs**. As illustrated in Figure 2.7, SL relies on example outputs, called 'target signals', to estimate an optimal mapping function. In robotics, this target signal is often constituted by the motor outputs, which can be obtained from human demonstration, evolutionary optimization, or simulation. Among the works involving SL in quadruped robotics, the hierarchical apprenticeship learning conducted on the LittleDog robot exemplifies how expert trajectories can lead to learning a complex set of motor commands for locomotion in rough environments (Kolter et al., 2007). SL has also found some applications in the field of 'imitation learning' like illustrated by the work of Z. Wang et al. (2017) on biped locomotion. Imitation learning directly derives from SL, with the most significant difference being that predictions are made sequentially. It is often preferred over RL when the reward is sparse and hard to optimize.

2.2.4.3 Reinforcement Learning

According to a prominent book on RL (Sutton & Barto, 2013), this concept can be defined as:

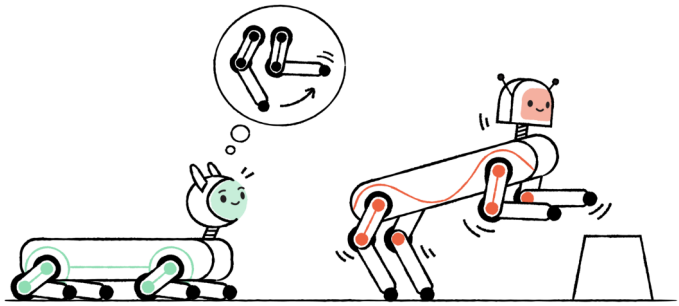


Figure 2.7: SL in robotics can be used to map sensor inputs to motor commands when the optimal target signals are already known. These signals can be obtained from a task demonstration, for instance. Among the other options, they can also be produced through simulation or after optimization of parametric models.

> A computational approach to understanding and automating goal-directed learning and decision-making. It is distinguished from other computational approaches by its emphasis on learning by an agent from direct interaction with its environment, without relying on exemplary supervision or complete models of the environment.

In robotics, the agent mentioned in the definition is generally the robot itself, which relies on sensory information and computes motor commands to perform a specific task like locomotion.

This framework has been put in the spotlight recently when deep RL succeeded in achieving high scores on old Atari games, with no prior information (Mnih et al., 2013, 2015). Although it initially focused on environments with discrete state and action space, RL has also successfully generalized to continuous control (Doya, 2000; Duan et al., 2016). This way, a method like deep Q-learning, initially used to solve the Atari games problem, has been extended to tasks in continuous time and space (Lillicrap et al., 2015). This approach **demonstrated successful results for end-to-end con-**

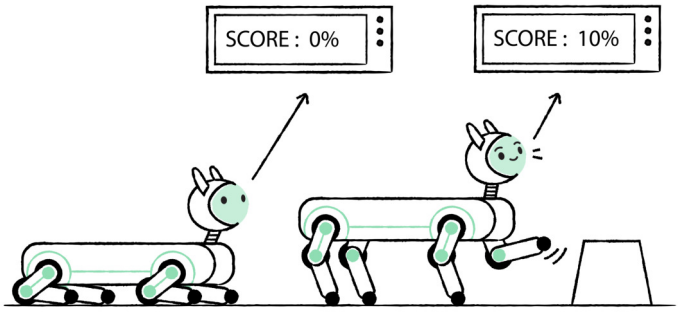


Figure 2.8: In RL in robotics, an agent learns to optimize a policy, generally mapping sensory inputs to actuation output through an iterative process where only a reward is provided to the algorithm. In contrast with SL, where the target signals are provided, RL only uses this sparse reward to learn the policy.

trol (i.e., from sensor inputs to motor commands) on real robotic arms (Levine et al., 2001), tensegrity structures (Zhang et al., 2017), or **for locomotion of biped robots** (Xie et al., 2018). Based on these results, major algorithms like Trust Region Policy Optimization (Schulman et al., 2015) or Soft Actor-Critic (Haarnoja et al., 2018) were also designed specifically for robotic applications.

Continuous end-to-end control is not the only domain in which robotics research has been performed. It has also been successfully applied in quadruped robotics to tune a parametric controller (Kohl & Stone, 2004; Shen et al., 2012) or to optimize some subsystems of the controller to make the LittleDog robot walk on challenging terrains (Kalakrishnan et al., 2011). In addition, combining continuous motor primitives using a higher-level system optimized with RL demonstrated a better transfer of control policies in different environments in simulation (Heess et al., 2016). The same concept has confirmed potential applications to enable a transfer from simulation to real quadruped platforms (Singla et al., 2019).

However, **most of the research about deep RL for locomotion is generally conducted in simulation** (Peng et al., 2017, 2016; Heess et al., 2017). The main reason behind this choice is the sparsity of the rewards employed in RL techniques. This makes the learning phase especially data-intensive and hard to conduct on a real robot, which is prone to wearing and breaking. Another clear drawback of RL methods is their low reproducibility (Henderson et al., 2018). Two reasons are pointed out to explain this phenomenon. First, the convergence of RL algorithms strongly depends on the initial conditions and the randomness of the problem. This calls for a good statistical analysis of the results for each new algorithm. Secondly, the research on new RL algorithms is often biased toward overfitting on a specific application, and they generalize badly to a larger set of problems.

2.2.4.4 Sim-to-real Gap

The vast majority of the studies involving machine learning and robotic are conducted in simulation. Indeed, direct learning on a robot presents several drawbacks, among which the training time (which can be generally sped up in a physics simulation), the wearing and hysteresis of mechanical parts, but also potential damages when the robot is exploring its own motor control capacities. **Partial training of policies in simulation is a straightforward idea to obtain a good initial state and avoid mechanical damage before transfer to a real robot. However, small variations between simulation and reality can lead to significant performance deterioration or failure in the real environment**. This issue is commonly known as the **'sim-to-real gap'**. To overcome this, three types of strategies have been investigated: relying on an accurate simulator with a calibrated model, randomizing the simulation, or learning the real robot dynamics.

The first approach is certainly the most conventional and relies on realistic simulations combined with important efforts on calibrating the models to reduce the accuracy differences with reality. A major drawback is the length of the calibration procedure, which

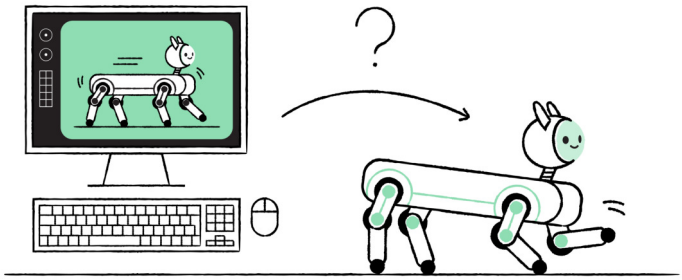


Figure 2.9: Despite the progress in data-based control for locomotion, most of the works are conducted in simulation and then transfer to a real robotic platform. This process can easily become an issue and decrease the final performance.

is also not immune to inaccuracies that would lead to large differences when learning control policies. Also, it shifts the problem towards a sophisticated implementation of the physics engine and the need for heavier computational resources. Another major drawback of this method concerns the under-actuated, compliant, or soft robots where an accurate physical model can be hardly provided. An illustrative example of a calibration procedure in classical robotics is given in Frigerio et al. (2017) for the HyQ robot and its simulation model in ROS gazebo.

The second approach relies on simulation randomization. A control policy can be transferred to a real robot after training in simulation to reduce the learning time. For instance, Degrave, Burm, et al. (2015) have demonstrated on the Oncilla robot that a control policy for quadruped locomotion can be trained partly in simulation and then transferred on the real robot to continue the training. To avoid this last step while guaranteeing robustness to potential inaccuracies, research in robotics has focused on randomizing certain aspects of the simulation. Adding noise to the sensors' observations has been considered positively to reduce the sim-to-real gap in Dosovitskiy & Koltun (2016). Randomization can also be added in the control policy itself, as investigated using progressive networks for vision and manipulation tasks (Rusu et al., 2016). However, the most fruitful approach consists of randomizing the

simulation dynamics directly. This idea was applied successfully on wheeled robot control using Evolutionary Algorithms (EA) in Jakobi et al. (1995). More recently, it achieved some valuable results on the stiff quadruped robot Minitaur (Tan et al., 2018) or in manipulation tasks (Peng et al., 2018). Furthermore, it has also shown substantial benefits in compliant robot locomotion by randomizing morphology parameters of the Tigrillo robot simulation model (Vandesompele, Urbain, Mahmud, et al., 2019).

The third category of methods to bridge the sim-to-real gap takes a different perspective. Instead of reducing the inaccuracies using sophisticated simulation models or flexible control policies, it relies on learning the robot dynamics from examples with no prior knowledge. In Martius & Lampert (2016), for instance, it is discussed how a neural network could be used to learn different sets of equations and, therefore, reliably model physics simulations based on example data from the real world. In (Hwangbo et al., 2019), the authors discuss the training of a neural network to map the complex compliant dynamics of the robot ANYmal and facilitate the transfer of policies from simulation to the real quadruped. This last category will be further detailed for the Tigrillo robot in Chapter 5.

2.2.4.5 Towards Realistic Neural Networks

Using more realistic neuronal models, accounting for the complex time dynamics observed in the biological brains, has been discussed as a potential path for developments in machine learning (Maass, 1997). These types of networks are generally grouped within the denomination Spiking Neural Networks (SNN). Computing with SNNs is based on a vast modeling effort from the theoretical neuroscience community (Paugam-Moisy & Bohte, 2012). The first scientific model of a neuron, the Hodgkin-Huxley model, has been suggested in Hodgkin & Huxley (1952) to describe the electrophysiology of a typical brain cell. An essential improvement has been carried in the integrate-and-fire model, which also takes the synaptic connections into consideration (Burkitt, 2006). Different neuronal models have been suggested since, to combine

the accuracy of Hodgkin-Huxley models with the efficiency of integrate-and-fire models (Izhikevich, 2003). Yet an important performance leap was brought by simulation of neural populations, where a statistical approach takes the effect of a group of neurons directly into consideration. This drastically simplifies the computation while keeping a satisfying degree of accuracy for many applications (Omurtag et al., 2000). The usage of SNNs has popularized through public implementations of these models in simulators like *NEURON* (Hines & Carnevale, 1997) or *NEST* (Gewaltig & Diesmann, 2007), and common description languages like *PyNN* (Davison et al., 2008). In recent years, dedicated hardware has also been developed to simulate these models in a computation- and energy-efficient way. These electronic devices generally rely on analog neural models like *BrainScaleS* (Schemmel et al., 2010), or adapted multi-processing architectures like in *SpiNNaker* (Furber et al., 2013), *TrueNorth* or *Loihi* (Davies et al., 2018).

Due to their dynamics, SNNs have proved to be useful for robotics applications, among which locomotion tasks (Sreenivasa et al., 2016; Espinal et al., 2016; Vandesompele, Urbain, wyffels, & Dambre, 2019). The bio-inspired coding into spikes seems to have an efficient throughput for real environment applications, and research has been conducted to develop spiking tactile devices in robotics (Georgiou et al., 2007; Dahiya et al., 2010). The ability to integrate DL with SNN (Tavanaei et al., 2019), or more largely with Neurosciences (Marblestone et al., 2016) has also been the focus of several recent research works. A crucial step in this direction has been made with the ability to run back-propagation algorithms in SNNs, demonstrated in (J. H. Lee et al., 2016). Using the self-adaptation properties of the integrate-and-fire neurons, this concept has demonstrated that SNNs can achieve similar performance to ANNs on specific benchmarks (Bellec et al., 2018).

2.2.5 Evolutionary Algorithms

Optimization of parametric controllers is an essential ingredient to obtain good performance in robotic tasks and cope with the un-

certainties related to the robot morphology and the environment. In this section, I consider evolution as a biological inspiration to create optimization algorithms.

Evolutionary Algorithms (EA) have been studied for a few decades in the frame of evolutionary robotics to achieve optimization of controller and morphologies. As suggested in Figure 2.10, they are directly inspired by the evolution of lifeforms in nature, and by the selection and promotion of genotypes that are more adapted to their environment. In this perspective, **EAs are iterative algorithms where populations of individuals, with their difference encoded in a set of parameters, are simulated independently at each generation. The best performing individuals are considered to be the most suited and their parameters are selected for the next generation, after mutation and cross-over**. This process is repeated until convergence is reached.

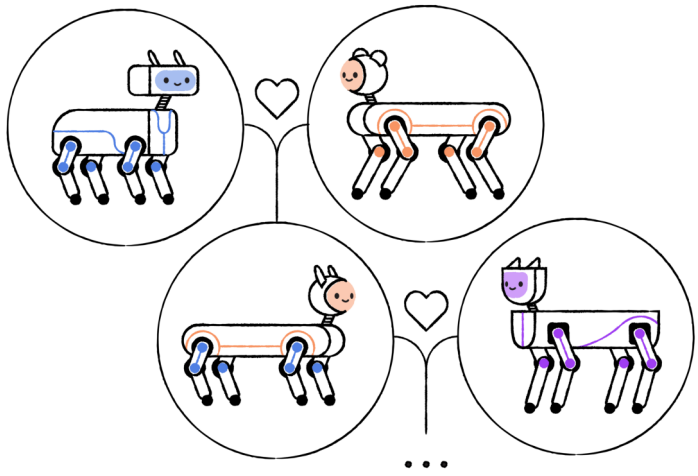


Figure 2.10: This illustration conceptualizes the functioning of EA. In the first generation, the parameters of two individuals are kept for their high performance (for instance, the body size of the orange robot and the leg morphology of the blue robot). Cross-over and mutations are carried at each generation until convergence.

In Lewis et al. (1992), the authors already demonstrated that EAs could be used to optimize a CPG during robot walking. A similar approach is still in use in today's research, as demonstrated by the study on HyQ described in Heijmink et al. (2017). Among the different methods to implement EAs, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) as formulated in Hansen (2006) has revealed especially beneficial to avoid getting trapped in local minima in the iterative optimization process. This is, therefore, advantageous in situations with a complex objective function landscape. CMA-ES has also been used in the gaming industry to animate characters with realistic movements. Although motion capture is generally more popular in the field, the development of powerful physics engines and realistic musculoskeletal models has played an important role in producing agile locomotion behaviors in this field lately (Geijtenbeek & Pronost, 2012). Among the relevant work using CMA-ES, I can mention the optimization of biped control models in J. M. Wang et al. (2012), and combined optimization of morphology and control with creatures using bio-inspired Hill-type muscles in Geijtenbeek et al. (2013).

Despite a certain decline in their usage, **comparative studies with EAs and RL showed that the same level of accuracy could be achieved in both categories** (Real et al., 2018). This is especially true at the early stage of the search process, **which makes EAs appropriate when few computing resources are available or for problems requiring fast convergence, like in robotics.** Nonetheless, EAs and RL present important differences in their formulation and cannot always be applied on the same benchmarks. Indeed, whereas EAs are largely based on heuristics, RL uses a framework mathematically grounded in the Markov decision processes. Also, EAs do not generally use gradient descent to update the value function. Neither do they keep temporal information about the rewarding states, in contrast with RL methods where this is the most common approach.

2.3 Embodied Intelligence

In psychological sciences, it has been demonstrated that individuals' personal representation depends largely on their perceptual and motor interactions and influences complex judgments (Markman & Brendl, 2014). This concept, called **'embodiment' has driven some focus on sensorimotor interactions to understand intelligence better and improve the state of the art in robotics** (Pfeifer et al., 2007). At the same time, the focus in robotics has evolved toward the control of highly compliant systems with many DOFs. Passive, compliant robots possess dynamical properties closer to animal bodies, whose performances can still not be reached, and show a real advantage for solving complex tasks in noisy human environments. However, the classical mathematical framework anchored in Newtonian dynamics presents some strong limitations with compliant bodies. To begin with, the concept of resonance, profoundly related to locomotion performance (Buchli, Iida, & Ijspeert, 2006), does not have a formal meaning with nonlinear systems (Carbajal, 2012).

Pushing this reasoning further, **the 'morphological computation' framework has suggested the existence of a computational potential in physical entities that could be outsourced to solve computing problems.** Successful implementation of morphological computation has been realized with Physical Reservoir Computing (PRC), itself originating from the theory of Reservoir Computing (RC). However, this theory still requires a clear theoretical framework to generalize its use in robotics, or reveal the full potential of the interaction between machine learning and embodiment (Sünderhauf et al., 2018). In this section, I give an overview of some works in morphological computation, RC and PRC, before concluding with some links to evolutionary robotics.

2.3.1 Morphological Computation

It is largely believed that the concept of morphological computation can partly solve the control of compliant complex bodies while

providing adapted embodied controllers that use the body itself as a computational means (Pfeifer et al., 2007; Paul, 2006). Nonetheless, this theory does not have a clear definition, as discussed in Müller & Hoffmann (2017). The definition that I have adopted in Chapter 1 is in line with the experiments conducted in Hauser et al. (2011):

> Morphological Computation embraces any way of increasing efficiency of computation or control in terms of energy, memory, time,... by outsourcing computational tasks to analogical physical systems.

Moreover, it constitutes a fundamental motivation to the embodiment theory, which states that steps towards adaptive intelligence do not come only from the controller complexity but also from the dynamic interactions between the body and its environment. Broader analysis about the quantification of morphological computation as well as the trade-offs with information processing include Polani (2011); Zahedi & Ay (2013); Hoffmann & Müller (2014); Haeufle et al. (2014); Ghazi-Zahedi et al. (2016).

My definition of morphological computation considers the interaction between the robotic structure and the software controller as the starting point to design new robots. Within the possible strategies, I inspire from the work of Müller & Hoffmann (2017) to differentiate 'pure morphological computation' from 'morphology that facilitates control and perception'. **Illustrative applications of the last category for robotic locomotion are numerous.** For instance, Dickinson et al. (2000) provide an analysis of how animals succeed in efficient locomotion using their muscles not solely as motors but to provide multiple functions varying from brakes to springs and struts. The passive walker presented in McGeer (1990) constitutes an extreme example of an engineered robot exploiting this concept. This two-legged physical structure can walk down a slope in a very natural way without any actuation. This work has been extended later in Collins et al. (2005) to robots with low-power actuators. They show a walking pattern that looks natural and energy-efficient compared to traditional stiff controlled robots.

In other fields of robotics, I can also cite the works of Iida & Pfeifer (2006) or Degrave, Caluwaerts, et al. (2015), in which dynamical properties of compliant quadruped robots are used to provide low power consumption, to reduce computational controller complexity, and to observe natural transitions between gaits. Examples that clearly benefit from compliance to perform embodied locomotion can also be found, among others, for hexapod locomotion (Cham et al., 2004), under-water locomotion (M. Ziegler et al., 2006) or closed-link robotic locomotion (Matsuda & Murata, 2006).

2.3.2 Reservoir Computing

RC is the starting point of a practical implementation of morphological computation. It describes a computational framework that enables the approximation of a broad range of dynamical behaviors for which a precise model is not available. RC originates from the domain of RNNs and is mainly based on the theories of Echo State Networks (ESN) and Liquid State Machines (LSM) as outlined in Lukoševičius & Jaeger (2009). At the time of their introduction, they offered a solution to the training of RNN, which was still considered difficult. They avoided having to train feedback connections, and the problems with bifurcations this brings, i.e., the discontinuities in the network output observed for some points in the parameter space, by training only the synaptic connections of the readout nodes. The core architecture consists of a randomly connected RNN, the 'reservoir', for which the synaptic weights are sampled from some distribution and then globally rescaled to tune the dynamical regime close to the edge of chaos.

Apart from simplifying training techniques, RC largely gained popularity due to plausibility to explain some biological concepts (Seoane, 2019). It is also well adapted to evolving topologies and transfer between systems (Chatzidimitriou & Mitkas, 2013) and showed promising results when trained using RL (Szita et al., 2006; Chatzidimitriou et al., 2011). It also resulted in different robotics applications such as learning the inverse kinematics of an iCub robot arm from a neural reservoir in Steil & Reinhart (2009)

or the creation of CPGs to control human movements in wyffels et al. (2014) and hexapod locomotion in Dasgupta et al. (2015).

2.3.3 Physical Reservoir Computing

As the reservoir network is constituted of randomly connected nonlinear entities, many physical dynamical systems presenting sufficiently complex transformations of their inputs provide similar dynamical properties and can be used as reservoirs. For instance, it has been demonstrated in Hauser et al. (2012) that generic types of physical bodies like Mass-Spring-Damper (MSD) networks can approximate any given time-invariant filter with fading memory and generate adaptive periodic patterns autonomously when a feedback loop is added. **This extension of RC is generally referred to as Physical Reservoir Computing (PRC). The expensive step of computing the reservoir transformation is now outsourced to a physical system's natural dynamics.** This means that the neuron states will not be explicitly updated digitally anymore but this computation is transferred to the body's dynamical evolution. The capacity of dynamical systems to process information has been identified in Crutchfield et al. (2010) and studied in Dambre et al. (2012) as a way to perform computing beyond digital architectures and silicon substrates.

The main advantage of PRC lies in the parallelism of the computations in the physical reservoir and, in the case of robotic locomotion, in the fact that the transformations computed by the robot body are a natural result of the gait. However, PRC is essentially a supervised machine learning technique. In contrast, robotic control is intrinsically an RL problem, in which the optimal desired actuator signals are not known a priori. In addition, successful reservoir implementations often require the observation of the reservoir state at many different points. In robotics, this implies that for each observation point a sensor needs to be installed.

Several applications of PRC have been demonstrated in the past decade. A recent overview is provided in Nakajima (2020) and a

review linking PRC with the state-of-the-art in neural networks is given in Tanaka et al. (2019). In robotics, highly compliant robot models have been addressed for example to simulated MSD networks (Hauser et al., 2011; Yamanaka et al., 2018), tensegrity structures (Caluwaerts et al., 2012, 2014), simulated ball robots (Martin et al., 2016) and soft robotic arms (Nakajima et al., 2014, 2015; Eder et al., 2018). Closed-loop control of a quadruped robot exploiting a spine made with soft material as a reservoir can be found in Zhao et al. (2013). Simulations or implementations of PRC outside robotics include water ripples in Fernando & Sojakka (2003), electro-optical devices in Larger et al. (2012) or pure optical devices in Brunner et al. (2013) and Vandoorne et al. (2014).

2.3.4 Evolutionary Robotics

The closed interaction between morphology and control defined in the theory of embodiment has brought the researchers to focus on evolution (Pollack & Lipson, 2000) and its contribution to robotics. **This theory, called'evolutionary robotics', has demonstrated the potential to guide engineers in designing new robot structures (Hermans et al., 2014) and to build robots and controllers that can deal with injuries and minor damages using an internal representation of themselves in the external environment (Bongard et al., 2006; Cully et al., 2015)**. Evolutionary robotics is also an efficient path to better design and control compliant and soft robots. In Cheney et al. (2018), for instance, it is discussed how to optimize modular soft robots to obtain high locomotion performance. The role of evolution has also been highlighted on the controller side. In Clune et al. (2013), the authors focused on biological neural networks and showed how artificial modular neural controllers are more adapted to survive through evolution, which encourages engineering developments in this direction. Finally, links with classical machine learning techniques have been investigated in Ha (2019), who used RL to generate an evolution strategy of a robotic morphology across various environments.

CHAPTER 3

—

Locomotion Control Methods

From control theory to machine learning, this chapter provides a brief theoretical guide through different control methods in robotic locomotion.

This chapter introduces a framework to study robotic locomotion control. In the first section, I discuss the control of linear and non-linear systems using a simplified example, with a single DOF. An extension to systems with multiple DOFs is given in the second section, and the concept of motor coordination is detailed thereafter. The third section presents the framework per se and breaks it down for different fields. Lastly, some useful metrics to evaluate locomotion are defined in the last section.

3.1 Control Theory

Before jumping into advanced locomotion control methods, **a simple control theory problem shall help to set up the theoretical foundations and terminology correctly**. To this effect, let me consider the system formed by a perfect bike with a fixed handle on flat ground with homogeneous friction coefficient as represented in Figure 3.1.

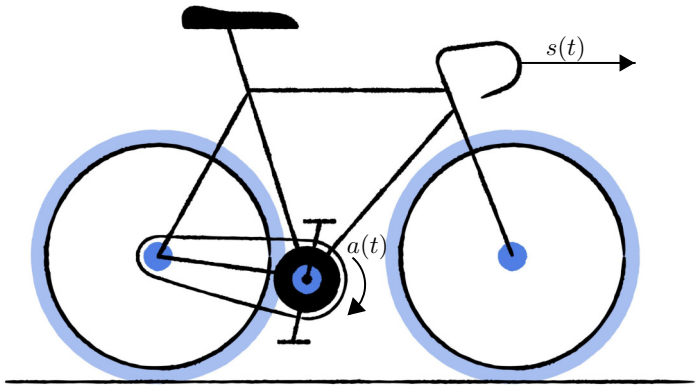


Figure 3.1: This bike is a linear system with one DOF. The input is the pedal speed $a(t)$ and the output is the bicycle speed $s(t)$.

For the sake of simplicity, I assume that the model takes the rotational speed applied on the pedals axis $a(t)$ as single input. The model outputs the bicycle speed on the ground $s(t)$. This

system is linear with one degree of freedom: the output speed is always proportional to the input speed, and it can be modeled with a simple linear function:

$$s(t) = \kappa \, . \, a(t), \tag{3.1}$$

where κ is the single parameter of the model. I use the following notations to describe the problem:

- $a(t)$ is called the **control** signal
- $s(t)$ is called the **feedback** signal
- $r(t)$ is called the **target** signal.

The goal of control theory is to find the optimal control signal $a(t)$ that shall be applied in order to produce the target signal $r(t)$ computed using the feedback signal $s(t)$ acquired from the system. For a linear and one-dimensional system like this bicycle, it is easy to define an optimal open-loop solution (Figure 3.2) satisfying the problem with the equation:

$$a(t) = \frac{1}{\kappa} \, . \, r(t). \tag{3.2}$$

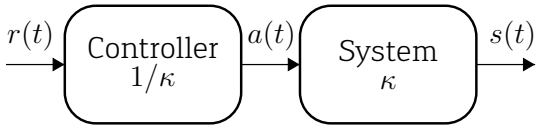


Figure 3.2: A linear system (like the bicycle from Figure 3.1) can be driven with a proportional control law in open-loop.

However, the physical phenomenons of real systems generally involve dynamical interactions and can rarely be described by a single linear equation, although some local approximation techniques can often be formulated. Therefore, there is a clear need to regulate the action based on the feedback signal in closed-loop. To illustrate this, I consider now the two-dimensional car system represented in Figure 3.3.

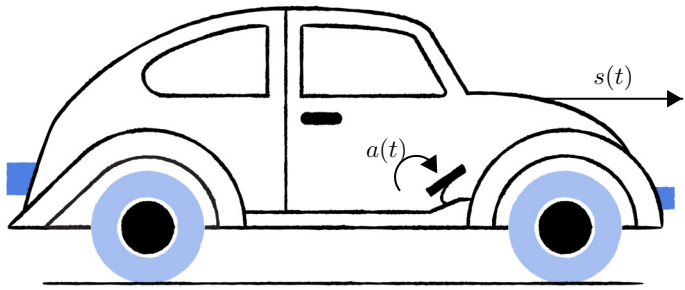


Figure 3.3: Between its input $a(t)$ and its output $s(t)$, this car acts as non-linear system, governed by differential equations and subject to external disturbance $\phi(t)$

I also assume a cruise controller, in charge of regulating the car speed $s(t)$, using the displacement of the accelerator pedal as a control signal $a(t)$. The system mapping the pedal movement to the car speed includes some non-linear processes, like the fuel compression and the combustion mechanism in the cylinder. But it can also include random disturbances like the wind applied on the car or the slope of the road. Due to this, **the model** linking the control signal $a(t)$ and the sensor output $s(t)$ **is now a differential equation** f **with stochastic disturbances** $\phi(t)$:

$$f(a, \dot{a}, \ddot{a}, s) + \phi(t) = 0. \tag{3.3}$$

In the general case, this equation can hardly be solved analytically to find the optimal control signal $a(t)$ for a target output $r(t)$. In control theory, instead of trying to define this solution, we compute the **error** signal $e(t)$, i.e. the difference between the feedback signal $s(t)$ and the target $r(t)$:

$$e(t) = r(t) - s(t). \tag{3.4}$$

The goal of the controller is therefore to minimize this quantity and keep $s(t)$ as closed as the desired output $r(t)$. To achieve this, we use a Proportional Integral Derivative (PID) law with three independent gains:

- k^{p}: the **proportional term**, which acts directly on the error $e(t)$. When the current car speed differs from the desired one, this term will compensate by applying a proportional action.
- k^{i}: the **integral term**, which accounts for the error accumulation as the time passes. The importance of this term strengthens when the error has been of the same sign for a long time.
- k^{d}: the **derivative term**, which accounts for the slow or fast variations of the error. This term becomes important when the regulated speed undergoes a sudden change.

The PID law can be written:

$$a(t) = k^{\mathrm{p}}\, e(t) + k^{\mathrm{i}} \int_0^t e(t')\, dt' + k^{\mathrm{d}}\, \frac{\mathrm{d}e(t)}{\mathrm{d}t}. \tag{3.5}$$

Tuning the three parameters of a PID control law can fill a book on its own and strongly depends on the characteristics of the system under control (J. G. Ziegler & Nichols, 1942).

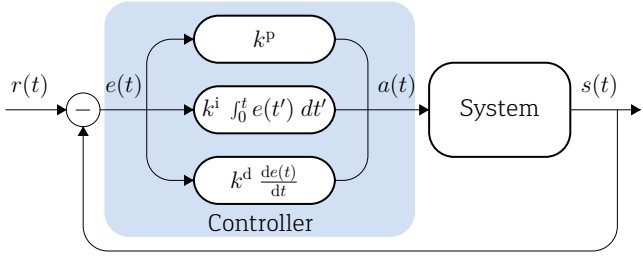


Figure 3.4: With non-linear systems and in the presence of external noise (car in Figure 3.3), a closed loop implementing a PID control law is more adapted than a linear controller to regulate the system correctly.

In today's controllers, the control process is generally discretized and runs on a digital computer. Conversions between analog and digital signals are realized on dedicated devices, which act as low-pass filters with a cutting frequency defined by the Shannon theorem (Shannon, 1949), therefore limiting the bandwidth of

$a(t)$ and $s(t)$. In this dissertation, I will use the same sampling frequency for processing the feedback and operating the PID controller. I will furthermore ensure that this frequency is sufficiently high to offer a correct bandwidth and neglect these adversarial effects, or it will be briefly discussed otherwise. The time will be discretized using a fixed time step and will be written as t_k.

3.2 Locomotion Framework

In general, a system can be modified using several action inputs, and the user may want to regulate multiple outputs together or a function of them. This becomes especially true when moving from wheeled systems to legged systems. The quadruped robot represented in Figure 3.5 has four legs with 3 DOFs each. The control signal at time t_k is now a vector of size 12:

$$\mathbf{a}_k = \{a_k^1, a_k^2, a_k^3, ..., a_k^{12}\}. \tag{3.6}$$

These 12 scalars are not independent of each other due to the mechanical linkage between them. For example, to move the foot tip of one robot leg with a given angle, one can find different combinations of hip and knee joint positions. Therefore, there is a need to find a control law taking this evidence into account.

Also, the feedback signal $\mathbf{s}_k$ is generally much more diverse than the theoretical cases presented above. In practice, it can include an Inertial Measurements Unit (IMU)[1], some load cells to measure GRFs, proprioceptive sensors to measure kinematic parameters, vision sensors, ... Therefore, in locomotion, the dimensionality of $\mathbf{s}_k$ can be quite high, although the end-user may want to regulate only a few behavioral outputs like speed or stability. We can write:

$$\dim(\mathbf{r}_k) < \dim(\mathbf{s}_k). \tag{3.7}$$

[1] An IMU is an electronic device that can sense the inertial movement of a body using accelerometers and gyroscopes. It can also integrate them to obtain speed and position

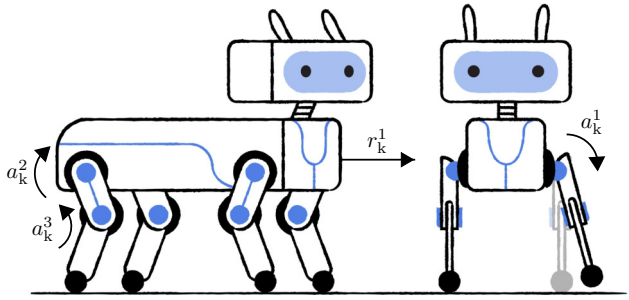


Figure 3.5: A quadruped robot typically possess a large number of inputs and outputs and requires control methods more advanced than a PID.

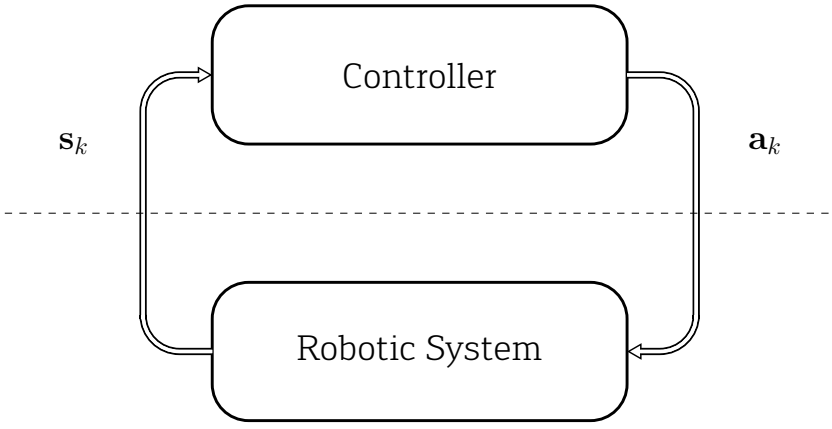


Figure 3.6: In this chapter, I consider the locomotion problem as a closed-system formed by a robot and a controller. Different control approaches are detailed in this chapter, and three robots are presented in chapter 4.

In this context, **I define a 'locomotion control framework' to discuss and compare the most common approaches that have been studied in classical robotics and machine learning.** This framework is presented in Figure 3.6. The robotic system is governed by the action $\mathbf{a}_k$ at each time step t_k, but is also subject to external disturbances (which are generally a combination of a statistical random process and unknown deterministic functions). The mechanical linkages and the interactions with the external environment produce a sensor signal $\mathbf{s}_k$ at each time step. The controller predicts the next control vector $\mathbf{a}_k$, using knowledge about the current sensors but also, in some cases, the previous sensors' and actuators' values. This set of variables is also called the past 'system's trajectory'.

The controller's mathematical function computing the action $\mathbf{a}_k$ is generally called a control 'policy', written π_k. By convention and for the sake of clarity, I consider that this policy transforms $\mathbf{s}_k$ into $\mathbf{a}_k$ at the same time step, whereas some other may consider a time increment within the policy (i.e. $\mathbf{s}_k$ leads to $\mathbf{a}_{k+1}$). In practice, the policy can take different forms to achieve the task, including finite state machines, optimizers, parametric bio-inspired models, neural networks, other machine learning architectures, etc. In the next sections, I will first detail a classical implementation of this controller for static and dynamic locomotion gaits, then move to their study in the field of machine learning. It may be important to remark that, in the mathematical notations, this control policy π_k is a function. In contrast, control policies using Markov decision processes will be employed in Section 3.4.3 to describe RL methods.

3.3 Classical Quadruped Locomotion Control

3.3.1 Static Locomotion

The approach for **control of static locomotion consists of keeping a stable pose at all time in the process**. A simple way to ensure this condition in quadruped robotics is to constrain the projection of the trunk's COG on the ground to always stay in the polygon

formed by the contact points between the legs and the ground. A direct consequence of this proposition is that there should be a minimum of three legs in contact with the ground at each time step. A typical gait following this description is the walking gait (see Figure 2.4 in Chapter 2).

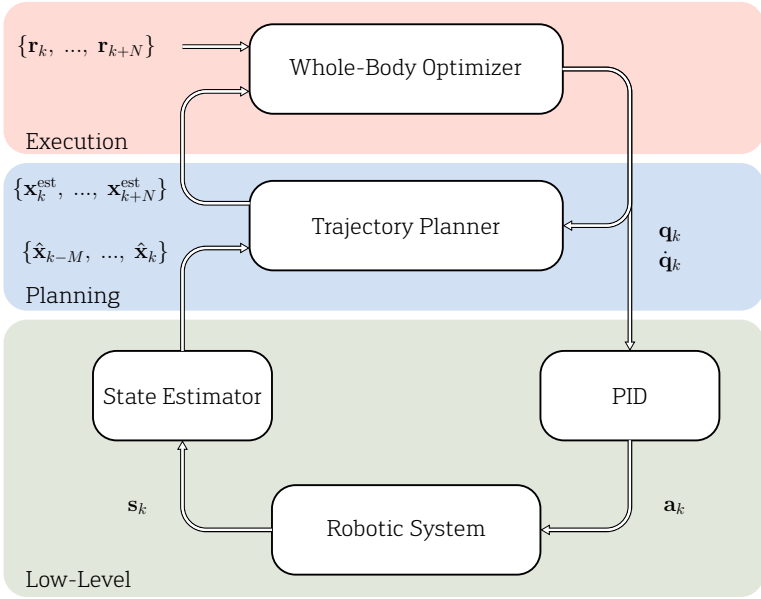


Figure 3.7: In MPC, an advanced model of the robot is a central element to provide the optimizer with the necessary information to predict the best action for the next step.

A convenient implementation for static gaits relies on Whole-Body Control together with Model Predictive Control (MPC) presented in Figure 3.7. In this architecture, the sensor's signal is processed in a state estimator that computes the real past trajectory of the robot in its environment $\{\hat{\mathbf{x}}_{k-M}, \ldots, \hat{\mathbf{x}}_{k-1}, \hat{\mathbf{x}}_k\}$. A **trajectory planner implements a model of the system** to forecast the motion of the robot $\{\mathbf{x}_k^{\text{est}}, \mathbf{x}_{k+1}^{\text{est}}, \ldots, \mathbf{x}_{k+N}^{\text{est}}\}$ over a horizon N and given a possible vector of joint positions $\mathbf{q}_k$ and velocities $\dot{\mathbf{q}}_k$ and the past trajectory from the state estimator. Using the target $\{\mathbf{r}_k, \mathbf{r}_{k+1}, \ldots, \mathbf{r}_{k+N}\}$ requested by the user or a high-level state machine, the predicted signal of the trajectory planner and the

effective past trajectory from the state estimator, **the whole-body optimizer aims at finding the optimal future actuation signal** $\{\mathbf{q}_k,\ \dot{\mathbf{q}}_k\}$ to regulate the system. This is done by minimizing the distance between $\{\mathbf{r}_k,\ \mathbf{r}_{k+1},\ ...,\ \mathbf{r}_{k+N}\}$ and $\{\mathbf{x}_k^{\text{est}},\ \mathbf{x}_{k+1}^{\text{est}},\ ...,\ \mathbf{x}_{k+N}^{\text{est}}\}$. To simplify this task, this optimization can be constrained. Classical constraints concern the stability criterion, slip detection, reflexes, and the Zero Moment Point (ZMP), further defined in Vukobratovi & Borovac (2004), which aims at finding an equilibrium for the forces at the ground contact point during tilting. The output of the whole-body optimizer is a vector of joint positions $\mathbf{q}_k$ and velocities $\dot{\mathbf{q}}_k$ that are both feeded to the planner and processed by parallel PID controllers to actuate the motors.

3.3.2 Dynamic Locomotion

The approach presented in the previous section has two major drawbacks. First, it requires an accurate model of body dynamics to predict the next actions correctly. In soft or compliant robotics, this model can be hard to design, computationally intensive, and sometimes inaccurate. Secondly, it relies on a complex optimization procedure, limiting the real-time application in unstable situations like dynamic gaits.

In a dynamic gait, the robot stays sometimes on less than three legs at a time: **the static balance is not ensured and the absence of a fall is based on the precise timing of the movement, and attitude correction** which guarantees a good posture in terms of body's height, roll and pitch. In this situation, **a simplification of the control can be realized by decoupling the control of stability and foot motion**, as presented in Figure 3.8. Foot motion is generated by rhythmic oscillators such as CPGs, producing a cyclic foot trajectory in a specific frame of reference $\mathbf{x}_k^{\text{mot}}$. Using the current state of the robot $\hat{\mathbf{x}}_k$, they can be shaped to integrate different reflexes and interlimb coordination mechanisms. Mapping to the joint space of reference is applied using Inverse Kinematics (IK), which produces the desired joint positions and speeds $\mathbf{q}_k, \dot{\mathbf{q}}_k$. In parallel, the stability controller ensures that the body stays in a

horizontal plane based on inertial sensing, i.e. using linear and angular accelerations of the rigid bodies, $\hat{\ddot{\mathbf{x}}}_k$. The adjusting accelerations $\ddot{\mathbf{x}}_k^{\text{stab}}$ are converted into torques $\boldsymbol{\tau}_k^{\text{stab}}$ using an Inverse Dynamics (ID) model of the robot.

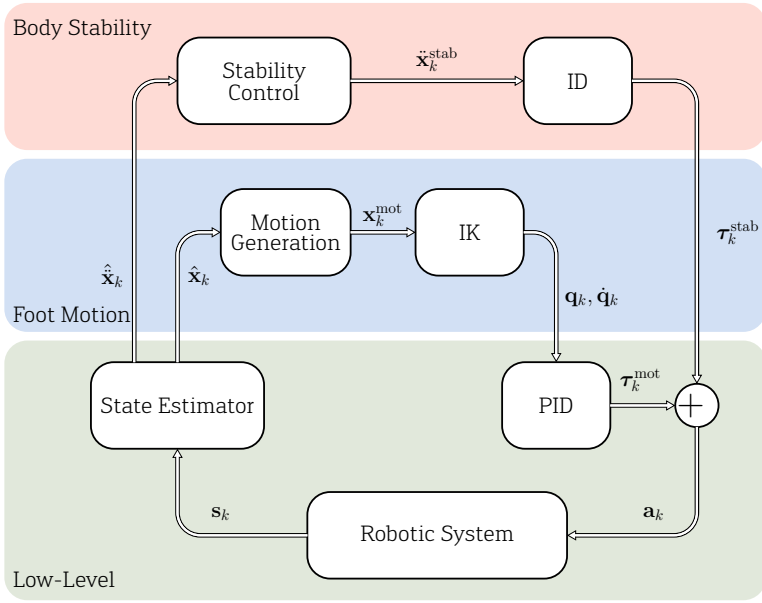


Figure 3.8: Managing dynamic locomotion of legged systems can generally be decoupled in two separate control processes: one generating a periodical foot motion and the other regulating the robot stability. In this Figure, $\hat{\mathbf{x}}_k$ and $\hat{\ddot{\mathbf{x}}}_k$ represent the estimation of the current positions and accelerations of the different robot rigid links, $\mathbf{x}_k^{\text{mot}}$ is the desired position of the robot rigid links produced by the motion generation controller, and $\ddot{\mathbf{x}}_k^{\text{stab}}$ is the stabilizing acceleration from the stability controller.

3.4 Learning-Based Control

A clear advantage of data- and learning-based control techniques is the reduced prior knowledge about the robot's model, itself leading to a simplified control architecture. Instead, these algorithms rely on some training data to produce a suitable model. Although several works use data-based models to only replace selected parts

of the full locomotion architecture (for some example, see Chapter 2), it is also possible to replace the full controller with a generic policy that is further trained on examples or exploration. As I will detail in each sub-section, this policy can be a high-level model, such as a parametric CPG, but it will generally be instantiated by a function approximator, such as a feed-forward neural network for instance. **Learning end-to-end control is strictly in line with embodiment theory.** First, **it avoids sub-optimal projections between different frames of reference** (sensor, actuator, Cartesian), and the important computation overhead that goes with it. Secondly, **it allows finding particularly efficient locomotion behaviors that exploit the body dynamics at their best** but which cannot be explicitly analyzed in the Cartesian space or planned by humans. Third, this kind of controller **is more adapted to under-actuated, compliant, and flexible parts** without adding extra complexity to model the dynamics.

However, this approach has also many drawbacks. First, we can mention the prohibitive time spent on the learning process. While this one can be partially conducted in simulation, there is a clear difficulty in transferring a policy to a real robot, which is still an open issue in scientific literature, as mentioned in Section 2.2.4.4. Secondly, learning involves potential damages when the robot is exploring its own motor control capacities. This includes robot falling, wearing and hysteresis of mechanical parts, or even breaking sub-systems of the robot. Thirdly, embodied results are hardly interpretable, which makes it difficult for engineers to optimize them partly after training or to correct undesired artifacts.

3.4.1 Evolutionary Algorithms

The goal of EA is to solve optimization problems for which it is difficult, if not impossible, to define an analytical solution in a decent time. **Inspired by biological evolution, genetic algorithms have been the original implementation of EA (Poli et al., 2009)**. They can be implemented through the following procedure:

1. Determination of an initial population of individuals with random parameters $\boldsymbol{\theta}$
2. Evaluation of each individual's performance score G_i
3. Selection of the best individuals in the population
4. Mutation and crossbreeding to create a new population
5. Iteration over steps 2 to 4 until convergence above a performance threshold or after a certain number of iterations

In this procedure, the evaluation step can be relatively long. In locomotion, this generally consists of maintaining a gait for a short time and evaluate its performance. Each population is made of several individuals, and the full process can involve hundreds or thousands of generations. Therefore, one main drawback of this method is the slowness of convergence. However, when performed in simulation, it can be easily parallelized, which has a significant impact if it runs on a computer with many processing cores. Another disadvantage is the necessity to select a performance score. This choice is critical and can considerably influence the result and the convergence speed.

In this thesis, I will rely on an implementation of the algorithm presented above, called Covariance Matrix Adaptation Evolution Strategy (CMA-ES), as described in (Hansen, 2016). In EAs, the populations introduced at each iteration are generally sampled according to a multivariate normal distribution to model the combination and mutation processes. The relation between the variables in the distribution is represented by a covariance matrix. The CMA-ES algorithm relies on the adaptation of this covariance matrix at each iteration, which is similar to learning a second-order model of the score function. The principal motivation behind this choice is the fast convergence of the algorithm in the presence of a non-convex parameter landscape with many local minima. The author of this method also provides good documentation for optimally choosing the meta-parameters. For instance, the population size λ is an important criterion: a small λ leads to faster convergence,

and a large λ helps to avoid local optima. The paper suggests using the following empirical equation to obtain a good balance between the two effects:

$$\lambda = 4 + 3\lfloor \ln(n) \rfloor, \tag{3.8}$$

where n is the size of the parameter vector $\boldsymbol{\theta}$. I will use this equation to compute λ in the rest of this manuscript.

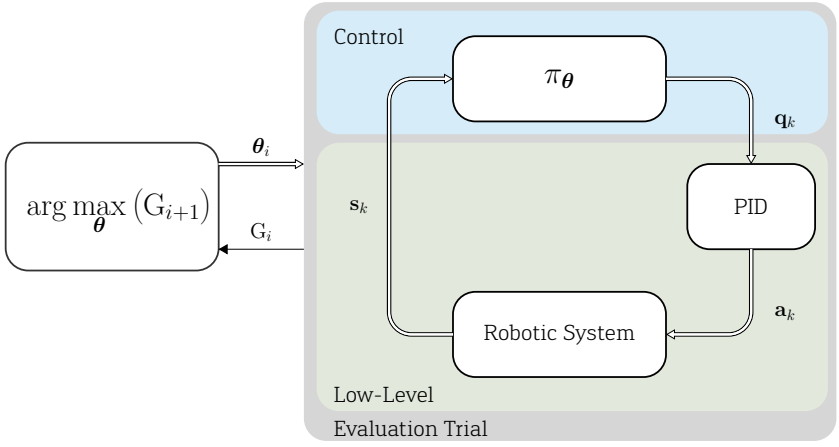


Figure 3.9: In EA, a parametric control policy is optimized in an iterative process to maximize a score function. The optimization process is generally inspired by the theory of evolution.

In Figure 3.9, I represent the framework architecture with an EA. The robot is controlled by a parametric policy $\pi_{\boldsymbol{\theta}}$, which predicts the next joint position $\mathbf{q}_k$, given the current sensor signal $\mathbf{s}_k$. This policy can take various forms and embeds more or less complex models. It can have a feed-forward structure or implements recurrences. The parameter vector $\boldsymbol{\theta}$ is fixed for the full duration of the evaluation. It will generally be a locomotion trial of a few seconds.

The score G_i describes how the gait performs on the robot, taking into account various criteria such as the distance, the robot's stability, or its energetic performance. A specific inventory of possible metrics is given in section 3.6, and discussions are provided further in the document to motivate the choices in different experiments.

With this score, the EA produces a new population of individuals. The iterative process continues until convergence or interruption of the algorithm. In the former case, it can be considered that the optimal policy $\hat{\pi^\star}$, defined by a set of optimal parameters $\hat{\boldsymbol{\theta}}^\star$, has been reached.

In Chapter 5, EA will also be used in a different context to optimize the parameters of a physics simulator. I will use the same principle, although the task, the policy, and the score will be different.

3.4.2 Supervised Learning

SL describes a subset of machine learning tasks, for which the goal of the algorithm is to estimate a mapping function using a series of known examples. In the framework understudy, I use $\mathbf{u}_k$ to denote the input of this function. It is obtained after processing the robot sensors signal $\mathbf{s}_k$. This processing task can include a memory buffer, which stores a certain amount of time steps in a queue, a discrete filter, to remove noisy inputs for instance, or a non-linear kernel, such as randomly connected neural layers. The output is the vector of joint positions $\mathbf{q}_k$ with

$$\mathbf{q}_k \in \mathbb{R}^J, \tag{3.9}$$

with J, the size of the vector $\mathbf{q}_k$. The learning algorithm is, therefore, a 'regression' problem.
Processing the sensors plays a capital role and will be detailed separately in each experiment. Reducing the amount of prior knowledge when designing the policy and sensor pre-processing can lead to a better embodiment of the controller and the body, while also reducing the computation. However, this can also increase requirements of example data and training time.

The parametric policy mapping the inputs and outputs is labeled $\pi_{\boldsymbol{\theta}}$. In this work, I will always rely on a single neural layer, also called the 'readout layer' to implement this policy, but other forms of function can be found in the literature. This layer has J neurons, fully connected to the inputs vector in the same way as displayed

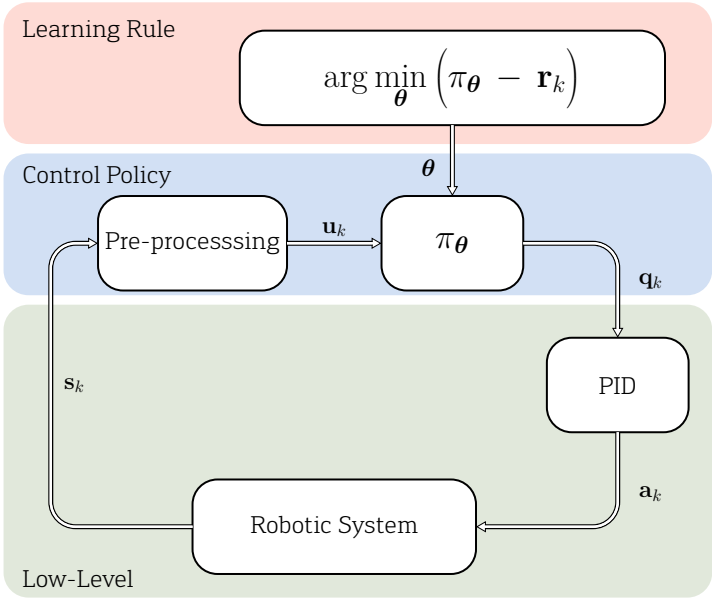


Figure 3.10: SL minimizes the error between a target signal and the prediction of a parametric policy to find the optimal set of parameters for this one.

in Figures 2.5 and 2.6. Therefore, each output $\mathbf{q}_k^j$, is the result of a linear operation (a weighted sum of the inputs), followed by a non-linear operation (the activation function of the neuron) and we can write:

$$\mathbf{q}_k = \mathbf{f}(\mathbf{W}_{\text{out}}^{\text{T}} \,.\, \mathbf{u}_k). \tag{3.10}$$

In this equation, $\mathbf{W}_{\text{out}}$ is the connectivity matrix of dimension $U \times J$, if u is of dimension U and q of dimension J. The size of $\boldsymbol{\theta}$ is equal to the number of elements in $\mathbf{W}_{\text{out}}$:

$$\dim(\boldsymbol{\theta}) = U \times J. \tag{3.11}$$

The function $\mathbf{f}$ is a multi-dimensional function of dimension J representing the activation function of all the neurons. The learning rule of the algorithm aims at finding the optimal set of parameters $\boldsymbol{\theta}^\star$ in order to minimize the error with the target signal $\mathbf{r}_k$:

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} \mathbf{e}(\boldsymbol{\theta}). \tag{3.12}$$

A typical method to compute this error $\mathbf{e}(\boldsymbol{\theta})$ between policy and target is based on Euclidean distance:

$$\boldsymbol{e}(\boldsymbol{\theta}) = \frac{1}{K} \sum_{k=0}^{K} \Big(\mathbf{f}(\mathbf{W}_{\text{out}}^{\text{T}} \,.\, \mathbf{u}_k) \;-\; \boldsymbol{r}_k \Big)^2, \tag{3.13}$$

where K is the total number of time steps for the full training period. A learning rule employing such an error metric is called 'offline': it requires to know all steps prior to computing the error and determining the best set of parameters $\hat{\boldsymbol{\theta}}^{\star}$. To avoid this issue, I use an 'online' algorithm instead. In this dissertation, I rely on the FORCE algorithm (Sussillo & Abbott, 2009), which is itself an extension of the Recursive Least Square (RLS) algorithm (Haykin & Simon, 1996), adapted to add a regularization parameter α. With this algorithm, the weights of the output matrix $\mathbf{W}_{\text{out}}$ can be updated at each time step using the following equations:

$$\mathbf{e}_k = \mathbf{f}\big(\mathbf{W}_{k-1}^{\text{out}\,\text{T}} \,.\, \mathbf{u}_k \big) \;-\; \mathbf{r}_k \tag{3.14}$$

$$\mathbf{P}_k = \mathbf{P}_{k-1} \;-\; \frac{\mathbf{P}_{k-1} \,.\, \mathbf{u}_k \,.\, \mathbf{u}_k^{\text{T}} \,.\, \mathbf{P}_{k-1}}{1 + \mathbf{u}_k^{\text{T}} \,.\, \mathbf{P}_{k-1} \,.\, \mathbf{u}_k} \tag{3.15}$$

$$\mathbf{W}_k^{\text{out}} = \mathbf{W}_{k-1}^{\text{out}} \;-\; \mathbf{P}_k \,.\, \mathbf{u}_k \,.\, \mathbf{e}_k^{\text{T}} \tag{3.16}$$

$$\mathbf{q}_k = \mathbf{f}\big(\mathbf{W}_k^{\text{out}\,\text{T}} \,.\, \mathbf{u}_k \big), \tag{3.17}$$

where the estimate of the inverse of the correlation matrix $\mathbf{P}$ is initialized to $\mathbf{I}/\alpha$ (with $\mathbf{I}$, the identity matrix). The larger α is, the more regularization there is in the process: learning will take longer but will generalize on more examples.

3.4.3 Reinforcement Learning

The RL framework shown in Figure 3.11 presents some similarities with the SL one presented in the previous section. However, as explained in Chapter 2, the essential difference is the absence of a target during training. Also, **the developments presented in this section are based on the concept of Markov decision processes. To this goal, all the signals are considered random processes**, and the policy π is a probability. According to Sutton & Barto

(2013), the model-free RL introduces three essential notions:

- **A policy**, $\pi_{\boldsymbol{\theta}}\ (\mathbf{u}_k)$, mapping from perceived states of the environment to actions to be taken when in those states. This policy is generally a neural network. It introduces a set of parameters $\boldsymbol{\theta}$ that is subject to training.
- **A reward signal**, r_k, defining what are the good and bad events for the agent at each time step. In locomotion, this can be the current speed of the robot, for instance, but other possibilities are given in Section 3.6.
- **A value function**, $V(\mathbf{u}_k)$, that specifies what is good in the long run – whereas the reward signal only indicates what is good in an immediate sense. Roughly speaking, the value of a state is the total reward that an agent can expect to accumulate over the future.

In the architecture under study (see Figure 3.11), the sensor signal $\mathbf{s}_k$ can be processed to provide the variable $\mathbf{u}_k$. Again, this processing can include filters, memory buffers, and convolutions with a non-linear kernel. I consider therefore that the policy takes the signal $\mathbf{u}_k$ as input that is called 'state', and that it transforms it into a vector of joint positions $\mathbf{q}_k$, called an 'action'

In the RL framework, the optimal policy $\pi^\star$, with optimal parameters $\boldsymbol{\theta}^\star$, **is determined for the set of actions that maximize the total reward** (Alpaydin, 2010). **In order to find it, one can use the value function** $V^{\boldsymbol{\theta}}(\mathbf{u}_k)$, i.e. the cumulative reward that will be received while the agent follows the policy with parameters $\boldsymbol{\theta}$ starting from state $\mathbf{u}_k$:

$$\begin{aligned} V^{\boldsymbol{\theta}}(\mathbf{u}_k) &= \mathrm{E}\left[\ r_{k+1}\ +\ \gamma\, r_{k+2}\ +\ \gamma^2\, r_{k+3}\ +\ \ldots\ \right] \\ &= \mathrm{E}\left[\ \sum_{i=1}^{\infty} \gamma^{i-1}\, r_{k+i}\ \right], \end{aligned} \tag{3.18}$$

where $E[\,\bullet\,]$ is the expectation of the random process between squared bracket. γ is called the discount rate and is included between 0 and 1.

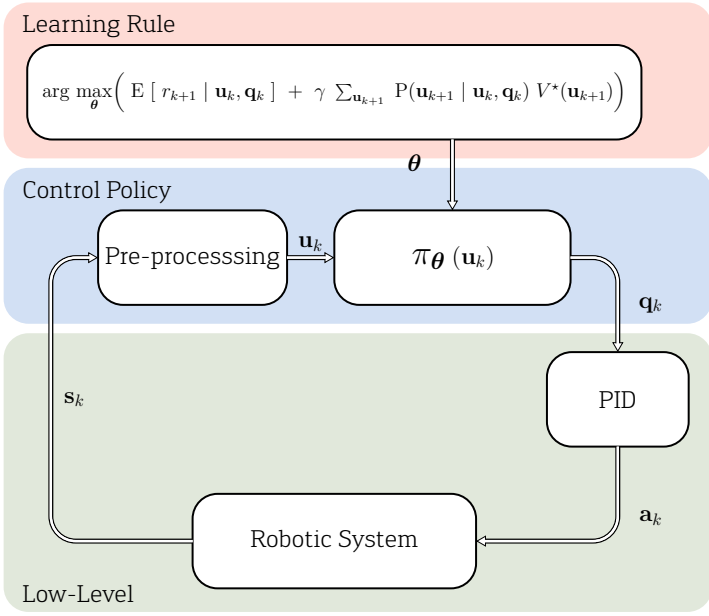


Figure 3.11: In this framework, the parameters $\boldsymbol{\theta}$ of a parametric policy are determined though a maximization of the value function, i.e. the total amount of rewards that a agent can expect to accumulate over the future if it follows the best trajectory.

Therefore, the learning problem can be rephrased to finding the policy $\pi^\star$ which maximizes the value function:

$$V^\star(\mathbf{u}_k) = \max_\pi V^\pi(\mathbf{u}_k), \ \forall \mathbf{u}_k. \tag{3.19}$$

Instead of working with the value of a state $V(\mathbf{u}_k)$, one can also use the value of state-action pair $Q(\mathbf{q}_k, \mathbf{u}_k)$, also called Q-value. $Q^\star(\mathbf{q}_k, \mathbf{u}_k)$ is then the cumulative reward if we take action $\mathbf{q}_k$ in state $\mathbf{u}_k$ and follow $\pi^\star$ afterwards. In other words, we can write:

$$\begin{aligned} V^\star(\mathbf{u}_k) &= \max_{\mathbf{q}_k} Q^\star(\mathbf{q}_k, \mathbf{u}_k) \\ &= \max_{\mathbf{q}_k} \mathrm{E}\Big[r_{k+1} + \gamma \sum_{i=1}^{\infty} \gamma^{i-1} \ r_{k+1+i} \ \Big] \\ &= \max_{\mathbf{q}_k} \mathrm{E}\Big[r_{k+1} + \gamma \ V^\star(\mathbf{u}_{k+1}) \ \Big] \\ &= \max_{\mathbf{q}_k} \Big(\mathrm{E}[r_{k+1}] + \gamma \sum_{\mathbf{u}_{k+1}} \mathrm{P}(\mathbf{u}_{k+1} | \mathbf{u}_k, \mathbf{q}_k) \ V^\star(\mathbf{u}_{k+1}) \Big), \end{aligned} \tag{3.20}$$

where $\mathrm{P}(\mathbf{u}_{k+1} | \mathbf{u}_k, \mathbf{q}_k)$ is the probability to move from a state $\mathbf{u}_k$ to a state $\mathbf{u}_{k+1}$ when the action $\mathbf{q}_k$ is undertaken. This equation is also called the Bellman's equation, as defined in Bellman (1957). If the probabilities $\mathrm{P}(\mathbf{u}_{k+1} | \mathbf{u}_k, \mathbf{q}_k)$ are completely known, the optimal policy can be found by taking the action that will maximize the value in the next state:

$$\begin{aligned} \pi^\star(\mathbf{u}_k) = \arg\max_{\mathbf{q}_k} \Big(& \mathrm{E}\left[\ r_{k+1} \mid \mathbf{u}_k, \mathbf{q}_k \ \right] \\ & + \gamma \sum_{\mathbf{u}_{k+1}} \mathrm{P}(\mathbf{u}_{k+1} \mid \mathbf{u}_k, \mathbf{q}_k) \ V^\star(\mathbf{u}_{k+1}) \Big) \end{aligned} \tag{3.21}$$

However, as I mentioned in the beginning of this section, **we focus on model-free algorithms, for their application in compliant robotics. Therefore, the probabilities $\mathrm{P}(\mathbf{u}_{k+1} \mid \mathbf{u}_k, \mathbf{q}_k)$ are generally unknown and we need to explore the possible state-action pairs to fill this gap.** A common exploration strategy is called ϵ-greedy: at each time step, the agent selects a random action with a fixed probability ϵ instead of selecting greedily one of the learned optimal actions (Tokic, 2010). Another strategy is called 'softmax': at each time step, the agent selects an action with a

probability determined by ranking the value function estimates using a Boltzmann distribution:

$$\mathrm{P}(\mathbf{q}_k \mid \mathbf{u}_k) = \frac{e^{Q(\mathbf{u}_k, \mathbf{u}_k)/T}}{\sum_{\mathbf{u}_k \in \mathcal{Q}} e^{Q(\mathbf{v}_k, \mathbf{u}_k)/T}} \tag{3.22}$$

where $\mathcal{Q}$ is the set of possible actions and T is a parameter called the 'temperature'. In a deterministic case, this exploration strategy allows to assign an estimate of the current Q-value of a state-action pair in an iterative process using:

$$\hat{Q}(\mathbf{q}_k, \mathbf{u}_k) \leftarrow r_{k+1} + \gamma \max_{\mathbf{q}_{k+1}} \hat{Q}(\mathbf{q}_{k+1}, \mathbf{u}_{k+1}). \tag{3.23}$$

In contrast, in a non-deterministic environment, like we generally have in robotic control, the next state cannot be known with certitude for a given action. Instead, it is determined by a probability $\mathrm{P}(\mathbf{u}_{k+1}|\mathbf{u}_k, \mathbf{q}_k)$. A common solution to update the Q-value is then to use the 'Q Learning' algorithm which performs:

$$\begin{aligned} \hat{Q}(\mathbf{q}_k, \mathbf{u}_k) \leftarrow \; r_{k+1} + \eta\Big(r_{k+1} + \gamma \max_{\mathbf{q}_{k+1}} \hat{Q}(\mathbf{q}_{k+1}, \mathbf{u}_{k+1}) \\ - Q(\mathbf{q}_k, \mathbf{u}_k)\Big) \end{aligned} \tag{3.24}$$

in a iterative way, where η is called the learning rate. Other exploration and exploitation strategies have been implemented and discussed to converge towards $Q^\star$ and find the optimal policy $\pi^\star$ (Sutton & Barto, 2013) but they are out of the scope of this dissertation.

Lastly, it is important to note that I have only considered discrete states and actions up to now, for which the values of $Q(\mathbf{q}_k, \mathbf{u}_k)$ determined in Equations 3.23 and 3.24 could be stored in a table. **However, robotic motor control generally involves states and actions continuous in time and space. To deal with this, instead of storing the Q-values, we can assume a regression problem** in which $\mathbf{q}_k$ and $\mathbf{u}_k$ are the inputs of a parametric model, such as an ANN, which will be used to compute the Q-value. This technique is called RL in continuous time and space (Doya, 2000),

and it has lead to encouraging results in robotics applications (Lillicrap et al., 2015). However, realistic continuous problems involve random stochasticity, shortened trajectories, or different dynamic properties. As a result, the performance of a given RL algorithm can strongly vary across multiple environments, and evaluating its performance can be subject to discussion (Henderson et al., 2018).

3.5 A Two-Steps Learning Approach

3.5.1 Motivation

RL has been rather popular in robotics research in the last decades because it allows optimizing complex problems based solely on a reward signal, sampled on each time step. However, the optimized policy generally includes a very large number of parameters (the connection weights of a neural network, for instance) to benefit from a great generalization capacity, and the algorithm requires a long learning process. Unfortunately, such an approach ignores biological evidence on locomotion. For instance, gait oscillatory patterns seem to be already present since birth in the mammals (Dominici et al., 2011). Moreover, many variations of RL have been suggested in the literature for tackling different tasks and problems. The practical difficulty of finding a suitable algorithm, adapting it, and training it in order to solve a specific robotic problem should not be overlooked. Also, RL makes the transfer of a control policy from simulation to a real robot very complex. In comparison, the SL has the undeniable advantage of being faster to optimize, given the high feedback rate delivered by the target. It can also more easily integrate prior knowledge into the target signal. Nonetheless, the need to provide this target signal is a crucial drawback, which cannot be overcome by definition.

3.5.2 Concept

To answer this major issue, and based on the work of Degrave, Caluwaerts, et al. (2015), I suggest an original approach presented in Figure 3.12. The concept of this approach is loosely inspired by locomotion evolution in biology. Broadly speaking, the optimization of locomotion patterns can be divided into three phases in nature (Pfeifer et al., 2007):

1. **Evolution:** The morphology and neural system of all animals on earth have been optimized throughout millennia of evolution to enable moving faster on rough terrains using little energy (the phylogenetic evolution). The results are encoded in the DNA and transmitted from parents to children. This means that, even at birth and before any learning process, most animals already possess the neural structures and prior knowledge to help them locomote (Dominici et al., 2011). For instance, this heritage could be the CPGs networks observed at birth in several mammals.

2. **Imitation:** In the first months or years of existence, newborns start to replicate their parents' actions (language, locomotion, manipulation, ...). This kick-starts the learning process of the plastic areas of the brain and considerably reduces the need for exploring all motor capabilities while making sure that the results are adapted to the environment in which the parents are evolving. This is part of the ontogenetic evolution.

3. **Exploration:** While the imitation phase has reached a certain potential, the individual is capable of locomoting in ordinary conditions while avoiding the most severe damages that could affect its body integrity. Concerning locomotion, it can now transit to a phase of exploration where the abilities are refined to adapt in difficult situations (agility on rough grounds, in presence of obstacles, holes, ...) or to allow more accurate control (turning, modifying speed, changing balance, foot lift-off height, ...).

In order to obtain robust locomotion on real robots, I believe that this biological story can be a source of inspiration to implement

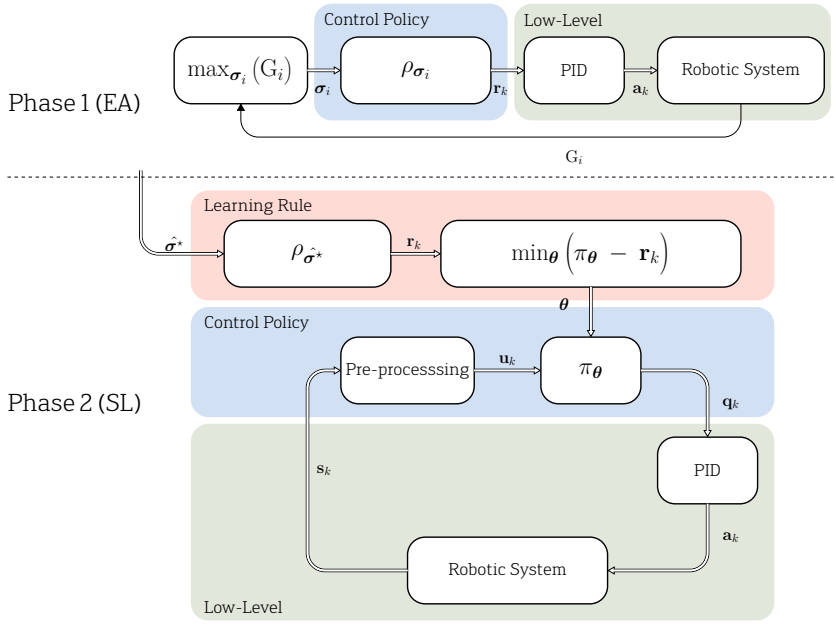


Figure 3.12: In this custom learning approach, a EA optimizes parametric CPG in the first step to maximize a performance score in open-loop. This result is then used as a target to train a policy, such as a neural layer, to control locomotion in closed-loop.

the learning algorithms in the correct order. **In the next chapters, I will use a learning method implementing the two first steps discussed above (evolution, imitation) translated on a robotic level**, presented in Figure 3.12:

1. **A CMA-ES optimization of a realistic parametric model for open-loop locomotion.** The model under optimization used in this step is labeled ρ. It consists of a CPG whose parameters in terms of amplitude, phase, offset, modulation, ..., described with the letter $\boldsymbol{\sigma}$, are unknown. At each iteration i, the CPG parameters $\boldsymbol{\sigma}_i$ are updated to maximize a locomotion score G_i. This step finally results in a set of CPG parameters $\hat{\boldsymbol{\sigma}}^\star$ that can be used for optimal locomotion in open-loop.

2. **A closed-loop SL algorithm, using the CPG signal as a target.** In this step, I use the CPG parameters $\hat{\boldsymbol{\sigma}}^\star$ to produce a target signal $\mathbf{r}_k$. Similarly to Sections 3.4.2 and 3.4.3, the sensor signal $\mathbf{s}_k$ are be pre-processed to provide the variable $\mathbf{u}_k$ using elements like filters, memory buffers, and non-linear projections. The policy optimized in this step is usually a neural layer transforming the signal $\mathbf{u}_k$ into a vector of joint positions $\mathbf{q}_k$.

At this point, it can be important to make a remark about the terminology used in the following chapters. As I mentioned in the introduction, this book is at the crossing of different disciplines and they all have their own conventions and words to describe rather similar concepts. For instance, **in biology**, literature will generally talk about **feed-forward locomotion control** to describe a top-down system where the locomotion patterns and the coordination of the limbs are decided centrally and forwarded to the motor organs (in a similar way as the architecture of the first phase of the methodology presented above). And **reflex-based locomotion** will be employed when the sensor feedback is directly used to regulate the locomotion process without the need for centralized action (similar to the second phase of the methodology described above). In **machine learning**, the second step is sometimes referred to as **feed-forward end-to-end neural control**, i.e. a control process where there is no need for centralized planning but rather

where sensors are indirectly attached to actuators through a feed-forward neural network instance. In **dynamical systems theory**, the same architecture (second phase of the approach described above) connects to the concept of **embodiment**, which describes the tight coupling between sensors and actuators while keeping a simple architecture using little computation. **However, in the rest of this book, I will refer to these two steps above as the 'open-loop' and 'closed-loop' architectures.** These terms should not be wrongly interpreted in the literal sense encountered in control theory and PID controllers but rather in a broader sense of 'the use of sensory feedback or not', perfectly in line with the perspective taken in this chapter.

3.5.3 What it is and what it is not

This two-steps approach is a proof of concept, a demonstrator. It has no ambition to mimick, prove or discard biological evidence, nor to replace current engineering solutions for robotic locomotion. However, if properly backed up with further experimental and theoretical contributions, this work could eventually lead to robust gait locomotion using simple parametric controllers and a short training time.

A direct advantage of this approach is to restrict the problem to a set of biologically plausible solutions, before a direct application on a robot. From a practical point of view, **this approach also greatly reduces the learning time and complexity**. Indeed, the hypothesis on the target's shape confines the training landscape in a very local region of interest. This technique in two steps can also play in favor of an easier transfer by performing the first step in simulation and the next on the real robot. These different aspects are discussed in Chapters 6, 7, and 8. The fact that the methodology combines both a model-based and data-based approach to find efficient gaits is also quite encouraging to bridge the knowledge between biological and engineering worlds. Finally, although the 'Exploration' step in the biological inspiration cited above has not yet been implemented in my work, RL algorithms

could be an ideal candidate for it. This could potentially enhance the end-to-end neural locomotion which was already made stable on flat terrains through the two first steps of EA and SL, while avoiding the sim-to-real transfer problem or damaging the robot too severely. This third step could lead in turn to better adaptability in difficult environments (presence of obstacles, slopes, uneven grounds, ...) but also improve the controllability of the solution (ability to change speed, turning angle, feet lift-off height, ...). This last step is however still an assumption and should be studied in future work as discussed in Chapter 9.

On the other hand, the main drawback of this approach is that **the splitting between an open-loop phase and a closed-loop phase somehow decreases the potential of finding very agile gaits.** Indeed, the robot is forced to hold a restricted cyclic closed-loop regime, also called attractor, which remains quite simple. In the presence of unknown disturbances, the system would either go back to the initial regime of the attractor (this is observed on the MSD structures, Tigrillo and the HyQ robots in Chapters 5, 6, and 8), either completely diverge and the robot will fall (this has been observed on the MSD structures and HyQ in Chapters 6 and 8). To put it differently, it is important to note that, with this architecture, **we cannot expect the closed-loop controller to perform better than in open-loop**, and this assumption will be verified in the next chapters. Similarly, in nature, you would not expect a newborn that learned to locomote from its mother to perform better than her without further individual explorations. The goal of the first two steps is therefore not to provide more performant, robust, or agile results but to showcase that reflex-based control can have the same level of performance as centralized hierarchical control, and that the approach and architecture can lead to an easier transfer of knowledge.

3.6 Locomotion Metrics

Benchmarking locomotion is essential to assess and quantify the added-value of different robot morphologies and controllers, but also to define standardization and regulation processes for industrial applications (Torricelli et al., 2015). **Because of their complexity, there are several ways to define performance criteria for locomotion. I divide them here into two categories: behavioral metrics and algorithmic metrics.**

The goal of behavioral metrics is to establish high-level features to characterize the performance of locomotion as a whole. The general complexity of the gait can generally be assessed by a small group of metrics. For instance, measuring speed is generally the first idea that comes to mind to check the effectiveness of a gait. However, if high speed is taken alone, optimizing a robot to achieve good performance would be biased toward high energy consumption. To deal with this, the concept of Cost Of Transport (COT) has been suggested in Tucker (1975). It represents the ratio of the speed divided by the weight and the power of a robot. Multiple evaluations have demonstrated its consistency across animal species, robotic systems, and other man-made vehicles. Nonetheless, the COT does not take into consideration the adaptability to different environments and potential disturbance. Measuring agility is, therefore, an essential practice, as suggested in Eckert & Ijspeert (2019). The authors introduce a benchmark for quadruped robots' agility using a weighted sum of different metrics, among which the ability in turning, leaping, slope running, standing up, sidestepping, and operating forward and backward locomotion. An important drawback of measuring agility is often the need to use costly material like 3D-tracking systems (Nakamura, 2011). Also, these complex metrics depend on the task under consideration, and the definition can easily vary across robots of different sizes, weights, and designs. To deal with this, another solution suggested in Torricelli et al. (2015) is to benchmark locomotion with a series of physical tasks in an arena. Similarly to how human body capacities have been challenged for thousands of years in

Olympic games, these standardized devices are a promising path to objectively measure high-level abilities in robotic locomotion.

The second category of metrics comprises all the performance of sub-systems, independently from the task they perform. These metrics **are especially relevant in computer science and generally helpful to design, optimize, or validate the choice of parameters and meta-parameters in a controller or a learning algorithm**. However, they generally do not tell anything about the ability of an algorithm to be used for a specific robotic task. This is why they cannot be used alone to draw conclusions in robotics.

3.6.1 Definitions

In this section, I specify some conventions and symbols to name the physical properties of the different robots that are used in simulated and real experiments in the next chapters. Each robot has a mass m that I will consider to be located on the COG of the trunk (i.e., the weight of the legs is neglected in first approximation). The position of the COG, $\mathbf{x}$ is taken in the world frame of reference (with coordinates X, Y, Z) and expressed:

$$\mathbf{x} = \{x, y, z\}, \tag{3.25}$$

where x is determined by the forward direction of the robot at the initial time step, y is the transversal axis in the ground plane and directed to the left, and z is the vertical axis directed to the top. The attitude of the robot body is given by the Euler angles:

$$\mathbf{\Omega} = \{\phi, \theta, \psi\}, \tag{3.26}$$

where ϕ is the roll around the X-axis, θ is the pitch around the Y-axis, and ψ, the yaw around the Z-axis. It is important to remind that the notation $\mathbf{x}$ is also used in the controller to define the full state of the body and the two definitions should not be mixed up.

Depending on the robot, I will consider either angular torque-based actuators or linear spring-based actuators. The angular actuators

are identified with the letter q^j where j varies between 1 and J. The linear actuators used in simulations of MSD networks are activated by modifying the spring reference length l^j. More explanations are provided in the next chapters. The vector of actuator angles is named $\mathbf{q}$, and the angular velocity and accelerations are respectively $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$.

Real experiments and simulations contains different phases for training and testing, identified by the indexes T_{train} and T_{test}. Between these two phases, I will also introduce a phase called T_{closing}, to transit gradually from an open- to a closed-loop controller.

3.6.2 Behavioral Metrics

3.6.2.1 Stability

Stability describes the property of the robotic system to stay in its own limit cycle, despite external disturbances. A stable locomotion induces low oscillation of the trunk in roll, pitch and height oscillations. **The stability is measured by the range of the robot's center of gravity:**

$$\mathrm{S} = \max_k (z_k) - \min_k (z_k) \text{ for } k \in [1, T_{\text{test}}], \tag{3.27}$$

where z_k is the height of the robot's center of gravity at time step k.

3.6.2.2 Power

Computing power consumption involves a good model of the actuators and caloric dissipation in the robot-environment interactions. In the scope of my investigations, I am mainly interested in comparing the consumption when varying different robot or controller parameters. Therefore, I will work with a virtual work power rather than quantifying it accurately from calorific or electrical measurements. **When working with angular actuators, it can be determined by summing up the product of all actuators' torque**

and angular speed:

$$\mathrm{P} = \sum_{k=1}^{T_{\text{test}}} \sum_{j=1}^{T_{\text{test}}} \dot{q}_k^{\,j} \cdot \tau_k^{\,j} , \tag{3.28}$$

where $\dot{q}_k^{\,j}$ is the joint angular speed of actuator j at time step k and $\tau_k^{\,j}$, its measured torque. **When working with linear spring-damper actuators, the power dissipation can be approximated according to Chow (2003) :**

$$\mathrm{P} = \sum_{k=1}^{T_{\text{test}}} \sum_{j=1}^{T_{\text{test}}} k_j \cdot \frac{\alpha^{j\,2}\ l^{j\,2}\,(1+\beta^2\,\alpha^{j\,2})}{4\pi} . \tag{3.29}$$

In this equation, k^j is the stiffness constant of the spring j. The springs are actuated using sinusoidal signals of amplitude α^j. β is the spring nonlinearity factor and q^j are the reference lengths on which one can play to modify the forces and actuate the spring linearly. These elements will be discussed in more detail in Chapter 4, where I present the different robotic platforms.

3.6.2.3 Speed

The robot speed is only evaluated on the X-axis, as deviations on the Y-axis are undesirable for the controller in this work. **It is simply measured by the average speed during the testing phase:**

$$\mathrm{V} = \frac{1}{T_{\text{test}} - 1} \cdot \sum_{k=2}^{T_{\text{test}}} \frac{x_k - x_{k-1}}{t_k - t_{k-1}} . \tag{3.30}$$

3.6.2.4 Cost of Transport

As I have already explained, speed is a poor metric to measure the locomotion performance of a system. **The COT defined in Tucker (1975) uses a ratio of the speed with the mass and the power** to take multiple effects into account:

$$\mathrm{COT} = \frac{\mathrm{V}}{m \,.\, \mathrm{P}} , \tag{3.31}$$

where the speed V and power P are defined above. Nonetheless, while this ratio is really useful to evaluate a system, it is still inefficient when used directly as a reward in an optimization process (for instance, with the EA or RL algorithms defined above). Indeed, to maximize the COT, it is possible to decrease P as much as possible, resulting in robots that will consume really low power but barely move at all.

To counteract this effect, I introduce a saturation of the score at low power using a hyperbolic tangent:

$$\mathrm{COT}^{\mathrm{sat}} = \frac{\mathrm{V}}{m} \, . \, \tanh\left(\frac{\mathrm{P}^{\mathrm{ref}}}{\mathrm{P}}\right), \tag{3.32}$$

in which $\mathrm{P}_{\mathrm{ref}}$ is the reference value to normalize the power. A comparison with the classical COT is given in Figure 3.13. Near the origin, we see that it is possible to obtain high scores with the classical COT. In contrast, the score of the saturated COT has a positive slope along the Y-axis, which prevents optimization algorithms to remain stuck in that region of low power and low speed. Nonetheless, this score presents a major disagreement as it includes the extra normalization parameter $\mathrm{P}_{\mathrm{ref}}$. In other words, the user needs to know in advance in which range of power the robot should operate.

3.6.2.5 Ground Reaction Forces

To quantify the impacts with the ground of a legged robot, **we are interested in the maximal GRF on the four legs during the testing period**. In the following chapters, I only keep the component of the interactions which is normal to the ground, named $\mathbf{f}_k^{\mathrm{GRF},\, i}$ for the leg i at time step k. This has the advantage to reduce the dimensionality of this metric to a single scalar:

$$\mathrm{GRF}^{\mathrm{max}} = \max_{k,\, i} \left(\mathrm{abs}(\mathbf{f}_k^{\mathrm{GRF},\, i}) \right) \quad \text{for } k \in [1,\, T_{\mathrm{test}}] \text{ and for } i \in [1, L], \tag{3.33}$$

with L the number of robot legs.

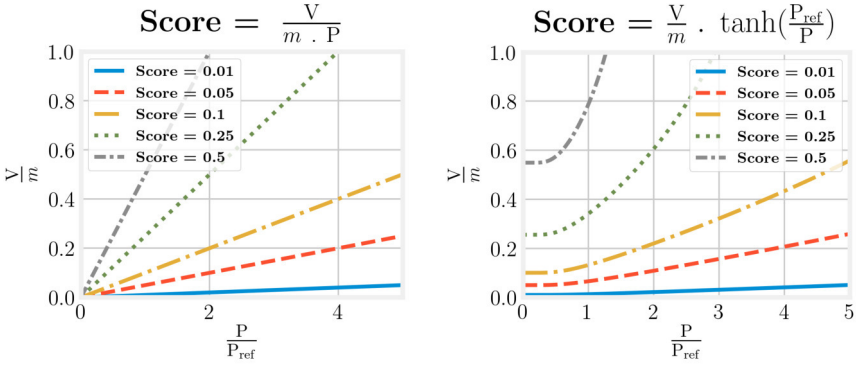


Figure 3.13: The efficiency score to quantify locomotion quality increases with the speed and decreases with dissipated power. In an optimization process, the COT (left) can lead to optima close to the origin, i.e. where the body barely moves. Using an hyperbolic tangent (left) solves this problem for small powers but requires to select P_{ref} carefully.

3.6.2.6 Qualitative Metrics

If the previous metrics already provide a good quantification of the locomotion performance, it is also convenient to build features that can describe the gait qualitatively to understand its internal dynamics.

Locomotive Cycle In legged robots, **the gait can be graphically represented by a closed curve called a locomotive cycle**, which shows the trajectory of the foot position during a full cycle. It provides information about the length of the stance and swing phases, the liftoff and touch-down times, but also the step height and length (or stride). It generally has the shape of a semi-ellipse, flat on the ground but with higher vertical acceleration for the lift than the touch-down, to increase the step height and avoid high impacts on the ground. In closed-loop control, the locomotive cycle can also help to visualize the divergence of a robotic system from its limit-cycle attractor. To represent this curve, **we simply draw the complete trajectory in a two-dimensional X-Z plane in the frame of reference of the robot trunk.**

COG Trajectory **The trajectory formed by the COG displacement during locomotion** is also an interesting feature to represent graphically the body evolution during locomotion. This essentially provides insights into gait stability and the reaction to external disturbances.

3.6.3 Algorithmic Metrics

3.6.3.1 Prediction Error

The algorithmic performance of supervised learning applied to time series forecasting can be estimated using a prediction error metric. Several versions have been suggested in literature (Hyndman & Koehler, 2005). In this dissertation, most experiments deal with the prediction of the robot joint signal q_k^j, given a target actuation signal r_k^j. Because the learning task is carried continuously in an online fashion, it is useful to evaluate the progression of this error metric over a window called a horizon H (i.e. a finite number of points at time steps t_k, t_{k-1}, ..., t_{k-H}) instead of the full duration of the experiment, in order to evaluate if the forecasting task is converging or not. In the experimental chapters 6, 7, and 8, the horizon will be always limited to 1, but larger values are of course possible. **The most straightforward metric available is the mean of euclidean distances between the target and the prediction:**

$$\mathrm{MSE}_k = \frac{1}{H}\frac{1}{J}\sum_{h=k}^{h=k-H}\sum_{j=1}^{J}(r_h^j - q_h^j)^2 \,, \tag{3.34}$$

which we call the Mean Squared Error (MSE). As equation 3.34 suggests, the error is also averaged on all actuators J to obtain a scalar value in this dissertation. In a variant, it is also possible to compute **the absolute distance between the target and the prediction to establish the Mean Absolute Error (MAE)**:

$$\mathrm{MAE}_k = \frac{1}{H}\frac{1}{J}\sum_{h=k}^{h=k-H}\sum_{j=1}^{J}|r_h^j - q_h^j| \,, \tag{3.35}$$

Nevertheless, these two metrics depend on the absolute values of the target and prediction signals. To cope with this, **a normalized version of the rooted MSE, called the Normalized Root Squared Error (NRMSE), has also been suggested:**

$$\mathrm{NRMSE}_k = \frac{\sqrt{\frac{1}{H}\frac{1}{J}\sum_{h=k}^{h=k-H}\sum_{j=1}^{J}(r_h^j - q_h^j)^2}}{(\max(r_h^j) - \min(r_h^j))}. \tag{3.36}$$

In this case, I am normalizing using the minimal and maximal joint positions. This has been chosen heuristically because it provided robust results in my optimization but normalizing the NRMSE with the variance is more common. Other normalizing factors have been also suggested. For instance, we can cite the Mean Absolute Percentage Error (MAPE), and its symmetric version SMAPE, which present the result as a percentage error (Hyndman & Koehler, 2005).

The NRMSE measure is inversely proportional to the accuracy of a system. Therefore, **I will also use the inverse in this dissertation to quantify more clearly the accuracy:**

$$\mathrm{A}_k = \frac{(\max(r_h^j) - \min(r_h^j))}{\sqrt{\frac{1}{H}\frac{1}{J}\sum_{h=k}^{h=k-H}\sum_{j=1}^{J}(r_h^j - q_h^j)^2}}. \tag{3.37}$$

When dealing with periodic signals, like in locomotion, the metrics presented above are strongly dependent on the phase error between the target and the prediction. This can be a major drawback in the testing phase if the predicted values are correct, but the frequency is slightly erroneous and the phase difference between the two signals is drifting. An alternative is to **compute the prediction error in the frequency domain, after applying a Discrete-time Fourier transform on the two signals** over the horizon H. For instance, **we can compute the Frequency Mean Square Error (FMSE)**:

$$\mathrm{FMSE} = \frac{1}{K}\sum_{j=1}^{J}\sum_{f=0}^{F}\left(R_f^j - Q_f^j\right)^2, \tag{3.38}$$

with Q^j being the Discrete-time Fourier transform of the signal q^j which assigns the time window $[t_{k-H}, t_k]$ to a frequency window $[f_0, f_F]$:

$$Q_f^j = \sum_{h=h-H}^{h} q_h^j \,.\, e^{\frac{-i2\pi fh}{H}} \,. \tag{3.39}$$

Another option that allows small phase shifts between the target and the prediction signal consists in **computing the minimal error over a time window**. This technique is based on the Mean Absolute Error (MAE) and **I call it Mean Absolute Windowed Error (MAWE)**:

$$\mathrm{MAWE}_k = \min_{l \,\in\, [k-M,\, k+M]} \quad \frac{1}{H} \sum_{h=l}^{l-H} \mathrm{abs}(r_h^j - q_h^j) \,, \tag{3.40}$$

where M delimits the shifting window on which phase difference is tolerated. In this form, the metric needs information from the future and can only be applied offline, but it is also possible to take the window shortly in the past. The MAWE is used in Chapter 5 with other variables and notations, as the task under consideration deals with the prediction of a sensor signal $\mathbf{s}^{\mathrm{s},\theta}$, given a target $\mathbf{s}^{\mathrm{r}}$.

3.6.3.2 Training Time

The training time is defined as the average time needed to train the network. It can be easily measured, but it is dependent on the programming language, the code implementation, the hardware characteristics, and the operating system load. This makes it hardly significant across different works, although it can be used to compare similar implementations of an algorithm with different parameters, running on the same computer. It is simply measured as:

$$\mathrm{T} \;=\; \frac{\sum_{k=1}^{T_{\mathrm{train}}} t_k}{t_{T_{\mathrm{train}}} - t_1} \,. \tag{3.41}$$

3.6.3.3 Limit Cycle

In closed-loop systems based on learning techniques, the concept of locomotion cycles, introduced in Section 3.6.2.6, highly relates

to the idea of limit cycles, which have been first discussed in Poincare (1881). A closed-loop dynamical system with a periodic activity is often considered as a meta-equilibrium that defines an attractor, in opposition to chaos, in which the behavior is not predictable, or to a steady-state, characterized by a static equilibrium. **A limit cycle is a graphical tool to represent this dynamical attractor.** By plotting two variables of the system on an X-Z graph, it provides a visual grasp of this attractor, and it allows us to check for potential changes, drifting, or disruption. Usually, these two components can have a physical meaning, such as the X and Z coordinates of one foot during locomotion in the trunk reference frame. Selecting latent variables is also a possible option. For instance, we can perform a Principal Component Analysis (PCA) to find the most representative components of a system (Pearson, 1901) and represent its limit cycle efficiently.

CHAPTER 4

—

Locomotion Platforms

Engineering choices are endless in robotics. This chapter presents the main assumptions on the physical and control systems to better define the scope of my work.

In this chapter, I present the robotic platforms on which the experiments are conducted. To carry out the research described in the introduction, **I have worked with three setups to study different aspects of locomotion**. **The first one** consists of networks of massive nodes connected together with spring-damper constraints, also called **MSD networks**. These simulated passive, compliant systems are used to investigate theoretical considerations about morphological computing. **The second setup, Tigrillo**, is a cheap passive, compliant quadruped robot used to train embodied locomotion gaits and test calibration methods to reduce the sim-to-real gap, discussed in Chapter 2. Finally, **the third platform is called HyQ**. It is a large state-of-the-art quadruped robot implementing active compliance, and I have used it to study the effect of compliance and stability control in closed-loop locomotion.

The reason for using different platforms in my Ph.D. has been mostly guided by conjuncture: Tigrillo originated from a Master thesis project and HyQ was fully developed at the *IIT* with other considerations in mind. Nevertheless, while each platform has not been designed as an explicit evolution of the previous one, **they rank chronologically with an increase of complexity and features that made it possible to extend the research conclusions formulated in this thesis at each step**. For instance, migrating from the MSD structures to Tigrillo raised the question of extending the learning method from simulation (see Chapter 6) to the real world (see Chapter 5). And moving to HyQ enabled a more in-depth analysis of the effects of compliance on locomotion (see Chapter 7), but also extended the validity of the two-step learning method presented in Section 3.5 on robots with more actuators and sensors while taking the stabilization into consideration (see Chapter 8).

For each platform presented in this chapter, I provide an overview of the general architecture and some details about the most important features. Hardware specifications, mechanical interactions, and simulation environments are also presented to the reader.

4.1 Mass-Spring-Damper Networks

The simulator for MSD networks is a theoretical framework fully implemented in *Python* and using the *Numpy* library[1]. The MSD networks are inspired by Hermans et al. (2014) and Caluwaerts et al. (2012), and consist of a set of massive nodes, connected by spring-damper links which are all actuated separately. The software provides not only the MSD models but also a simplified physics solver, specially adapted to the system under study. The simulation can be performed either in 2D or 3D.

4.1.1 Simulation Model

The MSD morphology is presented in Figure 4.1. Each of the I nodes, except those at the end or beginning, is sparsely connected to its closer neighbors by C connections. The total number of springs in the network J can be easily deduced with geometrical reasoning:

$$J = \left(I - 1 - \frac{C/2 - 1}{2}\right) \cdot \frac{C}{2} \tag{4.1}$$

Each node $i \in \{1, .., I\}$ is represented by its mass m_i, and the passive parameters for each connection are the spring stiffness k_j and the damper coefficients d_j for $j \in \{1, .., J\}$. The compliance of each spring is defined as the inverse of stiffness $1/k_j$. If not specified, the default values used in the experiments in Chapter 6 are I = 20, C = 6, m_i = 1 kg, k_j = 100 N/m, and d_j = 10 Ns/m.

In the dynamical model, the acceleration, speed, and position of each mass are updated using the force vector $\mathbf{f}^i$ which combines **the gravity force, the spring force, the damping force and the air friction force**:

$$\mathbf{f}^i = \mathbf{f}^{\mathrm{s},i} + \mathbf{f}^{\mathrm{d},i} + \mathbf{f}^{\mathrm{g},i} + \mathbf{f}^{\mathrm{a},i} \ , \tag{4.2}$$

where:

[1] https://github.com/Gabs48/SpringMassNetworks

- $\mathbf{f}^{\mathrm{s},i}$ is the spring force vector applied on the node i and equals the sum of the $j \in \{1, .., C\}$ connected nonlinear springs forces for which the equations can be found in Chow (2003):

$$\mathbf{f}^{\mathrm{s},i} = -k^j \,.\, \frac{\mathbf{l}^j}{l^j} \,.\, \left((l^j - l^{\mathrm{ref},j}) + \frac{\beta}{{l^{\mathrm{ref},j}}^2} \,.\, (l^j - l^{\mathrm{ref},j})^3 \right) . \quad (4.3)$$

  In this equation, $\mathbf{l}^j$ represents the spring length vector and $l^{\mathrm{ref},j}$ its reference length. The variable β is a nonlinearity coefficient which will induce a saturation of the spring force for large extension lengths. It also takes inspiration from the work of Hauser et al. (2011), which demonstrates the importance of these nonlinearities from a computational consideration.

- $\mathbf{f}^{\mathrm{d},i}$ is the damper force vector applied on the node i, which is the sum of the $j \in \{1, .., C\}$ connected dampers:

$$\mathbf{f}^{\mathrm{d},i} = d^j \,.\, \frac{\mathbf{v}_j}{v_j} \,.\, v^j \,, \quad (4.4)$$

  where $\mathbf{v}_j$ is the vector of extension speed.

- $\mathbf{f}^{\mathrm{g},i}$ is the gravity force vector:

$$\mathbf{f}^{\mathrm{g},i} = g \,.\, m^i \,.\, z^i \,, \quad (4.5)$$

  where g is the gravity constant and equals $9.81\ m/s^2$.

- $\mathbf{f}^{\mathrm{a},i}$ represents the drag force induced by air friction. It is assumed proportional to the speed:

$$\mathbf{f}^{\mathrm{a},i} = -\gamma \,.\, \mathbf{v}_i \,, \quad (4.6)$$

  where γ is the coefficient of air friction and equals 0.1 Ns/m. It has been included to avoid unrealistic models with very high speed.

The GRFs are modeled by setting the vertical velocity to zero and the horizontal friction coefficient to infinite. The masses perfectly stick to the ground as soon as they touch it. This is a hard constraint that can impact the nature and the performance of

locomotion. However, it simplifies the study of the body's influence by assessing perfect friction conditions in every simulation.

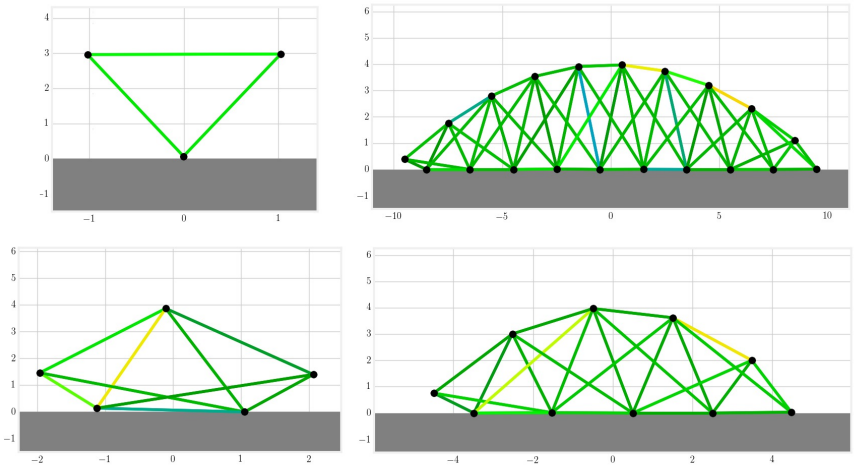


Figure 4.1: The MSD structures are built automatically with a simple morphology that takes the number of nodes and connections as an input. In this figure, structures with three, five, ten, and twenty nodes are drawn in a 2D space. Each black circle represents a mass and each line is a set of springs and dampers in parallel. The colors indicate the current amplitude of actuation.

4.1.2 Gait Controller

To actuate the spring, their lengths l^j are modulated. **The actuation signal is a simple sinusoid**, as practiced in Hermans et al. (2014):

$$l_k^j = l^{\text{ref},j} \,.\, (1 + \, \alpha^j \,.\, \sin(\omega^j \,.\, t + \phi^j)) \tag{4.7}$$

It induces a set of tunable parameters α^j, ω^j, and ϕ^j for each spring in the simulation. In the next chapter, I will discuss how these parameters can be tuned to obtain an emerging locomotion behavior. An example of this behavior is presented in Figure 4.2. It shows snapshots of the MSD networks at different moments, to illustrate how the structures can locomote when properly controlled.

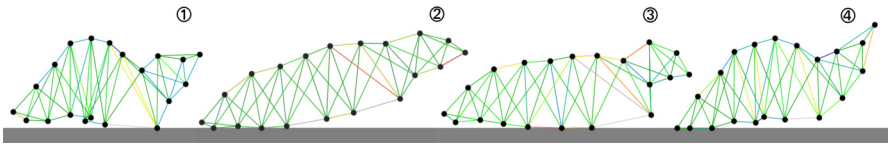


Figure 4.2: After optimization, emergent locomotion patterns with a 'natural appearance' arise on the MSD networks. In this Figure, we see four screenshots describing a full locomotion cycle from left to right.

4.1.3 Physics Solver

The simulation time is discretized using K time steps t_k and equations are solved numerically using the Verlet algorithm described in Thijssen (2007). The Verlet integrator leads to more accurate trajectories, especially for periodic oscillations where energy is rigorously conserved due to the time-reversibility of this operator. For non-periodic trajectories, one can prove that due to symplecticity, the energy does not drift away and errors remain bounded, as demonstrated in Yoshida (1990). Although it is more accurate, the Fourth-Order Runge-Kutta integrator requires four force evaluations per update step and is not symplectic. In my implementation, the update equations are slightly changed to take the effect of the ground reactions into account.

4.2 Tigrillo

4.2.1 Mechanical Design

The Tigrillo platform presented in Figure 4.3 is a small passive compliant robot based on the work of Willems et al. (2017), itself inspired by the Puppy robot (Iida & Pfeifer, 2004) and the Scout II robot (Poulakakis et al., 2005). **It implements four under-actuated knee joints, which can be manually replaced using detachable springs and dampers** to tune the passive compliance properties. A special effort was made to guarantee a **low cost**, a **reproducible design**, and **versatile utilization**. The robot weighs 950 g and fits

in a box of 30 cm by 18 cm. The track widths are 15 cm in the front, and 11 cm between the hind legs. The distance between front and back legs measures 16 cm, providing a stable balance with any configuration of slow gaits. The legs are directly coupled to four *Dynamixel RX-24F* servomotors selected as a compromise between weight, torque, and fast rotation speed.

Figure 4.3: On the left, the quadruped Tigrillo robot used in this thesis for experiments on calibration and transfer of control from simulation to a real robot. On the right, the corresponding parametric simulation model in *Gazebo*.

The use of passive compliance in the knees in place of rigid constraints has experimentally shown a decrease in the optimal COT but this also helps to obtain smoother behaviors in the overall locomotion process (Willems et al., 2017). The same leg principle has also been presented in research involving robots like Tekken (Fukuoka et al., 2003), Puppy (Iida & Pfeifer, 2004), or Bobcat (Sprowitzt et al., 2013).

4.2.2 Electrical Design

Following the same constraints on reproducibility and cost, **the electronics stack is made from three off-the-shelf boards**. A top view of the robot with its electronics is presented in Figure 4.4. The first element is a DC step-up voltage converter that supplies the other boards and motors with a 12 V regulated voltage and a stalk current that can rise to 10 A when the legs are pushing

together, and the motors have to deliver a high torque. Secondly, an *OpenCM* board is used to read the analog sensor values and send the position or velocity commands to the servomotors. The computer board is a *Raspberry Pi 3* running the *Robot Operating System* (ROS) (Garber, 2013) and streaming actuation and sensor signals to a computer over ROS topics.

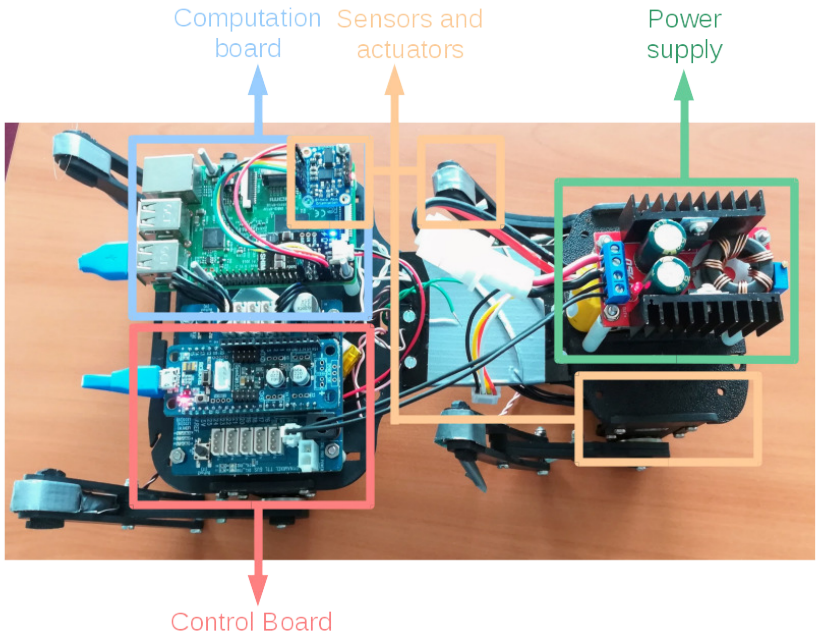


Figure 4.4: A top view of the electronics embedded on the Tigrillo robot.

Another version embedding neuromorphic hardware has been designed and built to enable research on control using spiking neural networks (Vandesompele, Urbain, wyffels, & Dambre, 2019). The robot presented in Figure 4.5 directly embeds a four chips *SpiNNaker* board connected to a desktop computer through a *SpinnIO* board over a Wi-Fi serial connection.

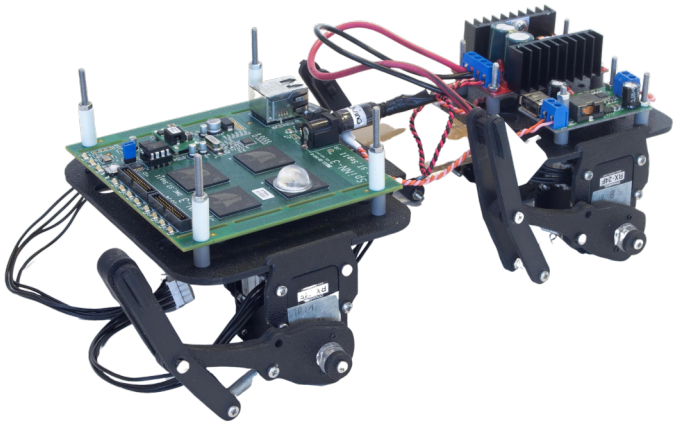


Figure 4.5: Another version of Tigrillo embeds a SpiNNaker board to generate open-loop control using SNN.

4.2.3 Sensors Design

One important aspect regarding the application of the morphological computation theory is **to extract nonlinear feedback from the robot compliant morphology** (Hauser et al., 2012). To this end, **permanent rare-earth magnets are attached to the lower parts of each leg and analog 'Hall sensors' to the higher parts**. The setup presents an advantage in price and does not obstruct the joint movement. This sensor is principally triggered by the Hall effect to output a voltage between 0 V and 5 V proportionally to the surrounding magnetic field. As a first approximation, this field is decreasing with the cubic value of the distance to the magnet. This distance itself varies with the square root of the cosine of the knee angle, according to the generalized Pythagorean theorem. As a consequence, the variation of the sensor voltage is high for small angles, and the order of magnitude of the accuracy is around 0.01°. However, it decreases quickly with the knee angle to reach approximately 1°when the legs are fully extended. For each leg, a conversion table between the sensor value and the measured angle is used to interpolate the transfer function of the sensor pre-

sented in Figure 4.6. All the curves have a shape in line with the theoretical expectations, but the manual fixation of the magnet and the Hall sensor leads to a different offset for each leg.

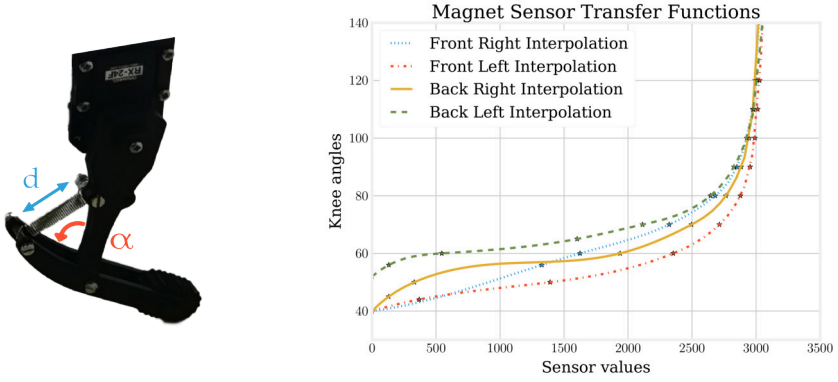


Figure 4.6: On the left, a close-up picture of the leg shows the direct coupling with the *Dynamixel* motor and the non-actuated knee joint embedding the angle sensor. On the right, the four sensor calibration functions are represented. Differences in offset and shape of the curves come from the manual fixation of the magnet and the Hall sensor.

4.2.4 Simulation Platform

To perform the simulation experiment, **Tigrillo is fully integrated into the Neurorobotics Platform (NRP). The NRP is an open-source web-based simulator designed to conduct experiments in neurorobotics.** It contains a closed-loop engine in its core to synchronize SNN simulations using *NEST* and robotic simulations on *Gazebo*, which can run with different physics engine like *ODE*, *Bullet* or *OpenSim*. The conversion between spikes and motor or sensor signals is based on the *MUSIC* framework and a custom domain-specific language described in Weidel et al. (2016). The NRP has demonstrated its capacity to simulate large-scale models of the entire mouse brain on the one hand, and accurate biomechanics of the mouse body using soft physics on the other hand. It is also enhanced with many user-oriented features to easily create experiments in neurorobotics.

4.3 HyQ

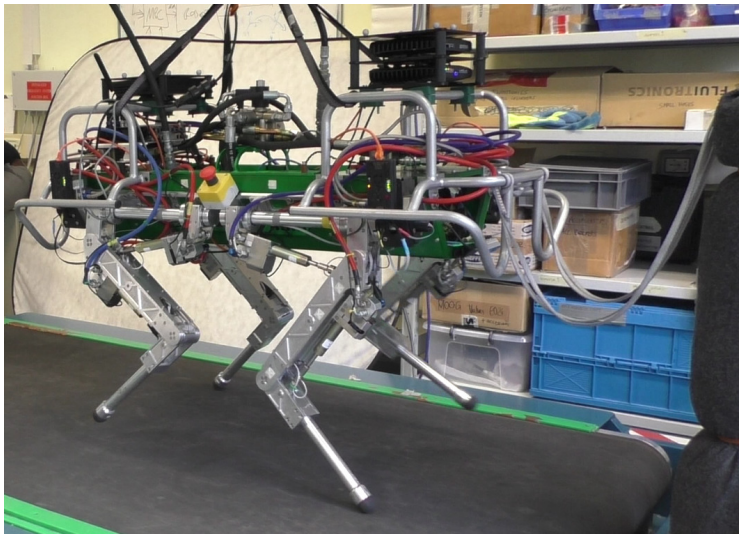
4.3.1 Robot Design

The robot HyQ and its simulation model are both represented in Figure 4.7. **HyQ is a state-of-the-art hydraulically powered quadruped platform** of 1.3 m length and 90 kg weight (Semini et al., 2011). Its design allows the implementation of versatile gaits ranging from static to dynamic tasks like running and jumping.

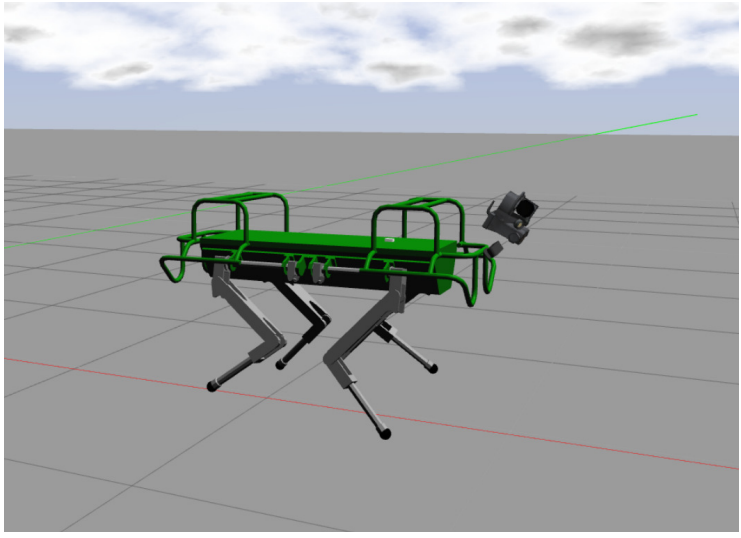
HyQ has four legs with three DOFs each, named Hip Abduction-Adduction (HAA), Hip Flexion-Extension (HFE), and Knee Flexion-Extension (KFE). All joints are hydraulically actuated. The advantage of this feature regarding the current work is twofold: first, the joints are capable of delivering or dissipating high torques (Semini et al., 2011) which allows fast actuation and makes the robot particularly robust for testing a feed-forward neural network controller, prone to oscillating behaviors that lead to larger GRFs; secondly, the actuation system can virtually produce adjustable levels of damping and stiffness.

4.3.2 Impedance Controller

The implementation of a torque controller for HyQ joints has been presented in Boaventura, Semini, et al. (2012) and Focchi et al. (2012) and its performance and stability when compensating external forces have been addressed both in simulation and on the real robot (Boaventura et al., 2013; Boaventura, Focchi, et al., 2012). **To understand the role of the impedance controller, the joint position control loop can be replaced by an equivalent rotational spring-damper system** (Semini et al., 2015). This operation is represented for the one single joint in Figure 4.8. To this aim, I first hypothesize that the effect of the 'Inner Torque Control Loop' is negligible compared to the oscillation of the impedance controller sampled at 250Hz, which is true for the impedance gains considered in this dissertation. The equation of this controller can be



(a) The real HyQ robot



(b) The simulated HyQ robot in *ROS Gazebo*

Figure 4.7: HyQ is a hydraulic-actuated robot measuring 1.3 meter long and weighting approximately 90 kilograms. It can walk (Focchi et al., 2020) and trot (Barasuol et al., 2013) robustly on uneven terrains with different heights obstacles thanks to its visual, torque and inertial sensors and the fast reactivity of its actuators.

therefore written:

$$\tau^{\mathrm{f}} = \tau^{\mathrm{ext}} + \ k^{\mathrm{p}} \ (q - q^{\mathrm{s}}) + \ k^{\mathrm{d}} \ (\dot{q} - \dot{q}^{\mathrm{s}}) \ , \tag{4.8}$$

where τ^{f} is the final motion torque applied by the controller on the actuators, τ^{ext} represents any external torques on the actuators (resultants of the GRFs, external stabilization controller, disturbances), k^{p} and k^{d} are the proportional and derivative gains of the PD controller, q and $\dot{q}$ are the desired joint position and velocity signals delivered to the impedance controller and q^{s} and $\dot{q}^{\mathrm{s}}$ the real joint position and velocity. Besides, the equation for an equivalent

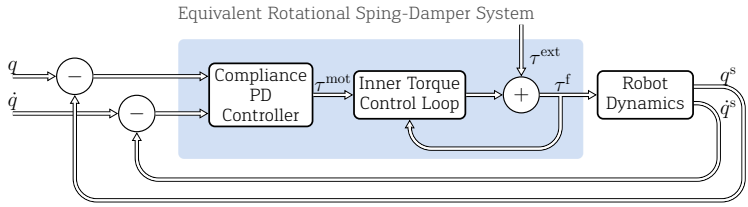


Figure 4.8: The torque controller of the leg's actuators allows to reproduce accurately virtual stiffness and damping properties. The speed of the hydraulic actuation and the fast PD loop is a key factor to its high performance.

spring-damper system can be written:

$$\tau^{\mathrm{mot}} = \tau^{\mathrm{ext}} - \ k \ (q^{\mathrm{s}} - q^{\mathrm{ref}}) - \ d \ (\dot{q}^{\mathrm{s}} - \dot{q}^{\mathrm{ref}}) \ , \tag{4.9}$$

where k and d are the stiffness and damping. A comparison between Equations 4.9 and 4.8 implies that the proportional and derivative gains k_{p} and k_{d} have respectively the same units as the virtual torsional spring stiffness k measured in N.m/rad and the damping c in N.m.s/rad, for which one would vary the reference angle q^{ref} and the reference rotational speed $\dot{q}^{\mathrm{ref}}$ during actuation. This representation of the actuation as a variation of the spring reference length can be put in parallel with the linear actuation of the MSD structures described in Equation 4.7.

4.3.3 Dynamic Gait Controller

The target **trotting gait of the HyQ robot is generated using a simplified version of the Reactive Controller Framework (RCF)** (Barasuol et al., 2013). In the simulations, the robot forward velocity is set to 0.25 m/s with a gait frequency of 1.7 Hz, a trot duty factor of 0.5, the desired step length of 7.3 cm, and a step height of 10 cm. These parameters are chosen heuristically to provide a robust trotting gait with slow speed and narrow foot trajectories. In the real experiments, the frequency has been lowered to 1.4 Hz with a duty factor of 0.6. This does not affect the comparison but helps to monitor the robot better to avoid inadvertent damage.

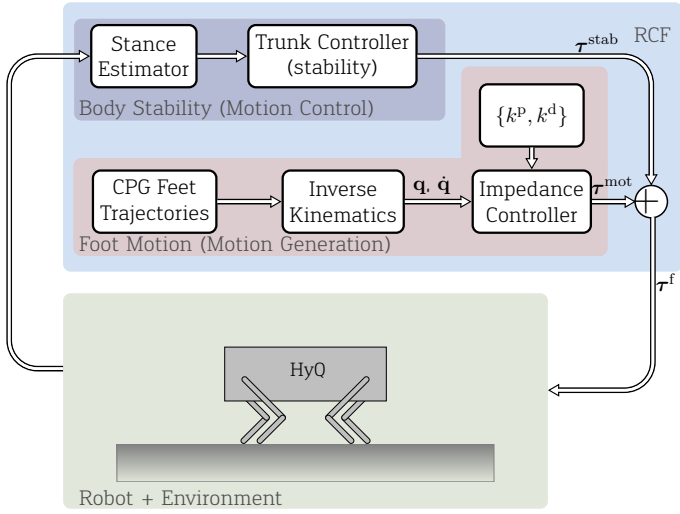


Figure 4.9: The RCF controller of HyQ is made of a two parts to control the foot movement and regulate the robot's posture.

In the core of the RCF controller presented in Figure 4.9, a module produces **four 2D CPG-inspired trajectories in the reference coordinates of each foot**. They **are subsequently projected in the frame of reference of the robot's trunk**. In this dissertation, the correction of the foot trajectories according to the robot's attitude ('Kinematics Adjustment' module of the RCF) has been deactivated to keep an open-loop behavior. Finally, the desired joint positions $\mathbf{q}$,

velocities $\dot{\mathbf{q}}$ and accelerations $\ddot{\mathbf{q}}$ are computed using inverse kinematics and transformed into joint torques via the PD impedance controller. **The robot body is stabilized in roll, pitch, and height using a module called 'Trunk Controller'.** It also produces joint torques that add up to the RCF contribution and regulate the stability of the robot. Other adaptations of the trotting gait using visual inputs for better foot placement detailed in the work of Barasuol et al. (2015) and Villarreal, Barasuol, Camurri, et al. (2019) are not used in this dissertation.

4.3.4 Simulation Platform

The control and simulation software of HyQ and its technical implementation have been discussed in Frigerio et al. (2017). The low-level simulation and motor control are based on the *SL simulator* (Schaal, 2009). The major advantage of the *SL simulator* is its ability to be deployed either on a computer connected with the real hardware or a simulation model. Also, its implementation is compatible with hard real-time constraints, which makes it executable on the *Xenomai*-based *Linux* operating system. A higher-level software layer running on *ROS* provides the packages for high-level control based on the *ROS control* architecture (Chitta et al., 2017), and the *Gazebo* simulator.

CHAPTER 5

—

Calibration for Sim-to-Real Locomotion Transfer

Bridging the gap from simulated to real compliant robots when learning locomotion is challenging. This chapter suggests a method in this direction.

5.1 Introduction

Addressing the issue of transferring a locomotion controller from simulation to reality has been introduced in Chapter 1 as one of the three main goals in this dissertation. This open question has driven much attention in the last years. Indeed, some of the most famous work related to the learning of control policies for locomotion have been conducted in simulation on rigid robotic platforms (Peng et al., 2017; Heess et al., 2017) because the training methods are generally too long and laborious to be conducted in real robots without wearing or damaging them. However, **efficient transfer of a policy learned in simulation to a real robot is critical, because the lack of accuracy and realism in simulation can easily lead to an important failure in the real environment.** In Chapter 2, this issue was described with a short review of the recent work in the domain. The first path for improvement consists of increasing the simulation accuracy. However, this shifts the problem towards extra modeling work. The second leverage is to use randomization processes to obtain robust policies that can deal efficiently with the transfer procedure, but this option directly affects the learned policy. An intermediate idea is based on learning the robot dynamics from real data. Such a solution has the advantage to avoid complex calibration procedures while preserving the learning method.

In this chapter, I assume that the embodiment theory, and in particular morphological computation in its larger sense, i.e., as presented in Füchslin et al. (2013), **may be a preliminary solution in this direction.** In this framework, body, controller, and their intricate relation are analyzed from a dynamical perspective. Each entity can be modeled as a non-linear filter with a computational capacity rather than explicitly in a kinematics parameter space. From that point of view, a good simulation works with an accurate representation of the transfer functions between actuation and sensor signals rather than a detailed physics implementation. This can be obtained through automated optimization rather than fine-tuning of model parameters.

This chapter introduces an automated calibration method for a simulation model that enables an optimal transfer of a controller to a mechanical platform. **Rather than trying to replicate the robot physics rigorously, a parametric model is optimized to maximize the similarities between simulation and real-world body sensor-to-actuator transfer function. The work is conducted on the passive compliant robot Tigrillo.** The first section presents the methodology followed in this chapter. The second section shows the results of two experiments: the first is an evaluation of the calibration method, and the second concerns the performance of an open-loop controller trained in simulation and transferred on the real robot with this method. Finally, conclusions and perspectives on the next research steps are given in the last section.

This chapter is strongly based on the following publication:

> **Urbain G.**, Vandesompele A., wyffels F., and Dambre J. (2018) Calibration Method to Improve Transfer from Simulation to Quadruped Robots. In: *International Conference on Simulation of Adaptive Behavior, 102-113.*

5.2 Calibration Method

The calibration method[1] has been tested on the Tigrillo robot described in Section 4.2. **The goal is to tune some parameters of the Tigrillo simulation model, in order to optimally match the simulated body's response to the real robot's response, as observed by the robot's sensors, when an actuation signal is applied.** To this goal, I select the set of parameters $\boldsymbol{\theta}$ both for their importance in the locomotion behavior but also because they are hard to measure or model accurately:

$$\boldsymbol{\theta} = \{k^{\mathrm{f}}, k^{\mathrm{h}}, \epsilon^{\mathrm{f}}, \epsilon^{\mathrm{h}}, \mu^{\mathrm{f}}, \mu^{\mathrm{h}}, m^{\mathrm{dist}}\}. \tag{5.1}$$

In this notation, the index f refers to the front legs of the robot, and h refers to the hind legs. The parameters k are the spring

[1] https://github.com/gurbain/tigrillo2

constants in the knee joints, expressed in N/m. These parameters are chosen because the complexity of the spring models in the physics engine is unable to render the non-linear effects, which can lead to insufficient results, as discussed further in the next section. The parameters ϵ are the contact depth coefficients expressed in m and represent how much two rigid bodies can overlap during simulation to compute the friction forces. They directly interfere with the static friction coefficients μ, and good manual tuning generally requires an empirical comparison with the real robot. Finally, the total mass of the robot is fixed and determined by weighing the robot, but the distribution ratio between the front and the back is represented with m^{dist}. Many other parameters like the damping values in the knee joints, the minimal value of knee angles, the kinematic friction coefficients of the feet and the motor characteristics parameters have been evaluated in this research but none of them have shown significant improvements with respect to the uncalibrated model. They were therefore discarded.

Figure 5.1 shows the architecture of the optimization process used for calibration. **The robot is actuated in open-loop with four sinusoidal signals** $\mathbf{a}_k$, with the same amplitude and phase for all legs such that its body will alternate between standing-up and sitting-down. The sensor values of the robot $\mathbf{s}^{\text{r}}_k$ are recorded to estimate the robot transfer function $\mathcal{F}^{\text{r}}$ between the actuators and the sensors:

$$\mathbf{S}^{\text{r}}_f = \mathcal{F}^{\text{r}}(\mathbf{A}_f), \tag{5.2}$$

where $\mathbf{A}_f$ and $\mathbf{S}^{\text{r}}_f$ are the Discrete-time Fourier transforms of the robot motor and sensor signals $\mathbf{a}_k$ and $\mathbf{s}^{\text{r}}_k$.
An optimization of the parameters $\boldsymbol{\theta}$ is performed with CMA-ES, as formulated in Hansen (2016). The algorithm generates simulation models with different sets of parameters and estimates the optimal one, i.e. which minimizes the error function $\mathbf{e}^{\boldsymbol{\theta}}$:

$$\hat{\boldsymbol{\theta}^{\star}} = \arg\min_{\boldsymbol{\theta}} \ \mathbf{e}^{\boldsymbol{\theta}}. \tag{5.3}$$

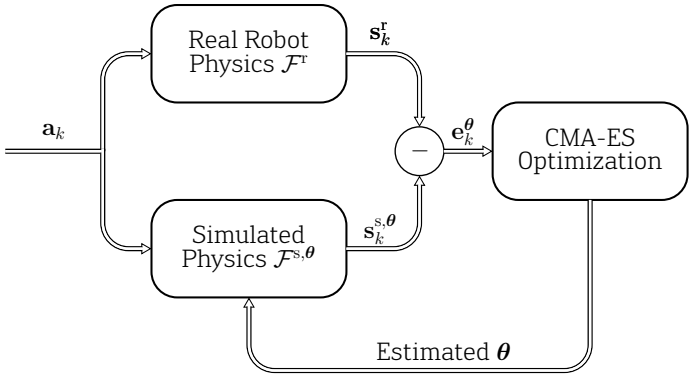


Figure 5.1: The calibration is represented on this diagram. Sensor values recorded on the robot are used to optimize the unknown morphology parameters of the simulation model with CMA-ES.

The error function is chosen to represent the difference between two temporal signals but shall also allow invariance against a possible slight phase shift. This invariance can be obtained by computing the MAWE presented in Section 3.6. To this goal, the MAE is estimated for different configurations where the robot sensors signal $\mathbf{s}_k^{\mathrm{r}}$ and the simulation sensors signal $\mathbf{s}_k^{\mathrm{s}}$ are shifted within a time window. The MAWE is obtained by taking the minimal error in this window:

$$e_k^{\boldsymbol{\theta}} = \min_{l \in [k-M,\, k+M]} \frac{1}{H} \sum_{j=1}^{J} \sum_{h=k}^{k-H} \mathrm{abs}(s_h^{\mathrm{r},j} - s_h^{\mathrm{s},\boldsymbol{\theta},j}) \;, \tag{5.4}$$

where H is the horizon of the measure, which is taken as the number of samples measured for the two signals during a full period (see later in Figure 5.2a), and M is the boundary of the shifting window. This window defines the phase shift tolerance to measure the minimal error between the two signals. J is the number of knee joints (which is equal to 4) and I sum all sensors to obtain a scalar. I could have divided this sum by J to obtain the average but, since J is fixed, this is fully equivalent in an optimization process.

Finally, it is assumed that the parameter set that is obtained in the optimization corresponds to a good approximation of the real robot

transfer function $\mathcal{F}^{\mathrm{r}}$:

$$\mathcal{F}^{\mathrm{s},\hat{\boldsymbol{\theta}^{\star}}} \approx \mathcal{F}^{\mathrm{r}}. \tag{5.5}$$

To decrease the noise on the sensors, which is caused by different factors like some electro-magnetic perturbations or the undesirable residual vibration coming from the motors, **the signal is segmented using the zero-crossing of the periodic actuation signal as a threshold. All the sensor segments are then** projected on a one-period domain with a fixed number of points to be easily **averaged and compared** (see Figure 5.2a).

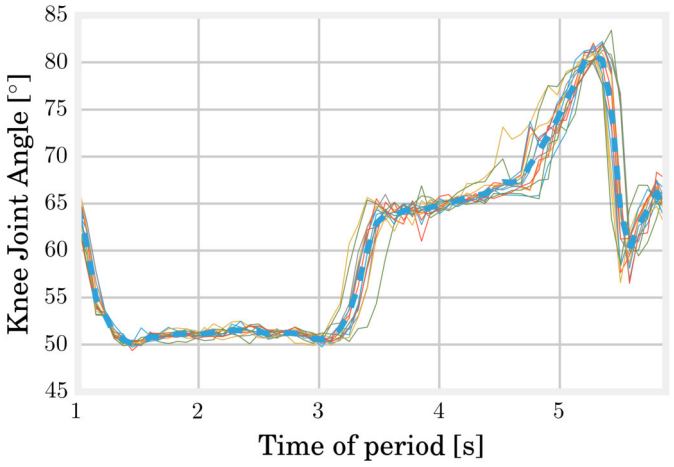
5.3 Results

5.3.1 Calibration of the Simulation Model

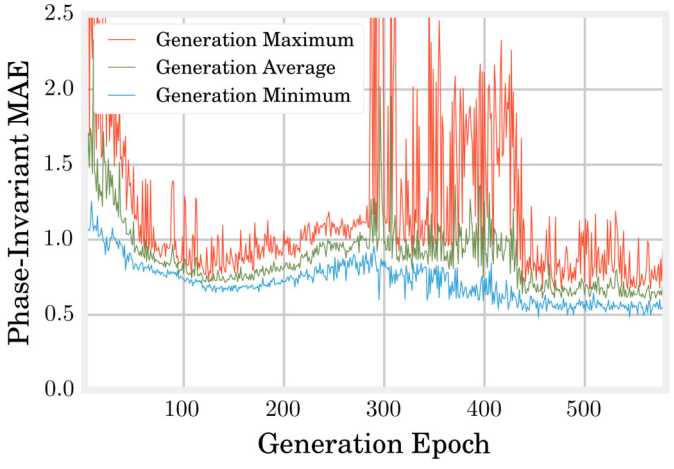
To perform the calibration method in practice, the robot is actuated with a 0.3 Hz sine wave for one minute and the real motor signals and the sensor signals are recorded during the last 50 seconds to provide a reference for the optimization (i.e. after the annulation of the initial transitory effect). After this step, a CMA-ES algorithm is run on a computer. Different initialization values have been tested, but a low variance was observed in the results. For each algorithmic epoch, a population of 10 individuals is generated: each of them has a different morphology sampled in the distribution of $\boldsymbol{\theta}$ provided by CMA-ES and is simulated with the same actuation pattern as the real robot. The optimization evolution presented in Figure 5.2b shows a saturation of the MAWE error to 0.5 after more than 500 generations.

To validate qualitatively this optimum, the NRP is used to control both the simulation model and the real robot, as described in section 4.3.4. A variety of actuation sinusoids with different amplitudes are applied to the motors. **A visual comparison indicates a good correlation between the simulation and the real observations.** Figure 5.3 shows a screenshot of the video illustrating this experiment[2]. Interestingly, failures like stumbling or

[2] https://youtu.be/CqpkC630fJA



(a) Front Left Sensor Period Average



(b) CMA-ES Evolution

Figure 5.2: An overview of the optimization method evolution. On the top, I represent the sensor signals recorded on the robot and segmented periodically to obtain a smoother average for scoring the optimization. On the bottom, the graph shows the evolution of the CMA-ES algorithm through successive generations. It converges a first time around the 100th generation then start an exploration phase to find a better minimum at the 500th generation.

falling are also observed simultaneously both in simulation and in the real world, which is very useful as the role of pre-training in simulation is to exclude actuation patterns that lead to instability of the physical robot.

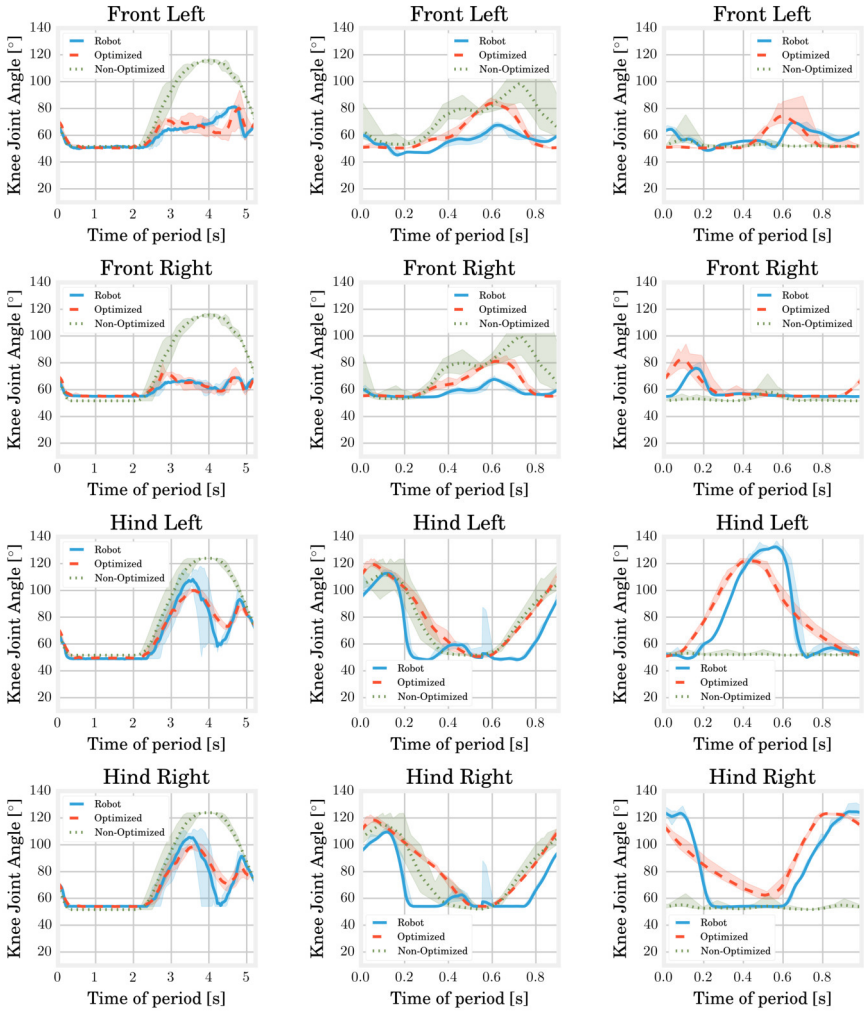
Figure 5.3: A picture of Tigrillo controlled through the NRP. The result of the simulation in the NRP can be seen on the screen in the background and corresponds to the observations on the robot.

To further understand these results, a single sine wave of 2 Hz (the same as during the optimization) is given to the motors. The average sensor signals over one period are plotted for the real robot, the optimum morphology in simulation, and the initial default morphology in simulation (see Figure 5.4a). The latter model is made with my best knowledge of the robot mechanical parameters and simulator parameters and serves as a baseline. A comparison of the parameters before and after optimization is also given in Table 5.1. It is quite obvious in the figure that **the sensor signals produced with the optimized simulation model are closer to the observation on the robot**. There are two main reasons to explain this. First, **the default parameters are unable to reproduce the friction forces between the feet and the ground correctly**. This

effect depends both on the friction coefficient μ but also the contact depth between two rigid bodies in simulation, which has no concrete meaning in the physical world. During testing, when the robot lifts its body off the ground, the COG moves forward above the supporting point of the front feet. The feet suddenly slip on the ground, and the knee angle value drops before increasing again when the robot moves back to the ground, and the COG is shifted backward. This phenomenon is observed on both the real robot and the optimized model but not with the default parameters. Another reason concerns the spring stiffness, which seems too low for the default morphology, although it has been determined by actual measurements of the components. This is caused by the limitation of the simulator, which only allows modeling the physical spring via an equivalent torsional spring linear model in the joint. **At low spring values, the model is quite accurate, but at higher values, the non-linearities induce a saturation on the real robot, which cannot be simulated consistently. The optimization copes with this problem** by converging to a larger stiffness in the hind legs as displayed in Table 5.1.

Parameters	No Optim	Optim
Hind Contact Depth Coefficient (mm)	0.5	7.8
Hind Friction Coefficient μ	0.1	0.000819
Hind Spring Stiffness (N/m)	181.6	440.629
Hind Mass (kg)	0.238	0.276
Front Contact Depth Coefficient (mm)	0.5	3.8
Front Friction Coefficient μ	0.1	0.283
Front Spring Stiffness (N/m)	181.6	181.4691
Front Mass (kg)	0.712	0.674
Mean Absolute Windowed Error	3.231	0.483

Table 5.1: This table shows the calibrated morphology parameters before and after optimization. If the mass distribution and the front spring stiffness do not change significantly, the hind knee joints become much stiffer to counteract the spring saturation on the real robot and the friction with the ground decreases in the front and increases in the back to render correctly the general movement.



(a) Optimization **(b)** Bounding gait **(c)** Walking gait

Figure 5.4: Sensor signals of the real robot (blue), the optimized simulation model (red) and the default model (green) during optimization, bounding gait and walking gait. The signal is closer to the real robot after optimization due to stiffer springs and more realistic trajectories (left). The optimized model is closer to the real robot during the bounding gait (middle) but fails to reproduce correctly the angle variation speed. For the walking gait (right), the optimized signal gives good results despite a phase shift due to a communication delay. However, the non-optimized model directly falls with this gait.

5.3.2 Validation with Open-loop Gaits

In order to corroborate the preliminary observations on the calibration method described in the last section, different open-loop controllers have been trained in simulation and we observed how the transfer performed on the real robot.

The controller is modeled by four coordinated CPGs with the equations introduced in Righetti & Ijspeert (2008). Constraints on the frequency and the phases between the different legs are added to obtain motor primitives for walking and bounding gaits at 1 Hz. The other CPG parameters are obtained by running a new CMA-ES optimization on the optimized simulation model but this time using the robot speed as a score. This optimization helps to explore the motor space to find the most stable gaits to locomote along a straight line. The resulting controller does not make use of sensor inputs, and only the actuation and the robot mechanical properties are discussed hereafter. Some videos can also be found online[3] to get a better understanding of the locomotion behavior. The two optimization processes should not be mixed up. In the previous section, I optimized the transfer function between the simulation and the real robot, whereas in this section I optimize the CPG parameters in simulation, as suggested in the first step of the method presented in Section 3.5, to test them on the real robot and validate the calibration.

5.3.2.1 Bounding gait

To come up with a bounding gait, a constraint is added on the CPGs phases such that the hind legs are going through stance and swing phases synchronously and the front ones as well. This gait is inherently very stable as it does not involve any movement perpendicular to the walking direction. Figure 5.4b compares the value of the knee angle on the real robot as well as the simulation default and optimized models. **The optimization introduces better value ranges for the legs but also much more homogeneous results, as highlighted by the larger variance of the non-optimized sig-**

[3] https://youtu.be/zCHRWxfo0MU

nal. However, it seems that the simulation fails to render the fast transition measured on the robot for both models. In future work, the optimization process should involve higher frequencies to solve this issue.

5.3.2.2 Walking gait

The walking gaits are obtained by optimizing all CPG parameters with constraints on the phase and frequency to make sure that the legs diagonally opposed have the same phase. These gaits are not especially stable as no efforts were made to optimize the robot's balance in the lateral axis, and the robot does not allow foot retraction during locomotion. However, **good performance was observed on the real robot and with the optimized robot simulation model** (see the curves in Figure 5.4c). A phase shift of approximately 100 ms can be explained by the delay introduced by the communication line and the motor inertia. I did not succeed to collect results on the non-optimized model for this specific gait, as the robot directly felt on the ground in the simulator. This is not surprising but indicates a good correlation between the optimized model and the real robot that does not exist with the default model.

5.4 Discussion

In this chapter, I have investigated a method to calibrate the simulation model of a cheap passive, compliant quadruped robot called Tigrillo in order to efficiently transfer motor primitives learned in simulation to the real robot. To derive a usable model of the mechanical platform, it was not chosen to configure the simulation models manually but rather to use a method inspired by morphological computation to calibrate a parametric simulation model with optimization techniques.

This method allowed to reduce the amount of knowledge and hypotheses introduced in the design process. This is especially true for parameters related to friction and passive compliance, difficult to accurately measure on the real robot, and yet extremely impact-

ful in the locomotion simulation. From the first observations, **the method could compete with classical techniques for simulator calibration based on accurate robot modeling, while decreasing the number of measurements on hardware. However, some limitations quickly appeared in robot configurations at the edge of the spring saturation. It advocates for combining classical simulators with external data-based models, which can render these effects correctly**. Such an idea has been recently exploited in Hwangbo et al. (2019). In that study, the authors relied on an extra layer between the simulator and the controller to match the behaviors in simulation and on the real quadruped robot. Although it has a clear advantage in accuracy and convergence speed, this strategy, however, requires a more complex architecture. Among the other paths for improvement, our approach could be generalized for a larger range of actuation frequencies to get a more significant optimization score. Also, the transfer could be characterized against behavioral metrics presented in Section 3.6.

Additionally, the results encouraged me and my colleagues to evaluate the potential of randomization to reduce the sim-to-real gap on the Tigrillo robot (Vandesompele, Urbain, Mahmud, et al., 2019). To this goal, Gaussian noise was applied to the physical parameters of the robot during the optimization of open-loop gaits with CMA-ES. In this paper, we concluded that body randomization in simulation increases the chances of finding gaits that function well on the real robot but it also stressed the strong sensibility of the spring stiffness parameter in the final results.

In this chapter, I also demonstrated the effectiveness of the first step of the method for learning embodied locomotion, presented in Section 3.5. Parametric CPGs for walking and bounding gaits, optimized in simulation with CMA-ES, displayed a realistic behavior on the real robot. In light of this observation, I will apply the same learning principle in the next chapters.

CHAPTER 6

—

Effects of Compliance on Mass-Spring-Damper Networks Locomotion

6.1 Introduction

In the previous chapter, I have demonstrated the optimization of walking and trotting gaits on a small quadruped platform. As discussed in Section 2.3, it is often claimed that the mechanical compliance property can become a strength in locomotion if properly outsourced. On the one hand, the use of flexible materials can lead to higher power efficiency and more fluent and robust motions. On the other hand, using embodiment in a closed-loop controller, part of the control task itself can be outsourced to the body dynamics. This can significantly simplify the additional resources required for locomotion control. However, the framework for a theory allowing a deep understanding of such control systems (and hence engineering opportunities) is still under construction. Morphological computation can be considered as a key concept in this direction (Paul, 2006). **In Hauser et al. (2011), the author suggested the use of MSD networks to study the relationship between physical properties of a body on the one hand (stiffness, damping, complexity, ...) and the computational requirements on the other side. In this chapter, I extend this study to locomotion applications with the MSD networks.**

To this goal, a small, scalable simulation setup was designed to study the effects of compliance on the locomotion of MSD networks empirically. In the first section, I investigate the influence of the network size and compliance on locomotion quality and energy efficiency by optimizing an external open-loop controller using evolutionary algorithms. This analysis helps to evaluate the potential of compliance for locomotion in terms of robustness, efficiency, and stability. Three main experiments are conducted. The first gives an overview of how increasing the number of nodes in a MSD network leads to more stable locomotion. The second experiment provides an analysis of the optimal frequency range for the setup, and the third experiment explores the maximal speeds that are achievable for different driving powers and underlines the limitations of the design to obtain high performance. **We find that larger networks, i.e. structures with more nodes and**

springs, can lead to more stable gaits and that the system's optimal compliance to maximize the traveled distance is directly linked to the desired frequency of locomotion.

In the second section, I analyze the computational capacity of a MSD body to generate motor control signals and I integrate them as regulation feedback to a forward controller. Since maximally efficient actuator signals are clearly related to the natural body dynamics, in a sense, the body is tailored for the task of contributing to its own control. Using the same simulation platform, I then study how the network states can be successfully used to create a feedback signal and how its accuracy is linked to the body size.

Finally, a discussion on the results is formulated in the last section. Benefits of compliance and complexity for simulated locomotion structures are summarized, and the potential and challenges for transposing these results on a real robot are briefly discussed in connection with Chapter 7.

This chapter is strongly based on the following publication:

> **Urbain G.**, Degrave J., Carette B., wyffels F., and Dambre J. (2017). Morphological Properties of Mass-Spring Networks for Optimal Locomotion Learning. *Frontiers in Neurorobotics, 11, 16*.

6.2 Open Loop Experiments

This section presents three experiments to assess the influence of the MSD network size and compliance on locomotion speed, power consumption, and noise robustness. The mathematical models describing the MSD structures and their control have been introduced in Section 4.1. In the first experiment, I increase the number of mass nodes in the network to determine its influence on locomotion efficiency. The second investigates how optimal compliance is related to the morphology parameters and the locomotion frequency. Finally, I discuss how optimized gait changes when driving power is constrained.

6.2.1 Loss Function

The goal is to develop a generic approach to obtain robust locomotion in open-loop without prior knowledge about body dynamics. This will be performed through an evolutionary algorithm, as presented in the first step of the two-steps learning approach (Section 3.5). In the case of simulated MSD networks, this implies the optimization of controller and morphology parameters for each specific network. This can be formulated as:

$$\hat{\boldsymbol{\theta}}^{\star} = \arg\max_{\boldsymbol{\theta}} \; f(\boldsymbol{\theta}) \; . \tag{6.1}$$

In this equation, the score function $f(\boldsymbol{\theta})$ describes the gait's performance, and the nature of this function is detailed below. The optimized parameters $\boldsymbol{\theta}$ characterize the open-loop controller and the mechanical properties of each MSD link:

$$\boldsymbol{\theta} = \{a^j, \phi^j, k^j, \omega\} \quad \text{for } j \in [1, J] \; . \tag{6.2}$$

It includes the controller amplitude a^j between 0 and 0.25, its frequency between 0 and 10 Hz, its phase ϕ^j between and 2π, and the spring stiffness k^j between 0 and 100 N/m. To synchronize the actuators together and constrain the fundamental frequency, the angular speeds ω^j are all fixed to the same scalar value ω. In the case of a MSD with $I = 20$ nodes connected to their six closest neighbors ($C = 6$), this represents a total number of springs $J = 54$ (see Equation 4.1) and therefore 163 parameters to optimize.

Locomotion characterization and evaluation are performed through two performance metrics:

- **Average speed** V: the difference between the centers of mass at the end and at the beginning of the simulation divided by the duration of the simulation which is generally 10 s. More information is provided in Section 3.6.

- **Power efficiency** P: the power dissipation of the nonlinear spring actuators can be approximated according to Chow

(2003):

$$P = \sum_j k^j \ . \ \frac{{\alpha^j}^2 \ \ {l^{\mathrm{ref},j}}^2 \ (1 + \beta^2 {\alpha^j}^2)}{4\pi} \ , \tag{6.3}$$

in which a^j are the relative amplitudes, β is the spring nonlinearity factor and $l^{\mathrm{ref},j}$ are the reference lengths of the springs.

To optimize the structure towards high speed and low power, it is helpful to use the saturated COT presented in Equation 3.32. Here, I use a slightly different form where the speed is also normalized. Using speed and power normalization will restrict the optimization to an acceptable operational range:

$$COT^{\mathrm{sat}} = \tanh\left(\frac{V}{V_{\mathrm{ref}}}\right) \ . \ \tanh\left(\frac{P_{\mathrm{ref}}}{P}\right) \ , \tag{6.4}$$

in which V_{ref} and P_{ref} are set respectively to 10 m/s and 3.6 kW. I also do not normalize this COT by the mass of the structure, as the later will be fixed to 20 kg for all structures and in all experiments as explained later. Therefore, the score can be compared across different trials.

6.2.2 Optimization

The aim is to develop an optimization approach that can be applied to highly compliant physical robots, without any need for an analytical model for body dynamics. CMA-ES, as formulated in Hansen (2006) has been selected from a pool of different optimization methods. Indeed, it fits very well for browsing non-convex parameter landscapes with a lot of local minima. In addition, it presents a good convergence speed and requires very few initialization parameters:

- The initial parameter distribution is a Gaussian centered in 0.5 and with a standard deviation of 0.2 after normalization of all parameters.
- The population size, the step size, and the covariance matrix parameter are set to their default values as recommended in Hansen (2006).

- The iteration number is set to ensure convergence, which will be assessed qualitatively by observing saturation in the score evolution.

6.2.3 Morphology Analysis

The choices made during the design of a system can contribute to more efficient and robust behaviors for solving sophisticated tasks. In the case of the MSD setup, we can intuitively assume that increasing the number of nodes will broaden the space of available trajectories, therefore increasing the number of optima at the expense of a longer learning process. It is interesting to note that such a tuning does not necessarily imply an increase of complexity, in the sense of the definition presented in Lungarella & Sporns (2006).

To verify this assumption, **I have optimized open-loop locomotion controllers for networks with an increasing number of nodes and springs**. As mentioned before, **this optimization consists of tuning the actuators' amplitudes and phases, the spring constants, and the global frequency of locomotion**. Other parameters of the MSD network are set to the same value for all bodies, except for the node masses. This is normalized by the number of nodes, such that the total mass of the MSD network (20 kg) remains the same in every simulation and the power levels required for locomotion can be compared.

To converge towards stable gaits, I add random acceleration impulse noise during the simulation. Their value is centered around ten percent of the mean absolute acceleration and is applied on random nodes five percent of the time. In the CMA-ES algorithm, the number of iterations is tuned specifically for each optimization to ensure convergence, since optimizing small structures will converge faster than larger ones. From each optimization run, the best individual is retained. Each optimization is repeated five times to average the results and obtain an estimate of the variability in the observations.

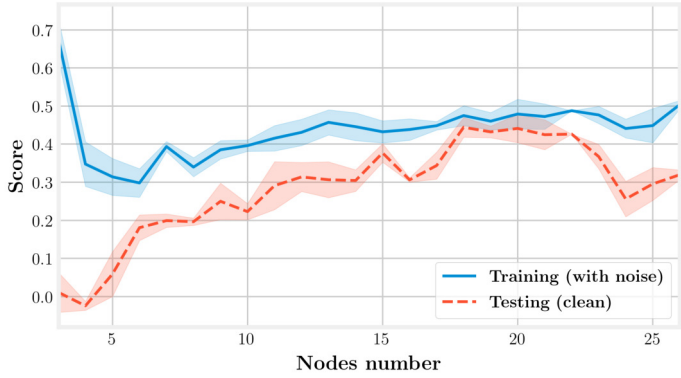


Figure 6.1: The best individuals obtained from CMA-ES optimizations of different MSD networks are plotted in blue. Testing simulations are then performed on the same individuals to identify the outliers due to noise and qualify the stability of locomotion. The low performance of 3-, 4-, and 5-nodes MSD structures indicates unstable gaits. For larger structures, the score first increases with the number of nodes but saturates rapidly for networks of more than twenty nodes.

Figure 6.1 shows in blue the evolution of the averaged best individual score during training for increasing body size. From left to right, we observe that the scores rapidly decrease for structures of up to five nodes before steadily increasing again. However, the good results in the first part of the curve should be interpreted carefully, taking their robustness to noise into account. To assess this property, I also represented the scores obtained by testing the best individuals in a new simulation without noise. We notice that the difference between the two curves decreases with the number of nodes. This shows that structures with more nodes are more robust to the noise added during the simulation. The evolution of the CMA-ES algorithm represented in Figure 6.2 also supports this hypothesis. It shows that the optima of the structures with a small number of nodes are found randomly instead of through convergence of the algorithm, unlike the structures with more nodes. High scores originate from these bodies' reduced stability. This makes them very sensitive to impulse noise as small disturbances can either make them fall over or push them forward. They can therefore rightly be regarded as outliers.

It is finally interesting to note that the score increases gradually starting from six nodes but quickly saturates. A more detailed analysis in Figure 6.3 shows that this is due to better performances in terms of speed, whereas dissipated powers are very similar. However, note that this is achieved at the expense of a longer learning process, as pointed out by the number of epochs represented on the X-axis from the graphs in Figure 6.2.

In conclusion, this experiment indicates that **increasing the number of nodes and springs in the MSD networks leads to increased robustness to external noise and better speed performances. This can provide interesting insights into real robotics when choosing the complexity of a mechanical design.** As a side note, it is interesting to have a look at the order of magnitude I am discussing here. The structures are weighting 20 kg and measure between a few meters to 20 meters. The graphs show that they are capable of locomoting from approximately 15 km/h (6 nodes) to 30 km/h (20 nodes), using between 2.5 and 4 kW, i.e. a power ratio between 125 W/kg and 200 W/kg taking gravity, friction, and air drag into consideration. These results are biased because they do not take the weight of the actuators, batteries or other technological devices into account. But the order of magnitude seem to be consistent with other vehicles of this size.

6.2.4 Frequency Range Analysis

In this second set of experiments, **I try to evaluate the nature of a link between robot compliance**, which is defined by $1/k$, the inverse of spring stiffness, **and the optimum efficiency of locomotion**.

The resonance frequency of a MSD system with one unique node and spring equals $\sqrt{k/m}$. It ranges from 0.6 Hz to 1.8 Hz for the m^i, k^j, d^j values that I am using in my setup (as a reminder, m^i is normalized by the number of nodes). There is, therefore, a bijective function between compliance and resonance frequency. It

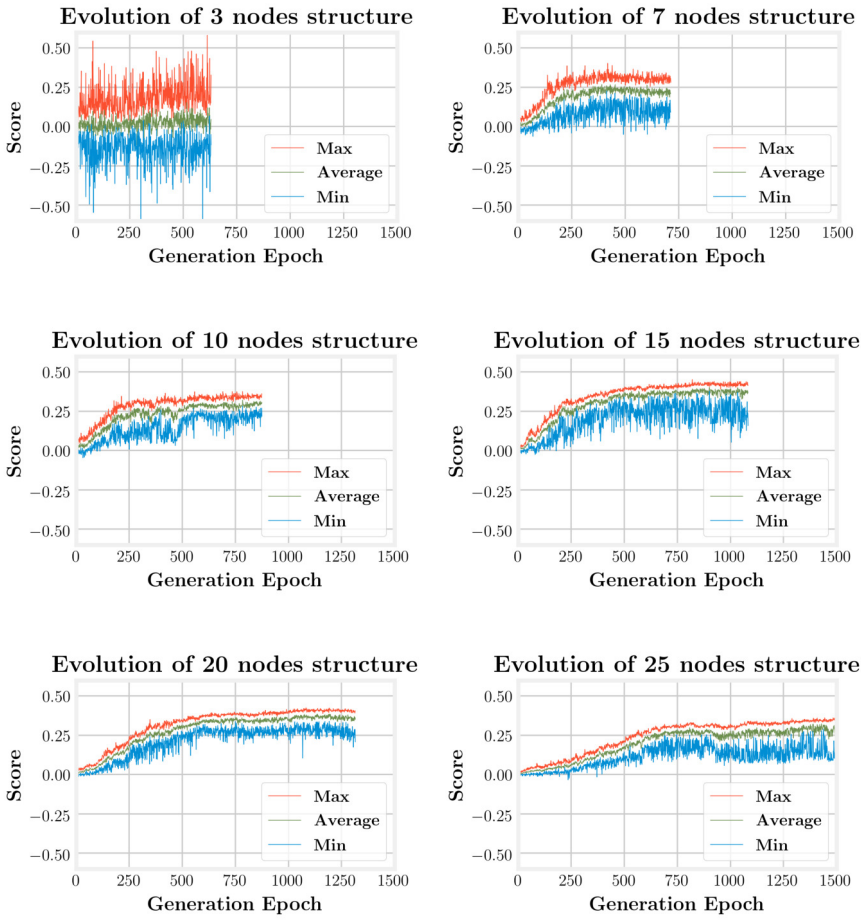
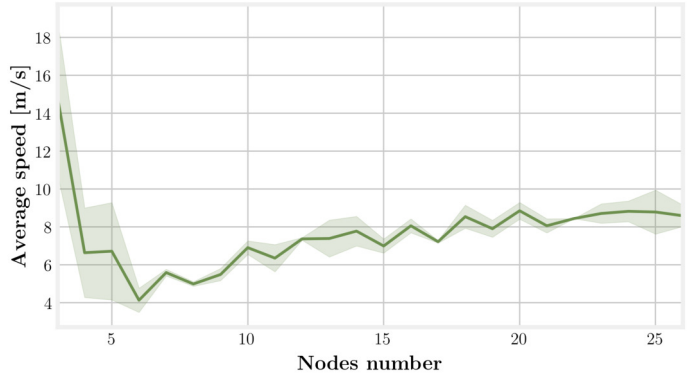
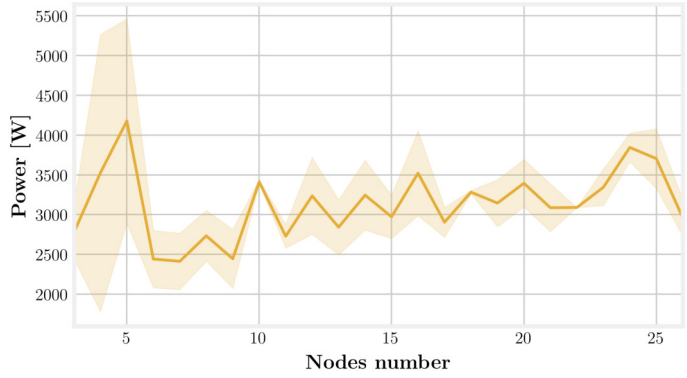


Figure 6.2: The graphs represent the CMA-ES evolution for different structures. We can qualitatively observe that the convergence time, increases with the MSD network size. This is expected as the problem becomes more complex and the number of optimized parameters is higher as well. When the structure is too simple like the three nodes one in the upper left corner, the problem cannot converge and the best results encountered during the exploration are mainly due to the random noise added in simulation.



(a) Distance



(b) Power

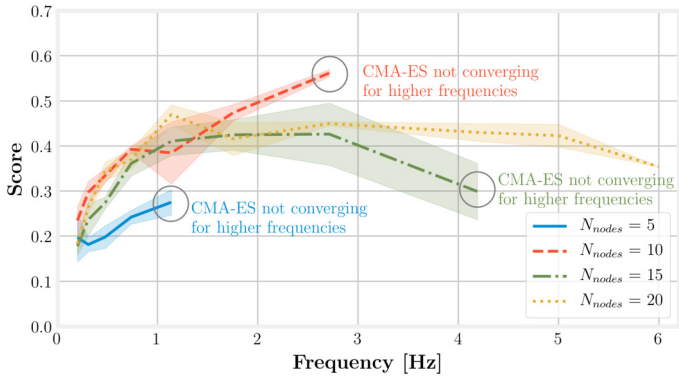
Figure 6.3: By displaying separately the distance and power components in the loss function of the CMA-ES optimization, we can acknowledge that the different performances across the structures are mainly due to the speed. As expected by the normalization factor, the driving power remains sensibly equals for each structure.

has been demonstrated that the concept of resonance disappears in non-linear systems Carbajal (2012). However, we can expect experimentally that the frequency efficiency of a MSD structure is coupled to compliance. Since they are composed of several masses and springs, we can expect that the performance peak will appear around the resonance frequency of an equivalent signal MSD node but with a larger bandwidth.

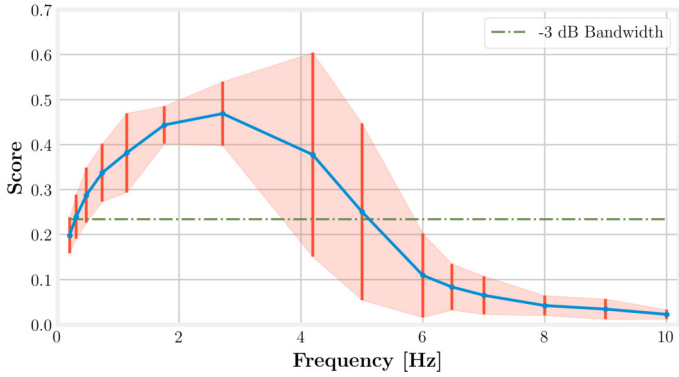
With this assumption, the study of the correlation between compliance and locomotion efficiency can be reformulated to focus on the link between actuation frequency and efficiency. Previous work like Buchli, Iida, & Ijspeert (2006) for robotic systems or McMahon & Cheng (1990) for models of mammalian gaits highlighted such a link: self-learning systems with different morphology properties tuned their actuation frequency to their body characteristics to reach optimal performance in locomotion.

In this setup, MSD structures with 5, 10, 15, and 20 nodes were optimized several times by fixing their global frequency to values between 0 and 10 Hz. In Figure 6.4 on the left, I have represented the results for different numbers of nodes. Each optimization corresponds then to a point on the graph. For some of those points, however, the optimization process was not able to converge to a gait that is both stable (whose pattern does not change in time) and robust (allowing noisy external perturbations). On the graph, this failure is particularly true for structures with a small number of nodes simulated at high frequencies. **A first empirical conclusion is then that the robustness of MSD networks at high frequencies increases with the number of nodes. This represents an additional advantage concerning the size of the system, along with the discussion from the previous section.** In terms of score, however, there is no significant difference between the topologies, and their optimal bandwidths are very similar. The optimal scores are a little lower only for the five-nodes structures, which corroborates the results from the previous experiment. To get a more accurate measure of the bandwidth, it may even be interesting to combine all the results. This is presented in Figure 6.4b, where we can observe that the structure is optimal over a 3 dB bandwidth in the range [0.3, 5.2 Hz]. The large confidence intervals around 4 and 5 Hz are again explained by the absence of convergence for the structures with a low number of nodes.

To sum up, this experiment guides the choice of compliance values in the design of a MSD network for locomotion. **Choosing global**



(a) Frequency analysis with different nodes numbers



(b) Frequency analysis averaged for all nodes

Figure 6.4: The two graphs in this figure show the evolution of score performances with the fundamental frequency of locomotion. In Figure 6.4a, I separated the result according to the number of nodes in the structure. By applying a simple CMA-ES convergence evaluation, we notice that the operating range can extend to higher frequency for larger structures. In Figure 6.4a, I combined all structures to determine a -3dB bandwidth ranging from 0.3 Hz to 5.2 Hz.

compliance to optimize a robot of a given mass is conditioned by the frequency at which we plan to actuate the robot. Hints about how to select these parameters are provided in the first approximation by an equivalent single MSD node with no non-linearities. Also, structures with more nodes tolerate a broader range of frequencies while keeping stability.

6.2.5 Performance Limits with Constrained Power

So far, I have used a loss function that combines performance for both average speed and energy consumption. However, it may be beneficial to analyze them separately to understand the limiting factors and to observe what can be the best compromise between them. The following experiment also allows me to characterize the gaits of the structures qualitatively and to observe possible transitions between different modes.

For this purpose, several optimizations have been performed by constraining the nominal power consumption of the structures and forcing their saturation to different values. In this way, one can expect to observe what is the maximum speed an individual can reach for a given power. Since I am working outside the boundaries of the desired operating range of the original cost function, I have now increased the reference value V_{ref} to 100 m/s to avoid a saturation effect due to the cost function itself.

Figure 6.5 shows the evolution of the optimal speed as a function of a constrained power budget. The best individuals are in the upper left corner. As might be expected from the conclusions of the previous section, the 3 Hz frequency gives the best results. Concerning the shape of the curve, we can see that the maximum speed increases almost linearly until 15 kW and starts saturating beyond that.

This saturation highlights the limits of the morphology. It helps to understand which factors like the spring saturation, the ground friction, the air drag, or the geometry play a larger role in performance compared to the driving power. In Section 6.2.3, I have optimized the structures for moderate speeds (results were situated between 15 and 30 km/h). In this section, I let the optimization reaching higher speeds and we have observed saturation of the efficiency in that range. For instance, the structures are capable of locomoting at 14 m/s, i.e. 50 km/h, but with a power consumption of 20 kW.

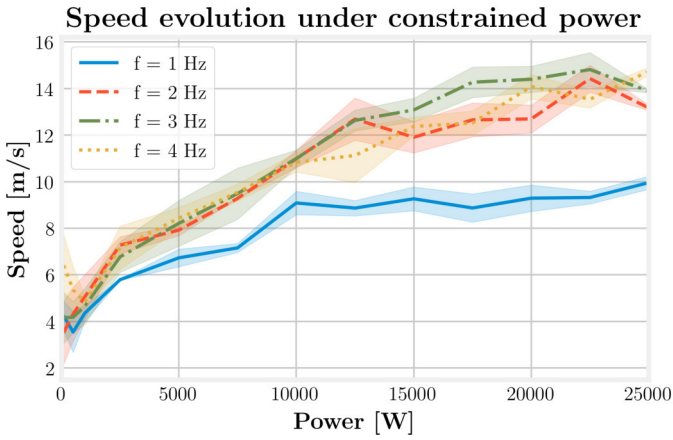


Figure 6.5: This curve shows the evolution of the maximal speed reached for different constrained power at different frequencies. The saturation effect for high powers demonstrates the physical limits of the structures while the slight decrease for very low indicates a change in locomotion gaits associated with different efficiencies.

Visual observation of the locomotion is useful to give more insights into the possible gait transitions on this curve. For this purpose, I have produced a series of video renditions of each individual simulation[1]. A qualitative analysis of those video shows that the most common gaits consist of displacing the whole structure along with a wave movement (each node touches the ground a little after the previous one) or locomoting in two steps (the body touches the ground two times per period with a phase difference of 180°). Concerning the high power saturation, a video was made for each point of the 3 Hz curve. It shows that the most energy-consuming individuals present spring extension close to their saturation which causes a loss of stability of the locomotion. In the same way, videos were produced in the low power domain for the points on the 4 Hz curve. **At low power, the GRFs follow a robust periodical pattern but the synchronization is lost at high power.** The same results have been established each of the five times the experiment has been conducted.

[1]https://youtu.be/E8Mo0UZuP0I

In short, we can stress the role of the body design in locomotion through two principal observations: firstly, a saturation of the spring leading to a degraded operation in high power; secondly, a qualitative influence of the type of gait for a fixed morphology.

6.3 Closed-Loop Experiments

6.3.1 Setup

The closed-loop system is composed of different elements represented in Figure 6.6:

- The MSD structure, that can be perceived as a physical reservoir because of its dynamics and high complexity. For each time step t_k, the system's current state is evaluated using the acceleration vectors $\ddot{\mathbf{x}}_k, \ddot{\mathbf{x}}_{k-1}, \ddot{\mathbf{x}}_{k-2}$, which comprise both X and Y components of all the nodes. The choice of acceleration is based on the work of Caluwaerts et al. (2012). Trials using integrated quantities like position or speed instead have also been evaluated but added a drifting error during training. Based on the same work, I have chosen a buffer size of three time steps. In my experiments, smaller values led to deteriorated results, but larger ones did not show any significant improvements.

- A sensor filter, whose principal role is to model the physical limitations on acceleration sensing. It is composed of an amplitude threshold followed by a low-pass filter. The cutoff frequency at 6 Hz has been chosen very low to eliminate possible oscillations due to the numerical integration method while keeping the locomotion fundamental frequency and its first-order harmonics. At the output of the filter, a vector $\mathbf{u}_k$ is sent to the next element.

- A readout layer, which computes the actuation signals for the next time step based on the current and previous states of

the MSD:

$$\mathbf{q}_k = \mathbf{W}_{\text{out}}^T \cdot \mathbf{u}_k \tag{6.5}$$

To learn the weights of the output matrix $\mathbf{W}_{out}$, I use the FORCE learning method as described in Section 3.4.2.

- A signal mixer to avoid a brutal transition from open-loop to closed-loop control. Its role is to incorporate the readout output contribution to the target signal gradually. It is defined by three parameters: the open-loop training time T_{train} when the MSD network is trained in open-loop mode only; the closing time T_{closing} in which the contribution of closed-loop signal increases linearly, and the percentage β which defines the ratio between closed-loop and open-loop at the end of T_{closing}. When this moment is reached, the system is operated in closed-loop only during a duration T_{test}.

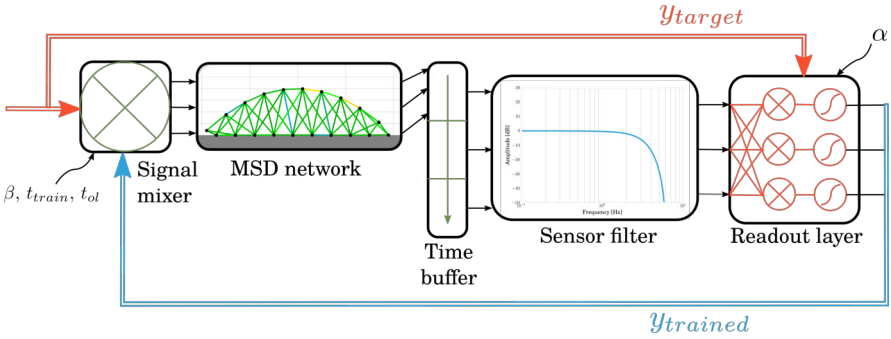


Figure 6.6: The principal components in the closed-loop learning pipeline consist in a readout layer whose weight matrix is trained at each time step and a signal mixer that gradually integrates the feedback in the actuation signal to transit from an open to a closed loop.

6.3.2 Parameter Tuning

The α parameter of the FORCE learning algorithm plays the role of a regularization variable in the process of learning the $\mathbf{W}_{\text{out}}$ matrix. It must be selected to avoid overfitting that would reduce

robustness to undesired forces on the MSD structure but also ensure a trained signal sufficiently close to the target. This is a major issue since too much noise can easily cause a divergence in the locomotion limit cycle. Tests on signal noise robustness as presented in Figure 6.7 allowed to estimate a value of $\alpha = 0.01$ as a good compromise.

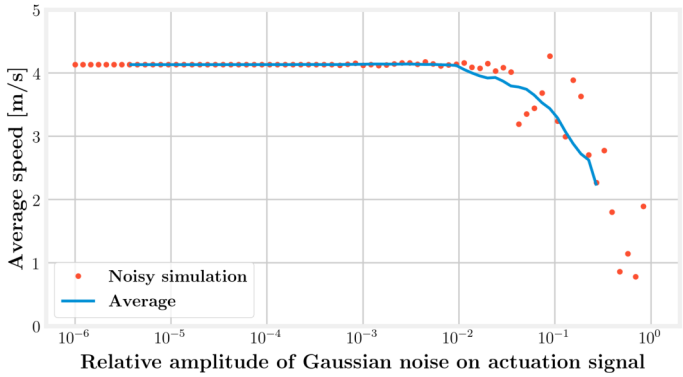


Figure 6.7: Adding some noise on the actuation signal in open-loop can give a hint about the maximum error we can accept on the trained signal in closed-loop without damaging the locomotion stability and is helpful to determine the regularization parameter. On this graph, each red point represents a simulation, and the blue line shows the average evolution. Performances start decreasing from a relative Gaussian noise of 0.01

The parameters T_{train} and T_{closing} can be estimated by analyzing the convergence error of the FORCE algorithm (see Figure 6.8) and are fixed to twelve seconds for T_{train} followed by thirty-eight seconds for T_{closing}, where the feedback signal is gradually added to the target signal to reach a value of $\beta = 95\%$ before closing the loop. Stopping the training before the actuation signal reaches 100% of feedback avoids convergence to a steady-state, as discussed in Caluwaerts et al. (2012).

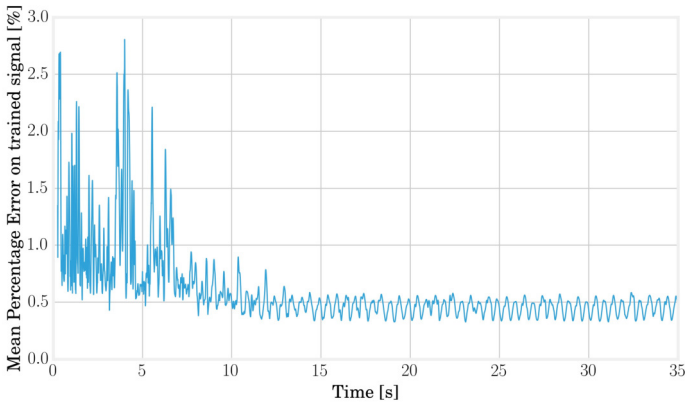


Figure 6.8: The learning error can be used to estimate the required training time and the maximal rate at which the loop should be closed. From this graph, we can deduce that twelve seconds of simulation is sufficient to consider the convergence of the readout weights.

6.3.3 Results

To determine the contribution of the system size in the process of learning its locomotion gaits, I simulated MSD networks with different numbers of nodes and evaluated the average speed during T_{test}. The same simulation was carried out in open-loop to provide a reference. The results of these simulations are presented in Figure 6.9. At first sight, it appears that the learning algorithm with its configuration can achieve performances of the same order of magnitude in open- and closed-loop for the structures between three and twenty-six nodes analyzed in this simulation. However, it is worth noting that MSD structures with less than six nodes could also not reach correct performance in open-loop.

Alternatively, the study of limit cycles indicates the stability of closed-loop control. In Figure 6.10, I represented the temporal evolution of the internal states $\mathbf{u}_k$ in a 2 coordinate space obtained by PCA. Larger structures lead to smoother limit cycles in closed-loop. The limit cycles even diverge from their basin of attraction for very small MSD networks. A simple interpretation is that more nodes

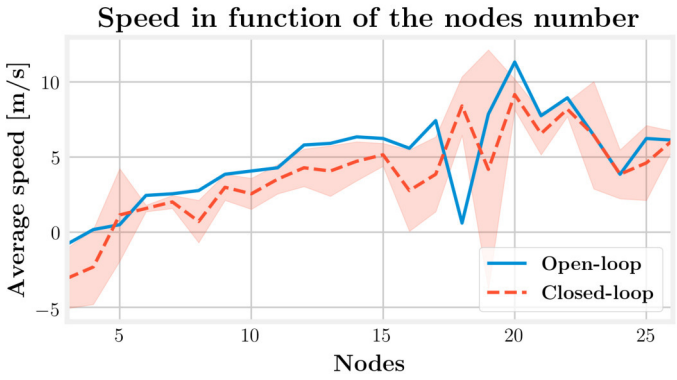


Figure 6.9: In this picture, I plot the traveled distances for the last ten seconds of simulation in open-loop in blue and closed-loop in red. There is no crucial difference between the two curves, which seems to indicate that the performances in closed-loop are similar to the ones in open-loop for all structures.

lead to more cycles in the physical reservoir, which provides more robust trajectories in the principal components reference. This hypothesis is corroborated by analyzing the quality of the generated actuation signals. This can be quantified by plotting the Normalized Root Mean Square Error, as shown in Figure 6.11, which decreases with the number of nodes.

In conclusion, the MSD morphologies are capable of computing alone their own actuation signals as a weighted sum of their sensors, without the need of intermediate computation power from a controller, and with sufficient accuracy to achieve stable locomotion. This architecture has the same level of performance as in open-loop. The computation and memory that were previously embedded in an external parametric controller can be fully distributed in the structure and the readout layer. The size and number of sensor measurements on the structure have a positive effect on the accuracy and stability of the feedback signal.

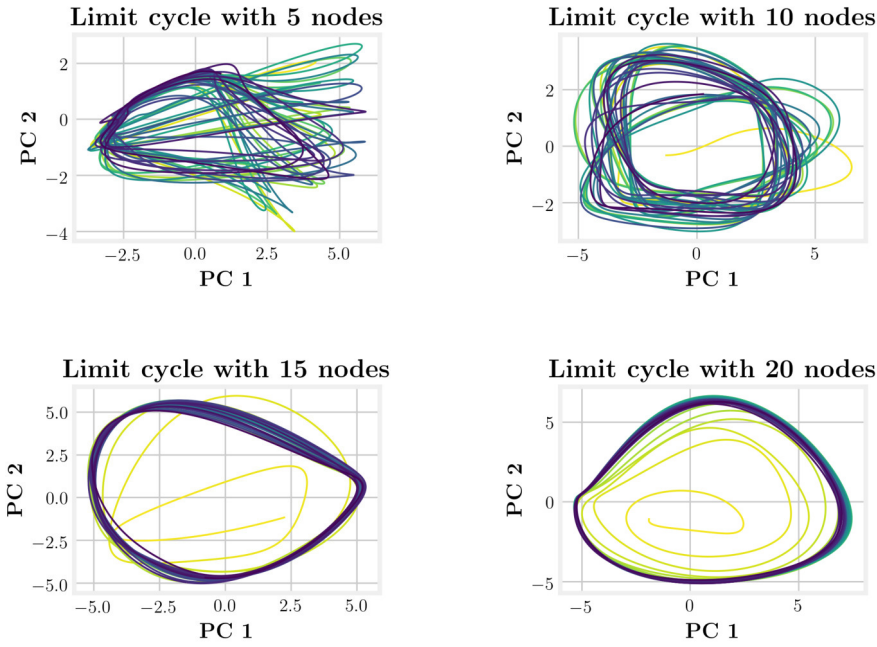


Figure 6.10: From left to right and top to bottom, the limit cycles during FORCE training are represented for structures with respectively 5, 10, 15, and 20 nodes. The color ranges from yellow for the initial seconds of the simulation (which point out the transient effect) to black at the end of the simulation. When the node number is too low, the trained signal can diverge from its basin of attraction.

6.4 Discussions

In this chapter, I have discussed the first research goal of this dissertation, i.e. the link between morphology and control in locomotion, through experiments on MSD models. These structures have been chosen for their modularity and simplicity, which allowed me to build a custom simulator to conduct the study. The total mass of the MSD networks was set to 20 kg and the size to 4 to 20 meters, depending on the number of nodes.

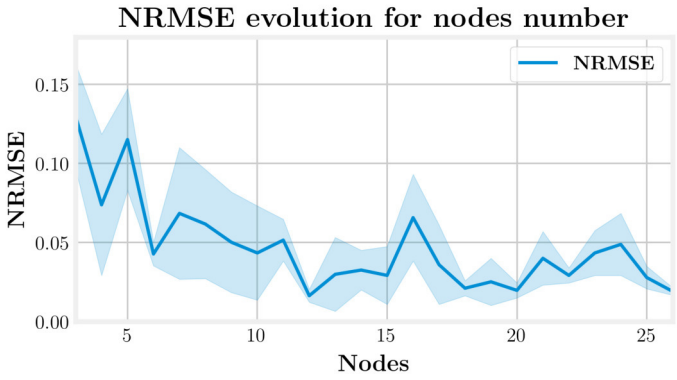


Figure 6.11: The NRMSE between the trained and target signals is represented for each node number. It indicates that the learning tends to produce better more accurate results with an increasing number of nodes.

I have applied the learning approach presented in Section 3.5 to train the structures. The first step consisted of a combined optimization (CMA-ES) of the morphology and the controller in open-loop (the phylogenetic evolution for an analogy with biology), and the second step, an online supervised learning algorithm (FORCE) to reproduce the optimal patterns in closed-loop (the ontogenetic evolution for an analogy with nature).

Different aspects have been discussed regarding the morphology and controller parameters in open-loop.

Firstly, we noted the emergence of gait patterns on the structures that look surprisingly very natural (see the video referred in the chapter). **Using very simple hypotheses** (undefined parameters of the MSD network, sinusoidal actuation of the links), **similitude with walking, trotting, and galloping gaits for different actuation frequencies could be observed**.

Secondly, Section 6.2.3 provides a discussion on the COT and how the number of nodes influences the efficiency. **It was shown**

the complex structures were more robust to noise and external disturbance than the small ones. We could not conclude about an evolution of the COT with the number of nodes. Nonetheless, the result indicated that, at a moderate speed (15 to 30 km/h), the power consumption was quite low, leading to realistic power-to-weight ratio (125 to 200 W/kg). It was further established in Section 6.2.5 that this performance saturates at high speed, pointing out the limitations of the structures in that range. **In real robotics, this seems to indicate that such tensegrity structures can be really efficient at low speed.** However, while they could present advantages on rough terrains and in modularity, the MSD structures I presented are not thought to be physically designed and conclusions should not be taken too fast. This is why Chapter 7 study the transfers of the methodology to a large hydraulic robot.

Thirdly, the relationship between the locomotion frequency, the compliance, and the efficiency was analyzed in Section 6.2.4. We have observed an optimum of performance in the range of possible frequency to actuate the structures. At low frequency, they cannot apply sufficient force to move forward, while too high frequency causes an increase of impacts with the ground which reduces the robustness. To find this optimum, **I believe that the concept of resonance of an equivalent single-node MSD (and more widely, the theory of mechanical vibrations) can provide a good first approximation, highlighting a link between optimal compliance and locomotion frequency**. However, this observation should not be over-interpreted: in real robotics and biological systems, stiffness is never set to similar values across the different joints and the role of the damping coefficient has also not been discussed. This is why this analysis is pushed further on HyQ in Chapter 7.

In closed-loop, the ability of MSD structures to generate their control signals based on a single, fully connected layer of neurons has been attested. Since the motion is fully generated by the interaction of the morphology and the controller, this architecture constitutes a good representation of an embodied sys-

tem. It can also be linked to the concept of reflex-based locomotion in biology, since no prior knowledge, clock, or recurrence is present in the controller. As an interesting observation, the size of the structure and the number of sensor signals induced a positive influence on the limit cycle's stability and the accuracy of the signals generated by the algorithm. In other words, the more complex was the structure and **the more connected were the controller and the morphology, the more robust was the closed-loop solution**. The relation between morphological parameters of MSD structure and their computational capabilities has been investigated in more depth in Yamanaka et al. (2018) since the completion of my experiments. However, applications to locomotion tasks in real robotics remain rare, and I am exploring this direction on a real quadruped robot in Chapters 7 and 8.

CHAPTER 7

—

Effects of Compliance on HyQ Locomotion

In this chapter, the end-to-end neural controller and the learning approach developed in this book are applied on HyQ, a hydraulic robot designed and hosted at IIT.

7.1 Introduction

The effect of compliance on locomotion performance has been investigated on simulated MSD structures in Chapter 6. Among others, it demonstrated a beneficial effect of compliance for locomotion using less energy. It also validated the methodology described in Section 3.5 to learn robot locomotion in closed-loop. However, the model used in these experiments contains strong assumptions and simplifications. Furthermore, such structures can hardly be produced and assembled in reality, and it is, therefore, difficult to evaluate the validity of the conclusions in the real world. In this chapter, I apply the same principle and methodology to the state-of-the-art hydraulic robot HyQ.

In general, control theory does not provide sufficient criteria to evaluate analytically how compliance acts at a body level on a real robot. As the concept of resonance disappears within nonlinear systems, it becomes hard to model optimal and stable solutions in complex systems (Carbajal, 2012) and to discuss optimal damping and stiffness values. Empirical studies on single-leg hopper robots have backed up this observation by showing that the eigenvalues of a compliant leg, i.e. analytical optima, are not fully aligned with the optimal COT in locomotion (Vu et al., 2015). A dynamical analysis seems, therefore, more appropriate for this investigation. Also, most works in compliant locomotion do not take full advantage of the dynamic interactions between the body and the controller. While morphological computation can remediate this shortcoming, the mechanical compliance property of the dynamical bodies is often arbitrarily chosen by using random spring and damping coefficients in real experiments in this field.

On top of this, the trade-off between compliance and the capacity of a physical body to facilitate its own control has not been thoroughly studied in a real locomotion task. In Lizier et al. (2011) and Hoffmann & Müller (2017), an important differentiation between 'morphological computation' on one side and 'morphology that facilitates the control' on the other has been suggested. In the

latter, the body is not literally involved in a computational process. However, its non-linear complexity can provide more dynamical landscapes where the location of attractors could facilitate the performance of a given task. To determine to which extent the morphology can perform computation or just simplifies the control, Rückert & Neumann (2013) have used optimal control strategies to discard the contribution of the controller itself in the computation process. However, the exact role of compliance in this study is only addressed briefly and qualitatively.

Therefore, **the first goal of this chapter is to clarify how compliance affects an open-loop trotting gait on the HyQ robot.** This characterization has already been carried out and discussed for separate legs alone (Boaventura, Semini, et al., 2012; Boaventura et al., 2013) and showed that some configuration of compliance cannot be theoretically achieved with an active impedance controller. Other work on HyQ has also investigated the effect of compliance (Semini et al., 2015) and the optimal impedance parameters (Heijmink et al., 2017) in trotting tasks on different surfaces. Here, I extend these results through an extensive analysis of all possible stiffness and damping parameters that can be achieved with this impedance controller in a trotting gait.

Previous work on locomotion has discussed the benefits of compliant robot's morphologies to improve energy consumption (Papadopoulos & Buehler, 2000; Vanderborght, Van Ham, et al., 2009) or locomotion speed (A. Spröwitz et al., 2013; Galloway et al., 2011). However, **to my knowledge, no paper has yet investigated the trade-offs between compliance in locomotion and the controller requirements in a closed-loop embodied approach. The second goal of this chapter is to provide such a discussion on HyQ with a perspective inspired by morphological computation.** To carry out this analysis, the robot is driven by a generic controller made of a Delay Line (DL), an Extreme Learning Machine (ELM), and a readout layer. It facilitates the search for a potential transfer of computation between body and controller and the generation of a closed-loop dynamic gait from simple biologically-inspired inputs.

The third goal of this chapter is to demonstrate how the conclusions can be successfully applied to a real robot to provide an end-to-end neural controller that can be trained in a realistic time. In comparison to some recent research that has studied machine learning control on the robot ANYmal (Hwangbo et al., 2019), the policies are learned directly on HyQ in a supervised way, which allows avoiding complex transfer procedures.

This chapter is strongly based on the following publication:

> **Urbain G.**, Barasuol V., Semini C., Dambre J., wyffels F. (2021). Effects of Compliance in Learning Embodied Locomotion on the HyQ Robot. *Autonomous Robots*. (Accepted for publication).

7.2 Methods

7.2.1 Control Architectures

Following the two-step learning approach presented in Section 3.5, I evaluate two control architectures to achieve locomotion with the HyQ robot: an open-loop architecture which is directly based on the RCF controller presented in Section 4.3.3, and a closed-loop architecture to study embodied control and its relevance in locomotion. A multiplexer enables the selection between the two architectures or allows to mix them gradually. All the developments[1] presented in this chapter have been made on top of the existing RCF controller used for dynamic locomotion on HyQ (see Figure 4.9). In Figure 7.1, I represent the modifications that have been carried to the pre-existing 'Motion Generation' component of the RCF.

7.2.2 Neural Network

The core element of the closed-loop architecture is the feed-forward neural network, detailed in Figure 7.2. At each time step t_k, sampled

[1] https://github.com/gurbain/hyq_ml

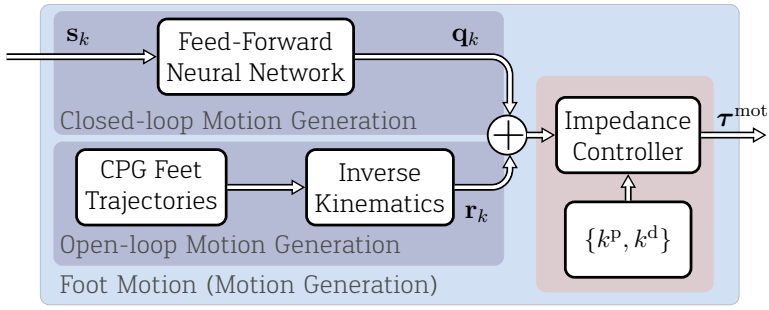


Figure 7.1: The setup allows to switch between closed- and open-loop architectures. In the open-loop architecture, a CPG determines the evolution of the foot trajectories, which are translated in the joints space using inverse kinematics. Given the absence of feedback, the dynamics of the motion generation and the robot are rather decoupled in this approach. In the closed-loop architecture, I use a feed-forward neural network that takes the GRF inputs from the robot and directly predicts the joints trajectories. The strong coupling between the robot and the controller dynamics makes it more appropriate to study the trade-offs between robot compliance and controller complexity.

at 250 Hz, the neural network model takes a set of inputs $\mathbf{s}_k$ and outputs a set of signals $\mathbf{q}_k$ that controls the flexion-extension actuators only:

$$\begin{aligned}\mathbf{q}_k \;=\; \{\; & q_k^{\text{LH}-\text{HFE}},\; q_k^{\text{RH}-\text{HFE}},\; q_k^{\text{LF}-\text{HFE}},\; q_k^{\text{RF}-\text{HFE}},\\ & q_k^{\text{LH}-\text{KFE}},\; q_k^{\text{RH}-\text{KFE}},\; q_k^{\text{LF}-\text{KFE}},\; q_k^{\text{RF}-\text{HFE}},\\ & \dot{q}_k^{\text{LH}-\text{HFE}},\; \dot{q}_k^{\text{RH}-\text{HFE}},\; \dot{q}_k^{\text{LF}-\text{HFE}},\; \dot{q}_k^{\text{RF}-\text{HFE}},\\ & \dot{q}_k^{\text{LH}-\text{KFE}},\; \dot{q}_k^{\text{RH}-\text{KFE}},\; \dot{q}_k^{\text{LF}-\text{KFE}},\; \dot{q}_k^{\text{RF}-\text{HFE}}\;\},\end{aligned} \tag{7.1}$$

where q and $\dot{q}$ represent the desired joint position and velocities; LH, RH, LF, RF are the leg index, i.e., Left Hind, Right Hind, Left Front, Right Front; KFE and HFE refer to the different actuators, i.e., Knee Flexion-Extension and Hip Flexion-Extension.

The neural controller inputs are given by the normalized GRFs and a bias:

$$\mathbf{s}_k = \{\; \mathbf{f}_k^{\text{GRF, LH}},\; \mathbf{f}_k^{\text{GRF, RH}},\; \mathbf{f}_k^{\text{GRF, LF}},\; \mathbf{f}_k^{\text{GRF, RF}},\; 1\;\}. \tag{7.2}$$

This critical choice of GRF inputs is guided simultaneously by practical reasons and biological evidence about quadruped locomotion. On one hand, using real joints' positions and velocities as feedback to teach a feed-forward neural network to predict the next desired joints' positions and velocities could ultimately split the global architecture into a set of independent controllers at a joint level. In other words, instead of training a controller to predict the next robot pose, this approach could drift to training each actuator to predict its next pose individually. In that sense, **GRFs are a better indicator of the full robot's dynamic than joints' states in an end-to-end architecture.** For instance, an error of actuation on $q^{\mathrm{LH-HFE}}$ can produce an strong direct effect on $\mathbf{f}^{\mathrm{GRF,\,RF}}$ but less importantly on $q^{\mathrm{RF-HFE}}$. This signal is also very informative about the distribution of the robot weight on its different feet (hence, its stability) and the interaction with the environment (through the impacts with the ground).

At the same time, **GRFs have a serious plausibility to represent the biological implementation of reflex-based locomotion in mammals** as detailed in Section 2.2.2. While limiting the input numbers, I am not aiming at an exact replication of the RCF dynamics. However, this representation should be able to learn a stable attractor with trajectories identical to the ones in the RCF, using no prior knowledge about the robot morphology and mechanics.

In the generic controller, I opted for a very abstract representation: normalized inputs are sent to a Delay Line (DL), which acts as a first-in, first-out memory buffer. It is fully connected to a hidden layer of hyperbolic tangent neurons, followed by another fully connected readout layer of linear neurons. The layers' architecture is inspired by the Extreme Learning Machines (ELM) (Huang et al., 2004). This approach has been used in robotics before (Degrave et al., 2013) and has the benefit of exposing only two parameters M and N to tune respectively the memory and the nonlinear hidden projections of the controller's model. **The absence of prior knowledge in the network structure does not restrict the range of motion to a defined subset but allows an infinite set of**

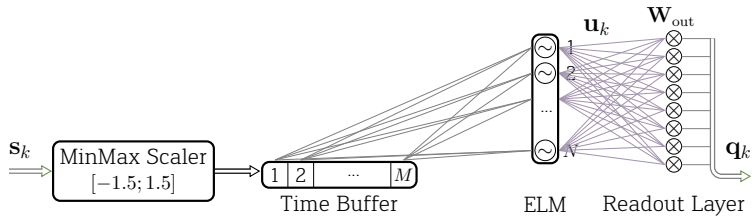


Figure 7.2: Tuning the memory and the non-linearities of the neural network can be performed by setting respectively the size of a time buffer and a hidden layer of hyperbolic tangent neurons. The time buffer is fully connected to the hidden layer and only the connections between the hidden layer and the linear readout are trained.

continuous trajectories. Furthermore, due to the feed-forward processing, no temporal components are added in the controller dynamics.

7.2.3 Training

The weights of the connections between the hidden and readout layer $\mathbf{W}_{\text{out}}$ are updated using the FORCE learning method (Sussillo & Abbott, 2009) (presented in Section 3.4.2, Equations 3.14 to 3.17) to learn to reproduce the trajectories provided by RCF target, $\mathbf{r}_k$.

As presented in Figure 7.3, the trials are divided into three phases of 15 seconds each. During the first phase, T_{train}, the robot is actuated by the RCF only and the signals are used as a target to train the readout layer of the neural network. In the second phase, T_{closing}, we switch to the neural network output. The RCF target is however still used as a target to train the network. The closing phase alternates between prediction and training until only the prediction remains at the end of the phase. Finally, in the last phase T_{test}, only the predictions of the neural network are used to investigate its performance.

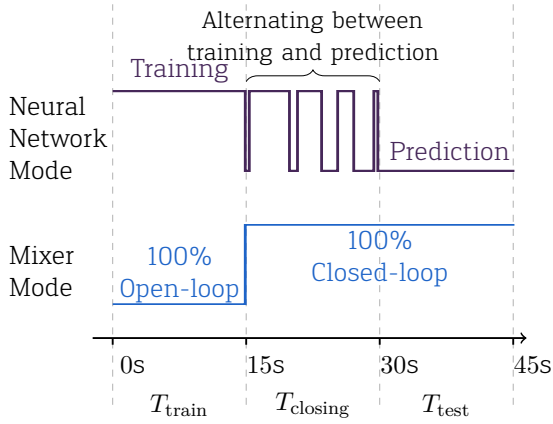


Figure 7.3: The simulations and trials on the real robot are divided into three phases of 15 s. In the first phase, the robot is actuated in open-loop while the neural network is in training mode. In the second phase, we switch to the closed-loop architecture, and the network alternates between training and prediction modes. It stays only in prediction mode in the third phase.

7.2.4 Experimental Methodology

The experiments realized to collect the results are divided into three parts.

First, in Section 7.3.1, **I am using the open-loop control architecture in simulation to evaluate different virtual stiffness and damping coefficients. To estimate the performance in open-loop**, the experiments run for 15 seconds and only the first phase T_{train} is carried. Performance indicators on stability, power, speed, and GRFs are measured to determine the behavior of the gaits and identify different locomotion regimes.

In the second set of experiments, presented in Section 7.3.2, **I evaluate the effect of compliance on the COT and the minimal controller requirements in a closed-loop architecture**. In the stiffness-damping region with the best open-loop performance identified in the previous step, I have trained the neural network

of the closed-loop architecture over the full duration of 45 seconds as discussed in Section 7.2.3. The interaction of four quantities (controller memory, controller non-linearity, joint stiffness, and joint damping) is evaluated for the dynamical system formed by the robot and the controller.

The third part, presented in Section 7.3.3, **details the experimental trials on the real HyQ robot** and discusses the qualitative observations made during these trials.

7.3 Results

7.3.1 Compliance Regions for the Open-Loop Architecture

This section provides a characterization of the different trotting regimes on HyQ in function of its compliance. To this goal, 3.600 different settings in the HyQ's active impedance controller are sampled on a logarithmic space formed by the joints' damping and stiffness. Ten trotting simulations with the open-loop architecture are averaged for each impedance, using the RCF as detailed in Section 4.3.3. Four metrics are represented as they together summarize the locomotion behavior: stability S, power P, speed V, and $GRF^{\max}$. Motivations and an explicit definition for each of the metrics are provided in Section 3.6.

From the results presented in Figure 7.4, we can cluster the different morphologies following their gait performance metrics. The graph in Figure 7.5 provides such a partitioning into six different regions in the stiffness-damping space. To understand how these regimes are determined, the effects on the final gait are first discussed for increasing stiffness (X-axis). At low stiffness, the robot is falling and the GRFs observed in the simulation become unrealistic. This is caused by the difficulty of the physics engine to cope with the diverse impacts due to the undesired robot position, lying on the ground. In Region 1, the power consumption remains small and the robot speed is reduced to zero. Increasing stiffness raises the frequency of the joint oscillations. In Region 2, these oscillations

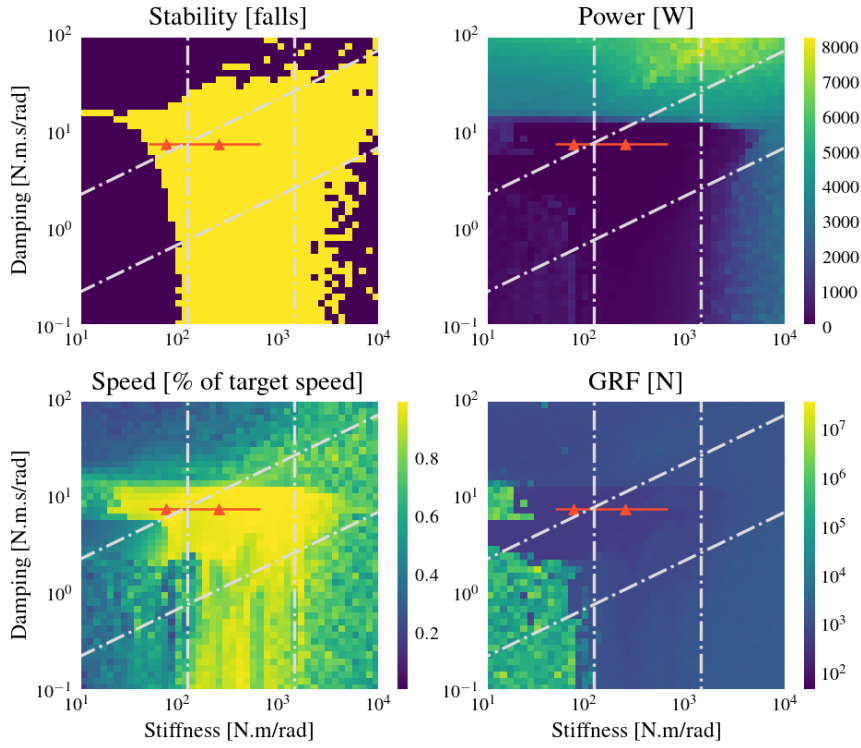


Figure 7.4: Gait metrics for different damping and stiffness values. Grid simulations represent the robot locomotion behavior in terms of stability (falling in blue and not falling in yellow), mechanical work power (in Watts), normalized speed (as a ratio of the desired target speed), and GRFs (in Newtons). Some unrealistic simulation results are discussed in the manuscript. A range of good performance on all four criteria is defined by the red line on the graphs. From left to right, the red triangles define two sets of impedance, called 'compliant' and 'stiff', selected to evaluate a generic controller.

have the same order of magnitude as the locomotion frequency. The stability module presented in Section 4.3.3 is, therefore, able to prevent the robot from falling. This region is characterized by a correct forward speed and a stable yet shaky gait. When shifting to higher stiffness in Region 3, the oscillations are too fast and lead to locomotion instabilities, which explains the high falling rate and the low average speed.

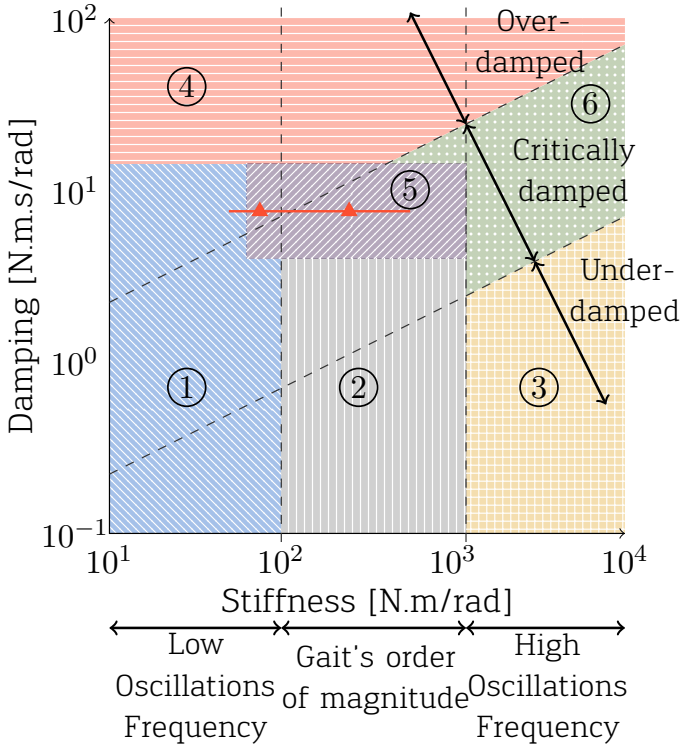


Figure 7.5: A description of locomotion regions. The damping-stiffness space is divided into characteristic regions describing the robot locomotion.

The role of the damping coefficient can be better understood by browsing the graph diagonally through the different curves of iso-damping ratios. For lower ratios (Region 3 for instance), the oscillations around the desired joint positions are under-damped. The robot has more undesired impacts on the ground, causing it to fall. The power consumption is also increased due to the high velocities of the joint oscillations. On the other end, in the upper left corner of the graph, the damping ratios are too high and the actuators are over-damped. In Region 4, the dissipated power measured in simulation becomes unrealistic due to the small amplitude of the oscillations. Between these extremes, critical damping can be reached in Regions 5 and 6. Despite the high frequencies, the robot can maintain a positive forward speed

without falling in most cases. **Region 5 constitutes the most interesting area for further studies as it combines acceptable damping ratios with correct stiffnesses. It is characterized by adequate speed and energetic performance and realistic GRFs for a transfer on the real platform.** A red line is defined in this area of the graph. It will be considered in the next section to study the compliance effect in closed-loop locomotion. Two parameter sets, called 'compliant' (left) and 'stiff' (right) are also identified with a red triangle on this line. They score equally for all the metrics in open-loop but have different stiffness values.

7.3.2 Effect of Compliance in Closed-Loop Control

The empirical evaluation of compliance in the last section constitutes a basis to discuss major assumptions in morphological control. This section aim at quantifying the effect of compliance in a closed-loop architecture by providing an answer to the two following questions:

1. **Can compliance improve the COT in reflex-based locomotion?** On one hand, it seems that compliant robots can be safer and use less energy (Papadopoulos & Buehler, 2000; Vanderborght, Van Ham, et al., 2009; Kashiri et al., 2018; Seok et al., 2013). On the other hand, some work has discussed the presence of an optimal stiffness value to maximize the locomotion speed of a robot (A. Spröwitz et al., 2013; Galloway et al., 2011). Here, I evaluate how these two effects contribute to the COT in a closed-loop control architecture.

2. **Can compliance facilitate morphological locomotion control on complex dynamic structures?** While non-linearities and fading memory have been identified as essential features to perform computation with generic morphologies in Hauser et al. (2011), the role of compliance and its interaction with the controller complexity requires further investigation. Here, I contribute to this topic with empirical demonstrations on HyQ.

7.3.2.1 Cost of Transport

In this first investigation, I use the closed-loop architecture and a fixed pre-processing pipeline is selected by setting the memory and non-linearity parameters M and N to 50. These values are selected to provide enough complexity for the task with a correct real-time execution, as shown later in Figure 7.7. Robots with different compliance are sampled from the red line introduced in Figure 7.5. For each of them, I average ten simulations in which the neural network is trained as described in Section 7.2.3. The results for different metrics are displayed in Figure 7.6.

In the top-left graph of Figure 7.6, the evolution of the COT is represented for different compliance values. The red line describes the closed-loop architecture under study and the blue line is given as a benchmark by using the RCF controller in open-loop. The first observation is that the COT of the closed-loop system increases almost exponentially with stiffness. **In other words, good performance with the end-to-end neural control architecture can only be achieved with compliant robots.** In the scope of HyQ's locomotion task, this is a serious motivation to use compliant morphologies in the search for optimal performance.

The COT is defined by the ratio between the robot's power consumption and speed, normalized by the robot's mass (see Section 3.6). The speed and power curves are, therefore, also represented in Figure 7.6 to better understand the reasons behind the increase of COT with stiffness. The power value is only theoretical as it is obtained by multiplication of all actuators torque and speed but calorific losses due to hydraulic compression and other mechanical drags need to be added in practice. The variations of COT seem to be mostly explained by the power contribution. To understand why, we rather need to consider the dynamic of the system as a whole. The last line of Figure 7.6 shows an increment of the actuators' oscillations around their limit cycles at higher stiffness. Despite these oscillations, we can note that the neural network accuracy presented in the same figure, remains quite stable with the stiffness, i.e. it can keep track of the target trajectories. So,

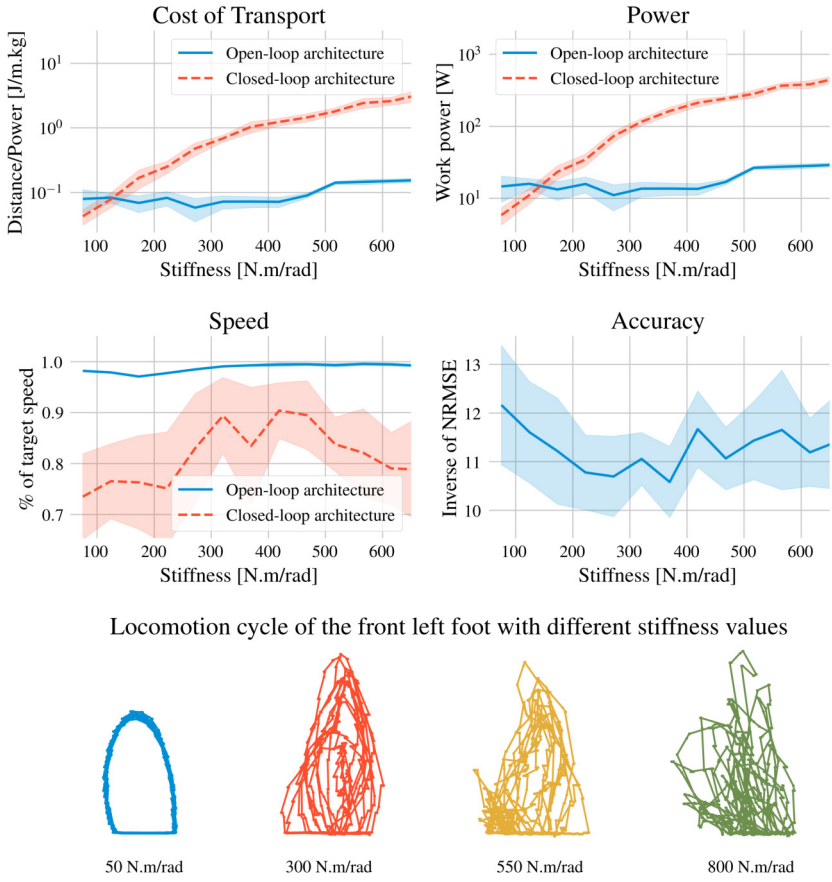


Figure 7.6: The COT for the closed-loop architecture increases with stiffness (top left). This effect can be linked to an increment in power consumption, itself due to higher impacts on the ground. The speed, however, slightly increases with the stiffness before to drop at higher values, because of reduced stability and gait efficiency.

the oscillations are not due to the controller or the robot alone but rather to their interaction as a dynamic system. In turn, this is responsible for larger impacts between the feet and the ground, as substantiated by additional graphs presented in the related publication referenced in the chapter's introduction. **The increase of power consumption with the stiffness also corroborates the explanations presented in other works** (Vanderborght, Van Ham,

et al., 2009; Hoffmann & Simanek, 2017; Kashiri et al., 2018) **and advocates for the use of compliance to optimize energy efficiency and reduce the risks of impacts**, particularly in situations where humans are involved.

In comparison with power, speed seems to play a smaller role in its contribution to the COT variations for the closed-loop architecture. The dominant trend shows that speed is first rising with the stiffness before dropping when the joints become too stiff. The positive effect relates to a better lift-off phase and a stronger push on the ground to propel the robot forward when increasing the joint's rigidity. However, the negative effect starts to take over at stiffness higher than 400 N.m/rad. Additional investigation (see related publication) shows that this is principally due to a reduced average step length, i. e. the feet are touching the ground sooner than expected. A visual analysis of the locomotive cycle in the last line of Figure 7.6, and the trajectories displayed in Figure 7.8 corroborate this effect.

A last interesting observation from the COT graph suggests that a closed-loop controller becomes better than an open-loop one against this metric for really compliant robots (far left side of the curve). **This highlights a positive point of morphological control: the ability to better handle compliant bodies, without forcing the joints to follow a given trajectory** or computing complex intermediate representations with finite elements theory. Nonetheless, it is important to stress that this benefit also comes at a certain cost, as featured by the reduced speed at low stiffness.

7.3.2.2 Computational Requirements

In the second analysis, I evaluate the trade-offs between the controller's complexity and the robot compliance, to contribute to stable and efficient locomotion. To this goal, a qualitative evaluation of the gait is conducted for the two individuals with 'compliant' and 'stiff' parameters described in Section 7.3.1. For each of them, several controllers with a different amount of non-linearity, N and memory, M, are trained. This choice is inspired by the fact that

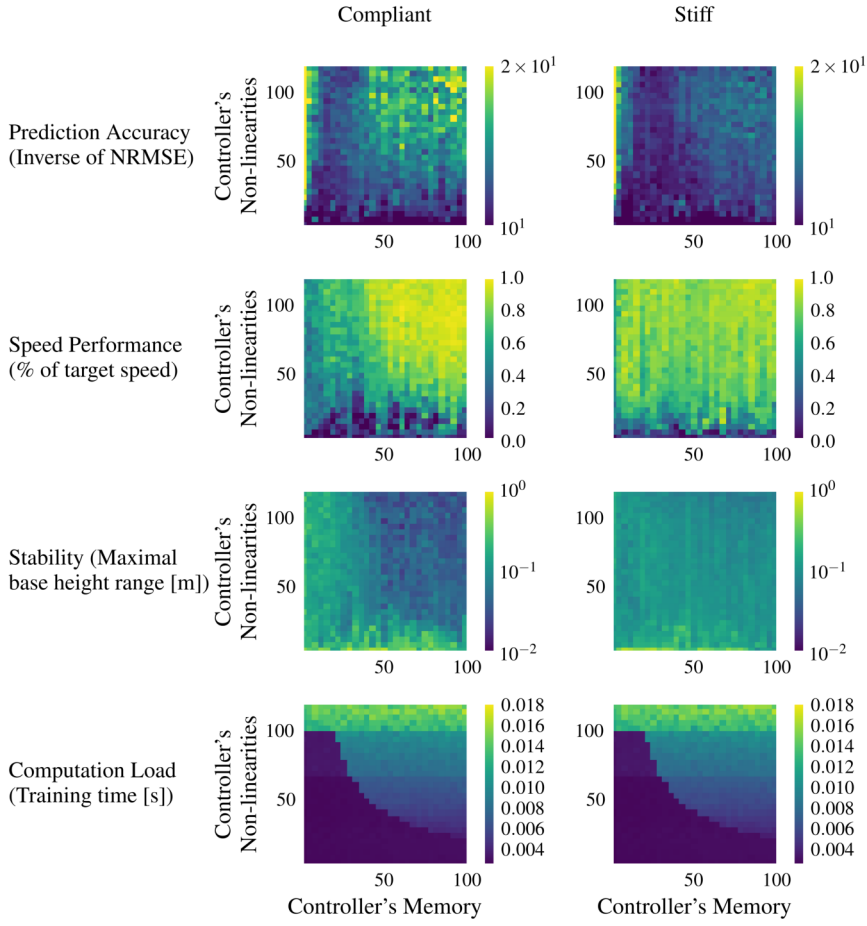


Figure 7.7: In each of these six graphs, various controllers are evaluated by tuning the delay line size M (to increase or decrease the controller's memory) on the X-axis and the ELM layer size N (to change the non-linear richness) on the Y-axis. The columns correspond to the 'compliant' and the 'stiff' joint parameters. Each row represents a different locomotion metric ranging from blue for low values to yellow for high values. The spreading of the region of efficient locomotion with the stiffer robot can be interpreted as a decoupling between the controller and robot dynamics at higher stiffness.

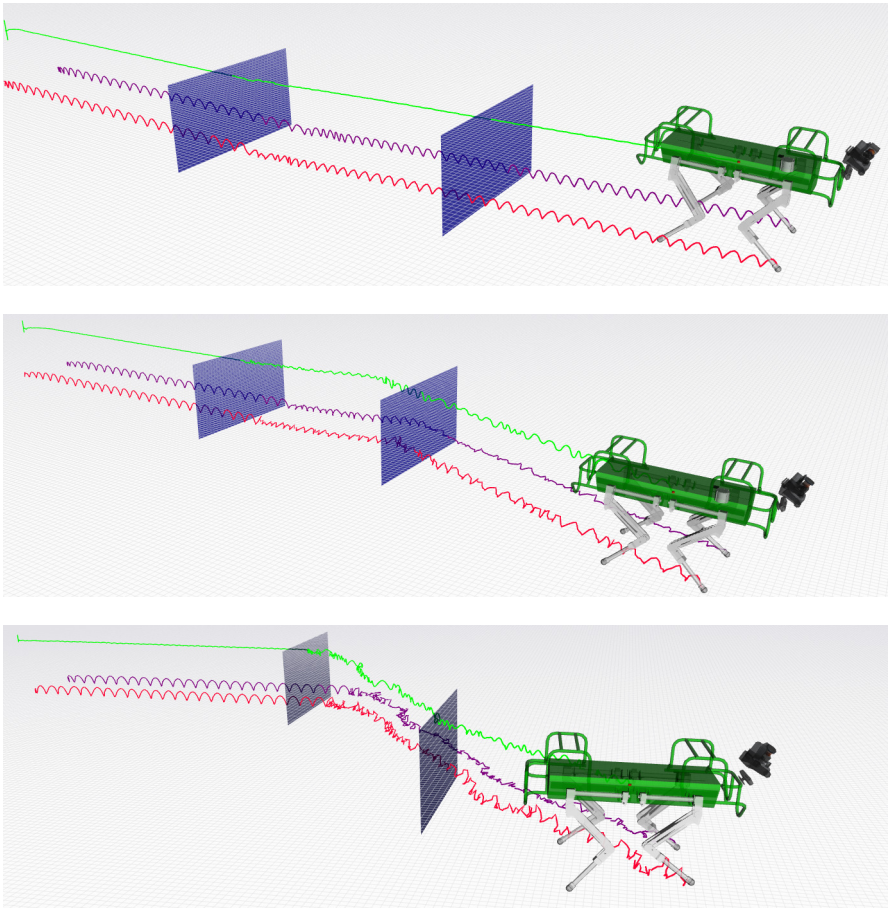


Figure 7.8: This figure shows the end-effector trajectories of the right front foot (red), the left front foot (purple), and also the trunk's center of mass (green). The blue planes divide the trajectory in the time domain into three phases:'Training', 'Closing' and 'Testing'. The experiment is repeated for the two compliance values discussed in this chapter: 'compliant' (top), and 'stiff' (bottom). It shows a deteriorated locomotion cycle at higher stiffness.

nonlinear dynamical systems with fading memory have a finite computational capacity and display a trade-off between memory and non-linearity (Dambre et al., 2012). Furthermore, memory and non-linearities are not only meaningful in the controller but also at the body level. Therefore, I believe that these parameters are relevant to measure the complexity of the dynamical system formed by the robot and its controller and to discuss the influence on the final controllability of the system. For each neural architecture, ten simulations are performed to provide a meaningful estimate and four metrics are investigated in Figure 7.7: accuracy, speed, stability, and training time. In this figure, the left column represents the 'compliant' robot, and the right column, the 'stiff' robot. In each graph, the X-axis describes the evolution of the controller's memory and the Y-axis, the controller's non-linearities.

A qualitative evaluation of the gaits obtained by the 'compliant' and 'stiff' robots is also presented in Figure 7.8 to visualize how the gait trajectories predicted by the neural network are affected by the compliance. To draw these pictures, I have selected a controller with $M = 50$ and $N = 50$.

We can first observe in Figure 7.7 that the prediction accuracy (see definition in Section 3.6) increases with the controller memory and non-linearities (i.e., in the direction of the upper right part of the graph). As can be seen in the second row of this figure, this improvement in prediction accuracy is also linked to a higher speed. It also has a beneficial impact on gait behavior, as it seems to correlate with better stability in the third row. However, this effect is mostly observed for soft impedances and disappears for stiffer ones. This is expressed by the spreading of the accuracy, speed, and stability levels for the 'stiff' robot on the graphs.

Further investigations of the different robot parameters have been conducted to validate the understanding of the robot behavior, although they are not presented in this book. They conclude that stability is correlated with the maximal impacts on the ground, which are destabilizing the robot. They also show that the power

consumption of the robot is mainly driven by the compliance value while the controller memory and non-linearities have only little impact.

An important remark needs to be formulated here. In the hypothesis of a strict transfer of computation to the robot facilitated by compliance, we could have expected the migration of the region of interest (yellow points for accuracy and speed, and blue points for stability) on a diagonal axis between the 'compliant' and 'stiff' morphologies in Figure 7.7. However, we failed to observe such an effect in the experiments. Instead, these regions start spreading on the graphs. **In other words, this means that the controller complexity, represented by its memory and non-linearities on the X and Y axis, has less influence on the locomotion metrics for a stiff robot than for a compliant one.** The robot becomes less sensitive to the computation power of the controller at higher stiffness. Therefore, these empirical results rather support the idea that **increasing stiffness leads to more decoupling between the controller and the robot dynamics in closed-loop, making morphological control a harder task with stiff robots**, rather than demonstrating an explicit transfer of computation between the controller and the body.

In the scope of morphological control, it can be concluded that **compliance is facilitating the control of locomotion on the HyQ robot. The feedback from a compliant body backs up the controller's memory and non-linearities to produce a better prediction in a virtuous cycle.** This observation is fundamental in the scope of embodiment and morphological computing as it emphasizes how the choice of morphology is important to obtain a stable limit cycle.

In addition to these metrics, it is useful to estimate the controller complexity with an analysis of the computer load induced by the different neural architectures. In the last row of Figure 7.7, I represented the average training time step. This value only depends on the architecture and the parameters M and N but it has no direct

relation with the robot morphology. There are three clear areas on the graphs, certainly corresponding to different implementation or optimization in the underlying algebra libraries. However, no specific focus was given on the code implementation, and better values could be expected with a different implementation. On the computer used for the simulation, only the robot with the 'compliant' parameters could be trained to reach a good accuracy at a minimum of 250 Hz, which corresponds to the default frequency used on the real robot. This was taken into account when selecting a neural architecture in the experimental phase with the real robot. Nonetheless, in the testing phase, it is easy to quantify the number of operations in the feed-forward neural network. The number of weighted sums computed by this network is equal to:

$$M \,.\, N \,.\, 5 \;+\; N \,.\, 16, \tag{7.3}$$

where M and N are respectively the sizes of the delay line and the hidden layer of the controller, as explained in the main manuscript. The number 5 corresponds to the dimension of the input vector and 16, the dimension of the output vector.

7.3.3 Evaluation on the Real HyQ Robot

Despite recent progress on locomotion using machine learning, most works are conducted in simulation, and only a few results have been tested on real robots. This issue primarily arises from what is called the 'reality gap': simulation models cannot reproduce the real physics accurately enough without a prohibitive amount of computation or excessive parameter tuning. In this chapter, however, I have ported the controller implementation to the real HyQ robot. Minor parameter and algorithm adaptations have been added in comparison with the previous section to deal with the 'reality gap':

1. To cope with differences in weight, inertia, or actuator torques between the robot and its model, the stiffness values were raised to 250 N.m/rad and 150 N.m/rad in the swing phase and 150 N.m/rad and 100 N.m/rad in the stance phase respectively

for the hip and the knee. This slightly differs from the value of stiffness in the 'compliant' morphology.

2. The GRFs are indirectly acquired using encoders and load cells on different joints. The sensor accuracy at high torques and the propagation of errors in computing the contacts lead to noisy sensor data in comparison to the simulation baselines. This effect is corrected by applying a moving window filter using 15 samples followed by a threshold function on the GRF data. The values before and after processing are presented in Figure 7.9. The simulation counterpart is also represented on the same figure and demonstrates that the process can provide correct amplitude, smooth transitions, and keep the important phase information (gait synchronism).

3. The robot simulation model assumes a uniform distribution of the mass along the trunk. In reality, some subsystems located in the body are heavier than others, which creates a heterogeneous weight repartition. Therefore, the robot is less stable than its simulated counterpart and covers a larger range of roll, pitch, and height during locomotion. This effect is yet increased by external forces applied on the hoses that provide pressurized oil to the robot and that randomly pull or push it to different directions during the gait. Finally, another disturbance comes from the communication delay between the robot and the computer where the neural network is running. For all these reasons, the experiment duration was increased to 180 seconds with 120 seconds of training, 30 seconds of closing phase, and 30 seconds of testing phase. The learning algorithm regularization was also raised from 0.001 to 1 to generalize on more samples.

The experimental validation results are presented in an additional video[2]. The transfer can be considered successful as there is no qualitative difference between the trot using the target gait of the classical controller or the neural network gait. The stability of the trunk is contained in the same value range and the feet trajectories

[2]https://youtu.be/J5Ann6cPEGs

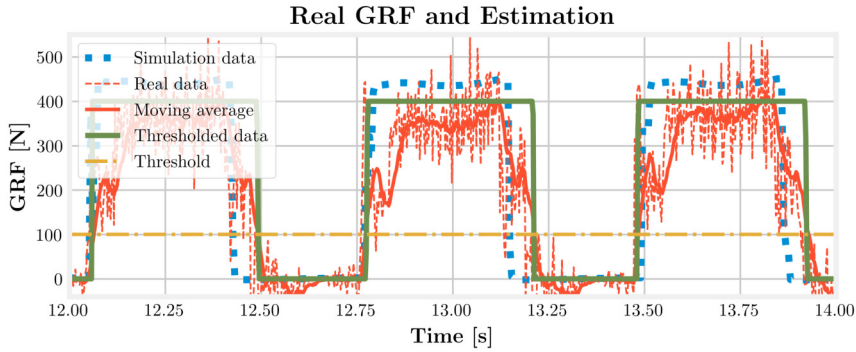


Figure 7.9: In this graph, the GRFs are presented with the raw signal (red), after processing using a window filter and threshold (green) as well as a comparison with the signal in simulation (blue). The preprocessing method keeps the phase of the real signal but it generates levels that are closer to their simulation counterpart.

show no visible deviation from the targets. However, it has been observed that the robot is more sensitive to external disturbance. This issue is discussed in depth in Chapter 8. Aside from this, **this section provides proof about the capacity of a neural controller to operate a large quadruped robot directly at a low-level joint level, to sustain an efficient embodied locomotion.**

7.4 Discussion

The experiments presented in this chapter advocate in favor of using compliant robots in conjunction with morphological control. They highlight the role of compliance for learning control and coordination of locomotion movement and the relevance of this approach on real robots.

Besides providing advantages on the COT, simulations on HyQ show that compliant robots can simplify closed-loop control of robot locomotion. A possible interpretation of the role of compliance can be found in morphological computation theory. A robot can be seen as a computational entity that transforms actua-

tion signals into sensor signals through its dynamical interactions with the environment. It is nevertheless essential to differentiate 'morphologies that facilitate control', in which a good design reduces computation requirements, from 'intrinsic morphological computation', where there is clear outsourcing of the computing power. This empirical study demonstrates that intrinsic morphological computation could be discarded in the locomotion task under consideration. At higher stiffness, however, we observe a decoupling between the control and the robot dynamics but this decoupling reduces when increasing compliance. From the empirical conclusions in this chapter and the previous one, it looks like this coupling helps to increase the robustness of the closed-loop system. Compliant robots have, therefore, 'morphologies that facilitate control'. Rather than a concrete transfer of computation to the body as discussed in other work on embodied locomotion (Caluwaerts et al., 2012), compliance of HyQ helps to regulate the locomotion trajectories and better outsource the body feedback.

By quantifying the needs in memory and non-linearities of a neural network and interpreting them in the light of the mechanical properties of a robot, **this chapter aims at bridging the gap between the fields of robotics and machine learning. In particular, the results could be used in further investigation of reinforcement learning methods** to estimate the requirement of a recurrent network controller. This work also enriches the efforts in reflex-based locomotion with an effective demonstration on a real robot, which can bring some insights into the fundamental understanding of the neural mechanism used for locomotion in biology.

Built on the simulation work, a successful validation on real hardware was investigated. The experiment also showed the possibility to operate locomotion only in a reflex-based fashion. However, the loss of phase synchronism with the motor target emphasizes the importance of oscillatory systems, like biological CPGs, or predictive systems, like a biological Cerebellum, to keep a stable locomotion phase in presence of external disturbance. This hypothesis is discussed in more detail in the next chapter. Furthermore, the use of a

stability module for PID-based attitude control and gravity compensation (called 'Trunk Controller'), in association with learning feet trajectories and coordination, was crucial to facilitate the transfer. I believe that this is a key to bring simulation results on locomotion robots without damaging them and further experiments regarding this module are also provided in Chapter 8.

CHAPTER 8

—

Biologically Inspired Stability Control on HyQ

Body stability is an essential ingredient to avoid damages and accidents when learning locomotion. This proposition is further discussed in this chapter.

8.1 Introduction

Despite a global endorsement of the CPG model in locomotion, physiological and functional experiments in mammals have also indicated the presence of descending signals from the cerebellum, and reflex feedback from the lower limb sensory cells, that closely interact with the centralized control. To this day, these interactions are not fully understood. In Chapters 6 and 7, I have discussed the application of the learning method described in Section 3.5 on MSD structures and the HyQ robot. In both cases, this has led to a robust reflex-based architecture for locomotion but HyQ was also equipped with a stability control module to carry the demonstration. In this chapter, I investigate the role and the constitution of this stability control module during treadmill experiments with HyQ. **The main hypothesis introduced here is that the stability mechanism of HyQ can benefit from a temporal representation of the gait. Despite the strong bias induced by the robotic hardware and software, I believe that the biological inspiration of the experimental set-up can contribute to a better understanding of the fundamental mechanisms in mammal locomotion**, and partially answer the third research question of this thesis, addressed in Chapter 1.

As discussed in Section 2.2.2, it has been shown that the human cerebellum influences the control of extensor muscles to maintain correct balance and a proper stance, and to modulate the rhythm of locomotion patterns in Morton & Bastian (2004). Experiments with patients affected by cerebellar ataxia demonstrated a decreased stability of the trunk's center of mass due to a deteriorated stance in the presence of lateral and backward disturbance during locomotion compared to healthy subjects (Bakker et al., 2006). Other investigations demonstrated that cerebellar impairment did not decrease reactive feedback-driven adjustments, but significantly damaged predictive feedforward motor adaptations (Morton & Bastian, 2006). **This evidence supports the hypothesis that the cerebellum helps during locomotion in predicting limb movements using a stored internal representation with spatial and tempo-**

ral components. To assess this hypothesis, I compare two models: one relying on a predictive pattern of desired stance (called PSE) and a second, less biologically plausible, relying on feet position above the ground (called RSE). I demonstrate that only the first model leads to robust locomotion on HyQ, which supports the idea that functional inspiration from the cerebellum can have a positive impact on robotics locomotion.

This chapter is strongly based on the following publication:

> **Urbain G.**, Barasuol V., Semini C., Dambre J., and wyffels F. (2020). Stance Control Inspired by Cerebellum Stabilizes Reflex-Based Locomotion on HyQ Robot. *Proceedings of the International Conference of Robotics and Automation, ICRA 2020*.

8.2 Methods

The overall control architecture used in the experiments is presented in Figure 8.1. It is based on the closed-loop architecture introduced in the previous chapter, but the posture control can be now parameterized. It is divided into three parts: the real robot in its environment, the reflex-based motion controller, and the posture controller inspired by the cerebellum. The HyQ robot and the RCF controller, which serves as a target in the SL procedure, have been presented in Section 4.3. The other important components of the experimental architecture are further detailed hereafter.

8.2.1 Neural Network and Training

The architecture of the reflex-based neural network has been presented in the previous chapter, in Section 7.2.2. It receives a signal $\mathbf{s}_k$, made of four scalar GRFs and a bias, and it outputs $\mathbf{q}_k$, a vector of eight positions and velocities for the HFE and KFE joints of all legs. The inputs are first normalized, then sent to a time buffer, which acts as a 'first-in, first-out' queue. It is fully connected to

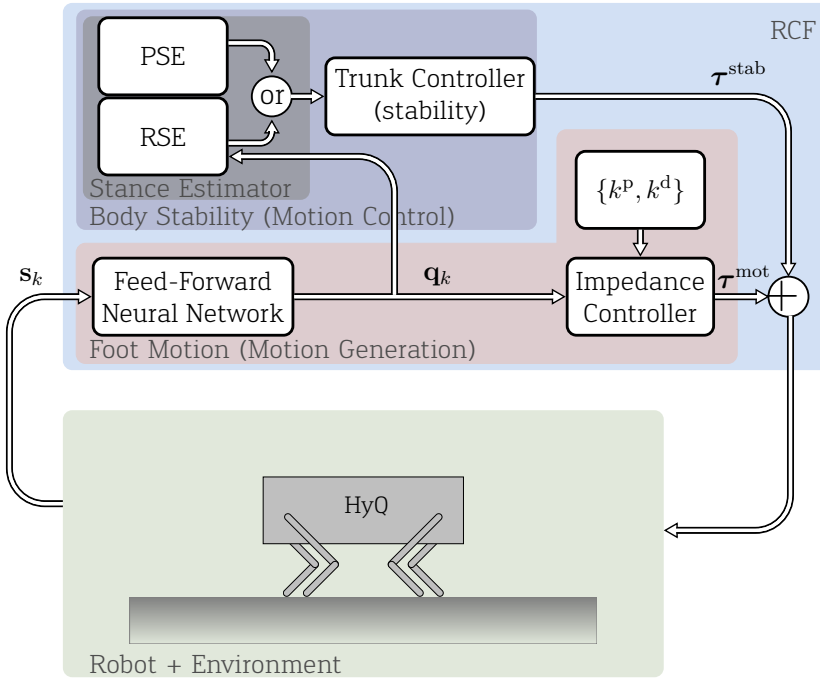


Figure 8.1: The lower box in this diagram illustrates the HyQ robot on a treadmill. The middle part includes the active compliance module for the different joints and the neural network that controls foot trajectories in closed-loop using GRF feedback. The upper part is a functional model of the cerebellum correcting the stance and the balance of the robot.

a hidden layer of hyperbolic tangent neurons, followed by another fully connected readout layer of linear neurons. This feed-forward architecture displays only two parameters, M and N, to tune the memory and the nonlinear hidden projections of the controller's model, respectively. In this chapter, I fix both M and N to 80. This choice guarantees accurate and stable predictions in closed-loop.

The chronology of each experimental trial is divided into three phases. It is similar to what has been presented in Figure 7.3 but with different values for the timing. In the first phase of 120 seconds, the weights of the connections between the hidden and the linear readout layer are trained using the FORCE algorithm (see Section 3.4.2, Equations 3.14 to 3.17) to learn to reproduce target cyclic feet trajectories produced by the HyQ RCF controller Barasuol et al. (2013). In this phase, the robot only uses the target signal to trot. In the second phase of 30 seconds, I switch from the target to the predicted signals, and the algorithm progressively alternates between training and prediction modes until it fully works in prediction at the end of the phase. The final phase also lasts 30 seconds and is dedicated to testing. In this phase, the robot is completely controlled by the architecture presented in Figure 8.1, and the data is recorded for further analysis.

8.2.2 Trunk Controller

The trunk controller module is part of the RCF controller. It performs robot stabilization by correcting the robot's body (or trunk) pose and compensating for its mass. The corrections are, in turn, translated to a torque signal on each of the quadruped joints. To compute these torques, the module also requires information on the stance status, i.e., which leg is in contact with the ground (stance phase) or in the air (swing phase).

8.2.3 Stance Estimator

The neural network is trained to predict the foot trajectories for the trotting pattern presented in Figure 8.2. In that figure, each bar

corresponds to the stance phase, and each blank represents the swing phase. As mentioned above, this information is crucial to produce stabilizing reaction forces. In this chapter, I suggest two different stance estimator models that I will refer to as 'Predictive Stance Estimator' (PSE) and 'Reactive Stance Estimator' (RSE).

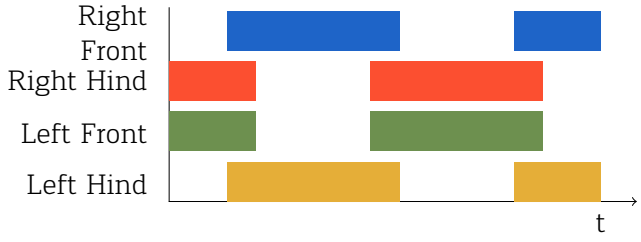


Figure 8.2: Trotting pattern for the four legs of HyQ. Each bar represents a leg stance phase and each blank a swing phase

In the PSE model, the stance is exclusively defined using the desired gait patterns of Figure 8.2, which themsleves depend only on time:

$$\text{PSE} = f(t). \tag{8.1}$$

As inspired by biological evidence, **this cerebellum function is entirely predictive and integrates a spatiotemporal representation of the desired gait with no reactive feedback** induced by the external environment and disturbances.

RSE, however, relies on feedback from the lower neural network. As explained in the introduction section, this model is not biologically plausible and builds on a reactive pathway where the desired posture is communicated from the lower spinal neural network to the cerebellum. However, from an engineering point of view, the resulting architecture has the advantage of removing all dependencies to the target patterns after training. In practice, stance/swing information is computed using the desired foot positions along their vertical axis predicted by the neural network:

$$\text{RSE} = f(z^{\text{FR}},\ z^{\text{FL}},\ z^{\text{HL}},\ z^{\text{HR}}), \tag{8.2}$$

where z is the height of the foot, and indexes correspond to the four different legs. To cope with eventual oscillations, this function includes a threshold coupled with a moving average, whose parameters are tuned heuristically. This dependency can also be simplified to the joint positions and velocities predicted by the neural network after applying direct kinematics:

$$\mathrm{RSE} = f(\mathbf{q}), \tag{8.3}$$

with $\mathbf{q}$, the vector of the desired joint positions and speeds defined in Equation 7.1.

8.3 Results

Together with the team at the *IIT*, we conducted twelve experimental trials divided into two categories of six trials each. The stance status was computed using PSE in the first category, and RSE in the second. In both cases, a disturbance was applied around t = 160 s, i.e. 10 s after the beginning of the testing phase (see Section 8.2.1), using a small delay between the neural network and the motors. Some experiments were realized on a treadmill with different robot forward speeds and some were conducted on the ground with the robot trotting in place. These forward speeds were, however, constant for the whole duration of each trial. Selecting different forward speeds modified the joint trajectories slightly. However, it did not seem to affect the results in any way, and I do not consider it further in this analysis. A video is also provided to demonstrate the robot gaits on the treadmill qualitatively[1].

8.3.1 Robustness

Among the twelve trials, all those using the PSE model succeeded in reproducing the target gait correctly. In contrast, five out of six trials relying on RSE failed in finding a robust attractor, which means that the robot eventually ended up falling before

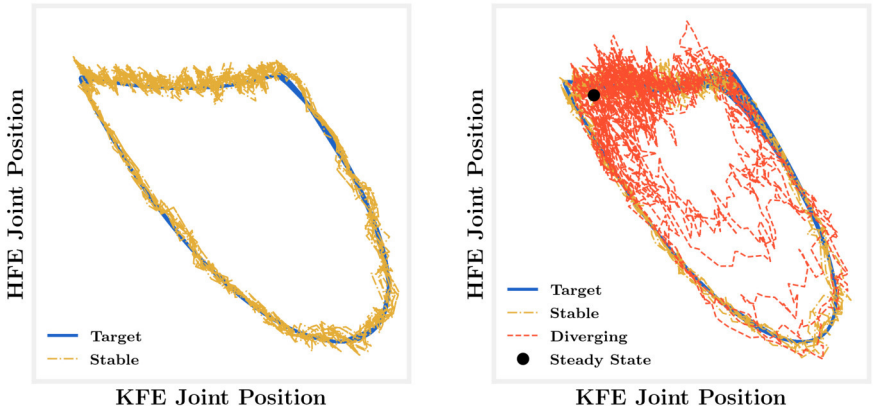
[1] https://youtu.be/prcQL6ukZl8

the end of the testing phase. These results are displayed in Figure 8.3. In this graph, I show the NRMSE between the target and the neural network prediction. A moving average with a window size equal to the gait period is applied on the curve to highlight the global trend. A low NRMSE stands for a good prediction of the joints' position and speed, which can reproduce the results of the target RCF controller. The NRMSE is clearly overshooting at different moments in time during the closing and testing phase for the controller models using RSE. Except for the last one, discussed in the next section, they indicate a divergence from the locomotion limit cycle, leading robot falls or acting chaotically before we needed to press the emergency stop button. It is also important to notice that this divergence does not happen only at the time where the disturbance is applied (t = 160 s) but also before or after it, in reaction to the unpredictability of the environment.

Figure 8.3: NRMSE of neural predictions for all experiments, smoothened with a moving average. The trials with a stability module using the RSE model (red) are diverging during the closing or testing phase but not with the PSE model (green).

To further demonstrate these observations, I illustrate a typical attractor for trials in both categories in Figure 8.4. The instability represented in orange in Figure 8.4b shows how **the limit cycle amplitude decreases until convergence to a steady-state.** Such a point is reached when the robot stabilizes, the position of its feet

becomes constant (by falling or standing still on the ground) and the cerebellum does not trigger a forced alternation in the stance pattern.



(a) Typical attractor with PSE **(b)** Typical attractor with RSE

Figure 8.4: Limit cycles for both models. On the left-hand side, the PSE model (yellow) is stable and follows correctly the target cycle (blue). On the right-hand side, the RSE model starts from an attractor (yellow), close to the target signal (blue), then diverges (orange) until it reaches a steady-state point (black).

8.3.2 Synchronicity

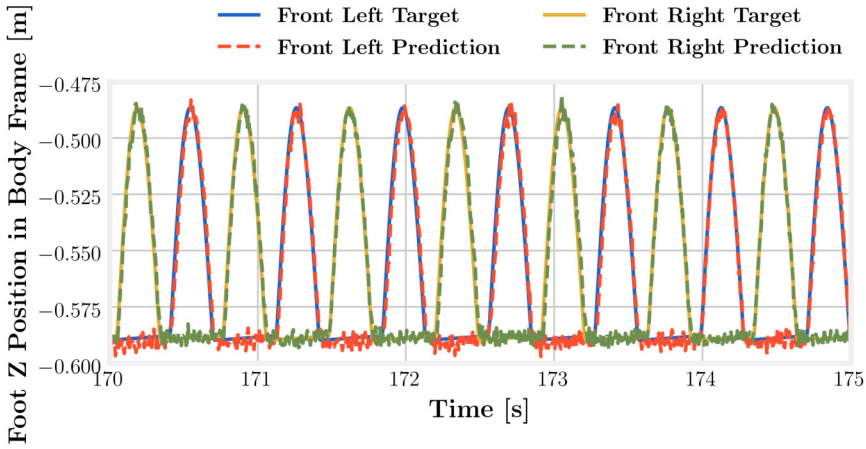
I have mentioned that the red curve on the far right displayed in Figure 8.3 does not suggest a failure. A visual inspection of the robot behavior during the trial shows that it does not correspond to a chaotic or freezing behavior but a progressive loss of the target signal's phase. In Figure 8.5b, I plotted the vertical position of the front feet (for both target and neural prediction) at the end of this trial. The target trajectory has a step height of 10 cm. In comparison, the same signals are represented for a successful trial of the PSE category in Figure 8.5a. **The absence of timing information from the cerebellum action leads to a frequency decrease,**

characterized by a slower robot gait. In contrast, the implicit temporal information embedded in the stability mechanisms of PSE in Figure 8.5a ensures a phase-locking on the stance phase, resulting in a neural network prediction synchronized with the target.

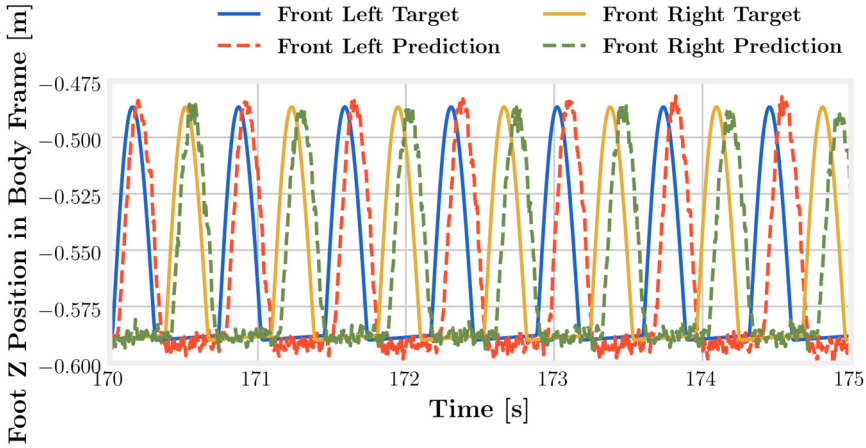
8.4 Discussion

In this chapter, I have discussed a cerebellar model for stability and a closed-loop neural network to generate leg movements based on GRF feedback. Inspired by evidence from biology, I have compared two models to acquire this information: one relying on a predictive pattern of desired stance (PSE) and the second relying on the desired foot positions position along with their vertical component (RSE). I have formulated the hypothesis that the first model should lead to more stable results as biological studies with patients affected by cerebellar ataxia concluded that a predictive notion of leg posture is required to improve locomotion stability. Experiments with both models were conducted on the active compliant quadruped robot HyQ.

Successful applications of neural-based control in robot locomotion generally rely on position control either of stiff robots (Heess et al., 2017; Tan et al., 2018), or of small and light compliant robot (Vandesompele, Urbain, Mahmud, et al., 2019). **On a heavy torque-controlled compliant robot, in contrast, the lift-off cannot be easily guaranteed in the absence of a gravity compensation mechanism.** In other words, the inherent flexibility of the leg joints and the balancing of the robot body can cause the foot to stay on the ground during the desired swing phase. This effect can have a dramatic impact. Firstly, because the vertical amplitude of the lift-off is crucial to avoid tripping on rough terrain or in the presence of obstacles. Secondly, a stable limit cycle entirely depends on the sequence of contact with the ground when using the biologically-inspired neural network with GRF inputs that I sug-



(a) The control signal of the left-front HFE joint with PSE model



(b) The control signal of the left-front HFE joint with RSE model

Figure 8.5: Quality of frequency-locking with PSE (top) and RSE (bottom). In both graphs, the neural predictions for the height of the left-front (orange) and right-front (green) feet are displayed in comparison with the target signal (blue and yellow). In the lower graph, the phase is not locked and the gait frequency decreases with time. This does not happen in the upper graph.

gested. Therefore, I believe that this architecture can contribute to better tackle neural control of locomotion.

Furthermore, an analysis of the stability attractors pointed out that a robust trotting gait can be achieved when using a predictive stance in the stability module. From a dynamic point of view, this indicates that **reflex-based locomotion requires a timing input to stabilize its limit cycle. Despite robustness in amplitude excursions, the model is quite sensible to phase jitter and needs to be corrected with a clock signal.** In the single trial where the robot did not fall, the gait frequency could not be held reliably, and the robot started to slow down with respect to the required frequency. The same explanation can be used to clarify this effect: in closed-loop and without the desired timing pattern coming from the cerebellum model, the system dynamics are determined by the neural connections and the interaction of the robot with its environment. Undesired external delays will accumulate, and frequency locking cannot be guaranteed. In an extreme case, this can lead to robots that slow down until they completely stop, and their limit cycle converges to a steady-state. It can also have a disastrous effect if the phase of the different motor commands does not evolve in synchrony, leading to chaotic behavior and falling.

The experimental results obtained with my robotic models display a **fair correlation with biological observations.** First, they fit **physiological and functional insights about the cerebellum** to work as a predictive circuit, relying on vestibular senses and signals from the cortex but not from the lower limb sensory feedback. Secondly, they **emphasize the role of a clock signal to achieve robust locomotion**. Third, they show that the lack of a predictive Cerebellum model has negative effects such as frequency loss and larger oscillations of the center of mass, which **relates to observations conducted on people with cerebellar ataxia**.

In conclusion, this chapter stresses the importance of stability control in closed-loop neural control of compliant legged robotic locomotion. In particular, it shows how a cerebellum-inspired PSE

mechanism ensures a good lift-off and increases robustness to external disturbance and phase shifting. It is important to note that this work on quadruped locomotion directly inspires from a functional model described in human experiments, for which the presence of CPGs has not been confirmed (Minassian et al., 2017). A better comprehension of how locomotion control evolved from quadrupeds to bipeds should integrate a spinal CPG in the model. Such an architecture could also help to clarify the mixed role of CPGs and Cerebellum to regulate the temporal sequencing in locomotion and should be investigated in future work.

CHAPTER 9

—

Conclusions and Future Work

OK, let me recapitulate...

In this dissertation, I have summarized and discussed the experiments undertaken to explore how mechanical compliance can be exploited for morphological computation in robotic locomotion. This work has three main goals:

- Exploring new strategies for **transferring an embodied locomotion controller from simulation to real robots**.
- Adding new elements on **how mechanical compliance can efficiently contribute to locomotion** of dynamic complex systems, especially in the context of morphological computation.
- Improving the understanding of the **biological processes involved in locomotion**.

The starting point of these investigations has been the selection of three platforms to conduct experiments in locomotion control, as presented in Chapter 4. The Tigrillo robot, created in our lab, provided the support to study calibration methods for efficient transfer of control policy from simulation to the real world (Chapter 5). The MSD networks have supplied a simplified simulation model for extensive research on the relations between stiffness and locomotion performance (Chapter 6). It also demonstrated an efficient methodology for learning and optimizing gait patterns. Backed by this demonstration, I applied the same principles to the state-of-the-art quadruped robot HyQ. Its active compliant mechanism allowed me to further explore the relationship between compliance and locomotion performance on a real robot (Chapter 7), as well as stressing the importance of stability control for closed-loop locomotion and its relation to biological hypotheses (Chapter 8). In this last chapter, I summarize the results obtained for these different topics and conclude with some ideas for future investigations.

Broadly speaking, all these experiments have brought insights on how to work with compliant robots controlled by neural networks. The learning methodology, the relevance of compliance in the locomotion performance and to enable efficient embodied control,

the importance of stabilization in neural locomotion control, or the techniques to carry meaningful results from simulation to reality are among the principal contributions of this dissertation. However, **they should not be seen as definitive conclusions or literal improvements in the view of a specific task benchmark. Rather, they provide a set of observations and proof-of-concept tools to better apprehend how to work with end-to-end neural control in compliant robotic locomotion**. The controllers designed in this thesis **do not directly enter in competition with other existing engineering solutions in robotics locomotion, but suggest a complementary approach to target this specific class of robots and controllers.** Furthermore, there is still some work along the way to demonstrate locomotion systems that are as agile and robust as with classical engineering methods or centralized control but the potential is certainly present. In Figure 9.1, I give a synthetic representation of the main conclusions in my work in the form of a decision tree that should help the design process when implementing end-to-end neural control for bio-inspired compliant locomotion.

Methodology for Learning Locomotion

As presented in Chapter 3, end-to-end closed-loop locomotion models are directly controlling the motor trajectories of a robot by taking the sensors inputs, without using intermediate representation (i.e., without projecting the trajectories in the Cartesian system of reference of the robot via inverse kinematics, for example). Despite an important gain in complexity, this representation has the disadvantage of being hardly interpretable, and one challenge was to find out how to optimize it effectively.

RL is a classic machine learning framework to address this kind of problem. However, the convergence time can be prohibitive, and I wanted to anchor the model in the biological observations on locomotion control, such as the progress of gait through Darwinian evolution, or the evidence of oscillatory circuits involved in locomotion control of various simple animals. To do this, I opted for a

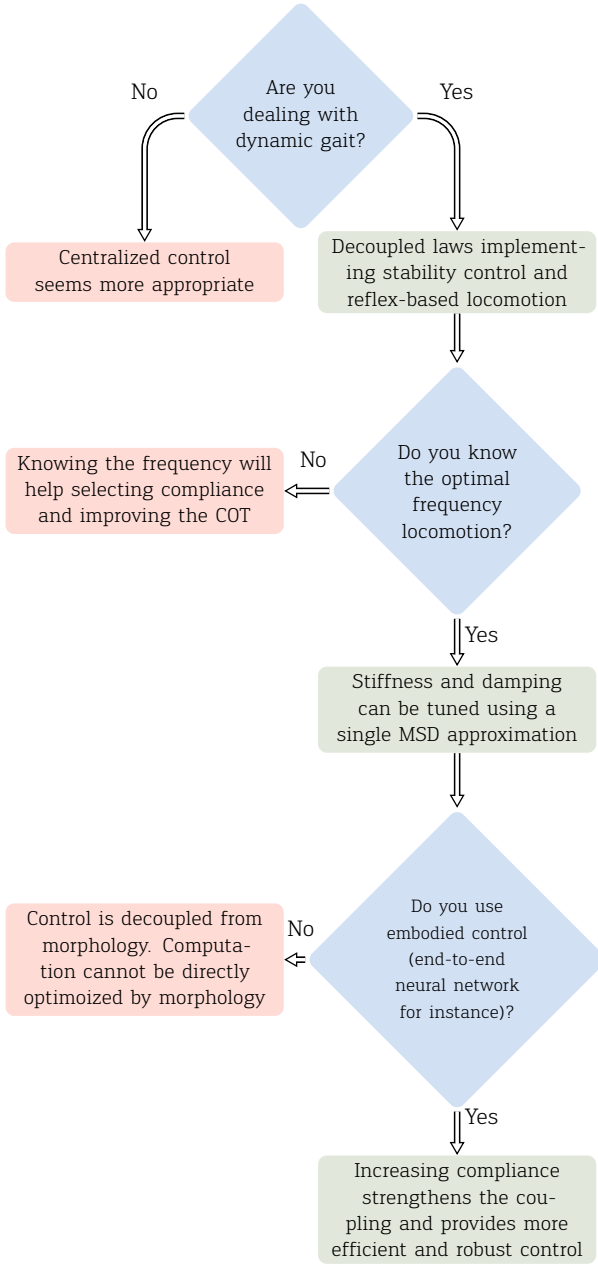


Figure 9.1: This decision tree summarizes the main conclusions from my work regarding the trade-offs between compliance, computation, and locomotion performance. This approach could help to guide the design process when working with compliant robots and embodied neural control.

learning strategy in two steps. In the first phase, a genetic algorithm is used to optimize a robot controlled by a parametric model such as a CPG (or a sinusoidal signal in the case of MSD networks) -loop. The algorithm takes as a reward the speed of the robot and its energy using locomotion metrics detailed in Section 3.6. This way, it is possible to find an optimal trajectory in the actuator's space. This phase is loosely inspired by phylogenetic evolution in biology and the ability of simple lifeforms to be controlled with no afferent feedback. The second phase finds its inspiration in imitation learning during lifetime. In this phase, I have used the FORCE supervised algorithm to teach a neural network to reproduce these target trajectories based on sensor feedback in closed-loop.

This proof-of-concept method has been successfully tested on the three platforms presented in this dissertation. In Chapter 6, I showed that the co-optimization of the morphology and the open-loop controller of MSD structure led to the emergence of surprisingly natural gait patterns and transitions. On the Tigrillo robot, this open-loop optimization also ended up in efficient gaits in Chapter 5, both in optimization and on the real robot, although less sophisticated given the semi-rigidity of the robot. In Chapters 6, 7 and 8, the robustness of the second phase of the learning procedure was demonstrated in simulation and on the HyQ robot. The same results were also observed on Tigrillo but were not discussed in this dissertation.

In conclusion, I believe that **this is a promising approach to connect the world of machine learning (data-driven), with the extensive knowledge acquired in biomechanics or robotics (model-driven)**. However, this method is yet unable to tackle locomotion in complex environments (uneven terrain, presence of obstacles) or to present a high level of controllability (speed, turning angle, lift-off height, ...) and some enhancements with a third step related to exploration and inspired by RL would be relevant to overcome this barrier, as discussed in Section 3.5.

Compliance and Locomotion Performance

In Chapters 6 and 7, I proposed a study on the influence of compliance on locomotion based on open-loop experiments. To set up a clean scope, this analysis has been limited to a locomotion behavior on simple terrains, while not taking complex environments and agility criteria into account. Different angles of approach have been addressed. First of all, **on the MSD structures, a direct link between compliance and robot speed was highlighted. The experiments indicated the presence of an optimal stiffness region**, although an analytical concept of resonance can not be expressed for non-linear systems. The same pattern was observed for different structures with different architectures and complexity levels.

On HyQ, the active compliance mechanism helped to consolidate this argument. By using a single open-loop controller, **I highlighted performance regions in the domain formed by two parameters: joint stiffness and damping. The same relationship between stiffness and speed was observed: speed increases with rigidity, before going through a maximum and decreasing when the robot is too stiff.** This suggests that a simplified single MSD node could be used to give a first approximation of the link between stiffness, damping, and locomotion speed. From this approximation, further study using non-linear simulation models or real experiments can help to determine this optimum more precisely. This concept, however, has not been pushed further, and an accurate mathematical representation should provide theoretical support to explain and exploit the phenomenon.

Compliance and Computational Capacity

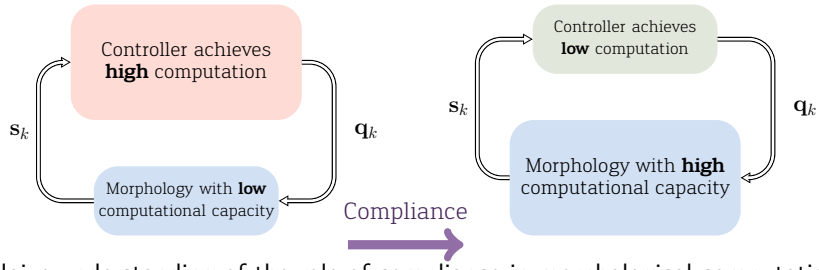
In Chapter 1, I suggested to define morphological computation as:

> Any way of increasing efficiency of computation in terms of energy, memory, time, etc by outsourcing computational tasks to analog physical systems.

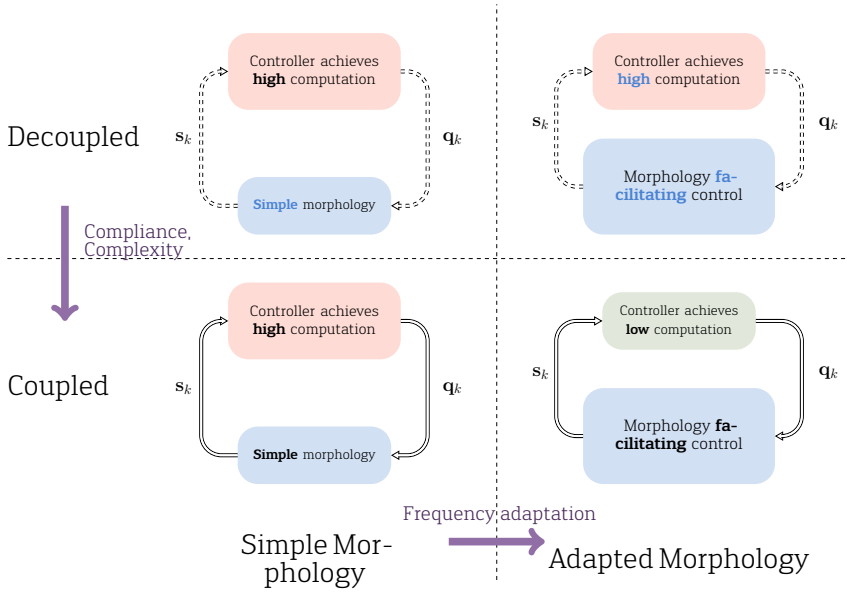
Throughout this dissertation, I have tried to investigate if using compliant instead of rigid bodies could bring this outsourcing phenomenon to light and help to reduce the computational requirements in the controller, as presented in Figure 9.2a. I must admit that, **on the three different platforms understudy, I have failed to demonstrate literal outsourcing of computation between morphology and controller.**

In Chapter 6, I showed that increasing the number of nodes (hence also the number of sensors and actuators) in the MSD structures could lead to more stable locomotion attractors. This effect was taken as the starting point in the search for a better embodied approach.

In Chapter 7, I conducted the same experiment on HyQ by comparing different compliance values and their effect on the minimal computational requirements of the controller (quantified by the non-linearities and the memory in the pre-processing pipeline) in closed-loop. In the graphs presented in Figure 7.7, I failed to see a clear 'migration' of computational capabilities from the controller to the morphology when increasing the compliance. In contrast, robust attractors started to fade out with body rigidity. I believe that **it can be interpreted as the sign of a decoupling between the controller and the morphology that appears for stiffer robots. Increasing the compliance to boost this coupling is, therefore, a necessary condition to enable `morphological computation'.** It helps to interact optimally with the morphology, which is impossible otherwise. It is however not sufficient and the co-tuning of frequency and compliance seems to be another important factor to facilitate the control. In other words, even with compliant robots, we need to find precisely the region of interest in the space defined by the morphology parameters to exploit the interactions correctly. This highly relates to the conclusions drawn in the previous section. In this region only, we can truly benefit from the morphology to implement low-computation and flexible data-based control strategies. A summary of this procedure is presented in Figure 9.2b.



(a) Naive understanding of the role of compliance in morphological computation



(b) Better understanding of the role of compliance in morphological computation

Figure 9.2: Instead of demonstrating a clear outsourcing of computation between morphology and control, compliance can help to improve the coupling between these two entities. In turn, this coupling leads to more robust attractors, which can be smartly exploited if tuning correctly the locomotion gait with the morphology.

Reducing the Sim-to-Real Gap

In the experiments conducted in this thesis, I had to find an equilibrium between the long and tedious optimizations of the EA, which could only be performed efficiently in simulation, on the one hand, and the specific physical properties present only on real robots on the other hand. The difficulty of transferring a controller from simulation to a real robot is a well-known problem in robotics. In a way, the two-steps learning approach discussed in this dissertation partly answers the issue. It allows optimization of the parametric controller in simulation, while the next shorter learning step is carried on the real robot. As a consequence, the inaccuracies encountered when modeling sensors, motors, or robot dynamics can be partially compensated during this phase of supervised learning.

Nevertheless, the first phase of this two-steps learning approach requires a correct approximation of the robot's dynamics. In a sense, it can be suggested that **the transfer function between the robot motors and the sensors must be accurately captured to enable the transfer of a controller from simulation to reality. In Chapter 5, I suggested a calibration method based on this hypothesis** with the Tigrillo robot. A selection of physical parameters known for their clear action on the locomotion dynamics was first conducted (for instance, the legs' friction coefficients, or the stiffness constant of the passive parts). Then, optimization was carried using a sinusoidal excitation signal to align the frequency responses in simulation and reality.

This method showed encouraging results on the Tigrillo robot and has certainly a lot of potential in 'cheap' robot designs, i.e. robots in which the kinematics and dynamics are not completely known, therefore not easily simulated. It could also find interest on tensegrity structures or bio-inspired soft robots. The benefit, however, disappears for stiff robots or robots that are precisely modeled and controlled robots. For instance, this method would have not made sense on HyQ for several reasons:

1. While stiffness and damping was hardly measurable on Trigrillo (cheap spring, with non- uniform parameters and which change with time), HyQ has active compliance, which means that stiffness and damping parameters are under control.

2. HyQ is heavy with feet covered with rubber and most experiments are generally conducted on a rubber treadmill. Therefore, slip is generally under control and easy to simulate.

3. HyQ has been used in several research tracks for many years and the accuracy of the simulation model has become quite good (Frigerio et al., 2017), which makes the method useless in this case.

Among the possible improvement for the method, I can mention the generalization of the excitation signal used in the procedure to a broad range of actuation frequencies. This could lead to a better formalization of the scope and limitations in the future. Secondly, I also suggested that a classical simulator could be concatenated to a neural network to better carry this calibration step in future work. This could allow representing the system's non-linearities better while keeping enough prior knowledge on the physics laws, and boosting the transfer between simulation and the real world.

Stabilization of Closed-Loop Reflex-Based Locomotion

In Chapter 8, I have addressed another crucial point of locomotion: stability. Based on functional and physiological observations of motor control in mammals, I have analyzed the influence of a cerebellar model on the equilibrium of HyQ reflex-based locomotion. In particular, this demonstrated that such model was appropriate in combination with a reflex-based neural network using only GRF inputs.

Additionally, an important point regarding the model's structure was highlighted. **The experiments demonstrated the importance of a spatio-temporal representation of the locomotion sequence in the 'Cerebellum' model of a robot.** This overlaps with some evidence about the biological Cerebellum structure and experiments

performed on patients with ataxia. I believe that **combining the architecture based on a neural network to generate closed-loop motor trajectories to a top-down stability mechanism for the equilibrium is an encouraging path for future investigations.**

Future Work

Hardware

The design and implementation work carried out in this dissertation has mainly focused on passive quadrupedal robots. Tigrillo and HyQ are both composed of a rigid body attached to four two-segments legs. However, **studies have also shown the crucial importance of a flexible spine** in the process of bounding and gallop but this important element has not been discussed in this dissertation. A future version of a passive quadruped robot should cover this feature to allow further studies. Besides, **Tigrillo does not have a mechanism for leg retraction**. This makes behaviors such as walking and obstacle avoidance, difficult to implement. A future design of a compliant leg using only a single DOF could trigger this ability while keeping a reduced actuation complexity, hence a low cost. In further designs, it would be useful to envision a compliant leg with a single DOF implementing this ability.

Regarding electronics, some major improvements could also be carried in future work. For instance, the importance of using GRFs as the primary trigger for reflex-based locomotion has been largely emphasized. However, in practice on HyQ, these signals were determined after post-processing motor encoders and load cells (extremely sensitive to damages) located in the joints' actuators. A simple, yet useful improvement would be to **equip the next robot with resistive, capacitive, or piezoelectric sensors in the feet, in order to directly measure the GRFs**. Arranged in a grid structure, the spatial information regarding the nature of the impact with the ground could be captured and exploited by the reflex feed-forward network and improve the performance. The relevance of this device has been demonstrated in prosthesis neural control (Valle et al., 2018) but remained to be thoroughly discussed in locomotion.

Theoretical Morphological Computation

Based on previous work demonstrating the ability of MSD structures to act as time-invariant filters with fading memory, I have begun to quantify this effect for locomotion. However, I have only analyzed a few parameters (spring stiffness, number of nodes), independently from each other. Further evaluation would be required to give a clear order of magnitude for morphology requirements in future experiments in the field of morphological computation.

Besides, the experiments carried out on the MSD networks and HyQ made use of a closed-loop framework where the motor actions are computed directly based on barely preprocessed raw sensors. In contrast, more conventional approaches use high-level features with a physical meaning. The crucial importance of the temporal content included in these features has been discussed with HyQ. Nonetheless, **an extensive investigation of the spatio-temporal complexity of the sensor representation and its contribution to the performance seems relevant for future work in embodied locomotion.**

In the same experiments, I also described **the influence of stiffness and damping on the open-loop locomotion of non-linear systems**, and I compared it to the resonance in a single MSD model. However, this relation **still lacks a rigorous mathematical basis, and consequent work should be carried in this direction.**

Learning and Transferring Controllers

The parametric model used in the EA simulation optimization described in Chapter 3 works purely in open-loop. However, some CPGs models which can handle a feedback signal have been validated on rough terrains. Therefore, **it seems useful that feedback can be already exploited during the first learning phase**. However, this requires an important adaptation of the learning procedure. Improving the two-steps learning approach with this consideration in mind is an interesting track for future experiments

On a different note, I have claimed the good performance of EA in Chapter 2 and motivated this choice in the description of the control framework in Chapter 3 based on general work from machine learning. However, **a clear comparison with RL performance was not conducted on the compliant robots of this thesis and could be scientifically relevant**. Besides, the conclusions acquired about the selection of optimal compliance, **the controller complexity, and the stability mechanism, could be directly exploited in a RL framework**. Insights from Chapters 7 and 8 of this dissertation could serve in designing an efficient architecture to carry out this study on HyQ. To my knowledge, this work has been started in 2020 at the *IIT* based on some development carried during my Ph.D.[1].

[1] https://github.ugent.be/gurbain/hyq_ml/tree/gym-rl

Epilogue

The technological advances in the last century have triggered the imagination of many people: ensuring sufficient energy and food production to provide a decent life to everyone, revolutionizing healthcare, exploring new frontiers, discovering and navigating the universe, mastering sub-atomical scales, understanding the human mind,... The list of expectations from society is long. In all these endeavors, robotics has taken a central place, and it even became one symbol of human technical capabilities, at the crossing of mechanical engineering, information technology, and biological inspiration.

Nonetheless, like all symbols, the robot also came with its lot of mystification and approximate understanding. Feared on one hand for its potential to replace human jobs and functions, idolized on the other like it would solve all current problems, it has become difficult to concretely appreciate its true impact on society and its place in our future. In this dissertation, I have tried to give some insights on which direction robotics could take in the next few years while restricting myself to the field of locomotion.

As indicated in the first chapters, soft and flexible structures have a huge potential in robotics and I am convinced that their progress will not stop anytime soon. This development could enable a progressive transition from hierarchical control, materialized in the electronics substrate, to adaptive control processes, anchored in the morphological properties of robots. However, the trade-offs in that evolution are not completely understood and I have tried to bring a modest contribution on the matter for different robotic platforms in this dissertation.

Besides, with an increasing potential for simulation, it is difficult

to predict if the control capabilities of the simulated and real robots will converge hand in hand, or if the gap between them will increase in the future. In this line of thoughts, I have suggested an idea to calibrate a small compliant robot between these two worlds. In the current situation, robotics is still struggling to take full advantage of recent developments in AI and the question about this transfer will strongly impact the opportunities that will appear in the next few years.

In the end, this dissertation is also a reminder that building a robot does not only have a utilitarian objective. Designing its shape, assembling its pieces of plastic and metal, writing the code for its sequence of movement, all these complex steps also constitute a tool to validate diverse fundamental hypotheses and understand the complexity of living organisms. In that sense, engineering provides an endless source of knowledge about the world we live in, and I hope that I could give the reader a short impression of this beautiful potential.

References

Ajallooeian, M., van den Kieboom, J., Mukovskiy, A., Giese, M. A., & Ijspeert, A. J. (2013). A general family of morphed nonlinear phase oscillators with arbitrary limit cycle shape. Physica D: Nonlinear Phenomena, 263, 41–56.

Alpaydin, E. (2010). Introduction to Machine Learning (2nd ed.). The MIT Press.

Aoi, S., Manoonpong, P., Ambe, Y., Matsuno, F., & Wörgötter, F. (2017). Adaptive control strategies for interlimb coordination in legged robots: A review. Frontiers in Neurorobotics, 11, 1–21.

Aschenbeck, K. S., Kern, N. I., Bachmann, R. J., & Quinn, R. D. (2006). Design of a quadruped robot driven by air muscles. In IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob 2006 (pp. 875–880).

Bakker, M., Allum, J. H., Visser, J. E., Grüneberg, C., van de Warrenburg, B. P., Kremer, B. H., & Bloem, B. R. (2006). Postural responses to multidirectional stance perturbations in cerebellar ataxia. Experimental Neurology, 202(1), 21–35.

Barasuol, V., Buchli, J., Semini, C., Frigerio, M., de Pieri, E. R., & Caldwell, D. G. (2013). A reactive controller framework for quadrupedal locomotion on challenging terrain. In IEEE International Conference on Robotics and Automation, ICRA 2013 (pp. 2554–2561).

Barasuol, V., Camurri, M., Bazeille, S., Caldwell, D. G., & Semini, C. (2015). Reactive trotting with foot placement corrections through visual pattern classification. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015 (pp. 5734–5741).

Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., & Maass, W. (2018). Long short-term memory and learning-to-learn in networks of spiking neurons. In Advances in neural information processing systems (pp. 787–797).

Bellman, R. (1957). Dynamic programming. Princeton University Press.

Bicchi, A., & Tonietti, G. (2004). Fast and "soft-arm" tactics [robot arm design]. IEEE Robotics Automation Magazine, 11(2), 22–33.

Bieze, T. M., Largilliere, F., Kruszewski, A., Zhang, Z., Merzouki, R., & Duriez, C. (2018). Finite element method-based kinematics and closed-loop control of soft, continuum manipulators. Soft robotics, 5(3), 348–364.

Bledt, G., Powell, M. J., Katz, B., Carlo, J. D., Wensing, P. M., Kim, S., ... Kim, S. (2018). MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018 (pp. 2245–2252).

Boaventura, T., Focchi, M., Frigerio, M., Buchli, J., Semini, C., Medrano-Cerda, G. A., & Caldwell, D. G. (2012). On the role of load motion compensation in high-performance force control. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012 (pp. 4066–4071).

Boaventura, T., Medrano-Cerda, G. A., Semini, C., Buchli, J., Caldwell, D. G., Cunha, T. B., ... Caldwell, D. G. (2013). Stability and performance of the compliance controller of the quadruped robot HyQ. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2013 (pp. 1458–1464).

Boaventura, T., Semini, C., Buchli, J., Frigerio, M., Focchi, M., & Caldwell, D. G. (2012). Dynamic torque control of a hydraulic quadruped robot. In IEEE International Conference on Robotics and Automation, ICRA 2012 (pp. 1889–1894).

Bongard, J., Zykov, V., & Lipson, H. (2006). Resilient Machines Through Continuous Self-Modeling. Science, 314(5802), 1118–121.

Borel, L., Harlay, F., Lopez, C., Magnan, J., Chays, A., & Lacour, M. (2004). Walking performance of vestibular-defective patients before and after unilateral vestibular neurotomy. Behavioural Brain Research, 150(1-2), 191–200.

Brown, T. G. (1911). The Intrinsic Factors in the Act of Progression in the Mammal. Proceedings of the Royal Society B: Biological Sciences, 84(572), 308–319.

Brunner, D., Soriano, M. C., Mirasso, C. R., & Fischer, I. (2013). Parallel photonic information processing at gigabyte per second data rates using transient states. Nature Communications, 4, 1364.

Buchli, J., Iida, F., & Ijspeert, A. J. (2006). Finding resonance: Adaptive frequency oscillators for dynamic legged locomotion. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006 (pp. 3903–3909).

Buchli, J., Righetti, L., & Ijspeert, A. J. (2006). Engineering entrainment and adaptation in limit cycle systems: From biological inspiration to applications in robotics. Biological Cybernetics, 95(6), 645–664.

Burkitt, A. N. (2006). A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. Biological Cybernetics, 95(1), 1–19.

Calisti, M., Picardi, G., & Laschi, C. (2017). Fundamentals of soft robot locomotion. Journal of The Royal Society Interface, 14(130).

Caluwaerts, K., Despraz, J., Iscen, A., Sabelhaus, A. P., Bruce, J., Schrauwen, B., & SunSpiral, V. (2014). Design and control of compliant tensegrity robots through simulation and hardware validation. Journal of The Royal Society Interface, 11(98).

Caluwaerts, K., D'Haene, M., Verstraeten, D., & Schrauwen, B. (2012). Locomotion Without a Brain: Physical Reservoir Computing in Tensegrity Structures. Artificial Life, 19(1), 35–66.

Carbajal, J. P. (2012). Harnessing nonlinearities: behavior generation from natural dynamics.

Cham, J. G., Karpick, J. K., & Cutkosky, M. R. (2004). Stride period adaptation of a biomimetic running hexapod. International Journal of Robotics Research, 23(2), 141–153.

Chatzidimitriou, K. C., & Mitkas, P. A. (2013). Adaptive reservoir computing through evolution and learning. Neurocomputing, 103, 198–209.

Chatzidimitriou, K. C., Partalas, I., Mitkas, P. A., & Vlahavas, I. (2011). Transferring evolved reservoir features in reinforcement learning tasks. In European Workshop on Reinforcement Learning (pp. 213–224).

Chen, D., Li, N., Wang, H., & Chen, L. (2017). Effect of Flexible Spine Motion on Energy Efficiency in Quadruped Running. Journal of Bionic Engineering, 14(4), 716–725.

Cheney, N., Bongard, J., SunSpiral, V., & Lipson, H. (2018). Scalable co-optimization of morphology and control in embodied machines. Journal of the Royal Society Interface, 15(143), 1–15.

Chitta, S., Marder-Eppstein, E., Meeussen, W., Pradeep, V., Rodríguez Tsouroukdissian, A., Bohren, J., ... Fernandez Perdomo, E. (2017). ros_control: A generic and simple control framework for ROS. The Journal of Open Source Software, 2(20), 456.

Chow, J. H. (2003). Modeling, analysis and control of dynamic systems. Wiley New York.

Clune, J., Mouret, J.-B., & Lipson, H. (2013). The evolutionary origins of modularity. Proceedings of the Royal Society B: Biological Sciences, 280(1755).

Collins, S., Ruina, A., Tedrake, R., & Wisse, M. (2005). Efficient bipedal robots based on passive-dynamic walkers. Science, 307(5712), 1082–1085.

Conde, C., Zinn, M., Khatib, O., Roth, B., & Salisbury, J. K. (2004). Playing it safe [human-friendly robots]. Robotics & Automation Magazine, IEEE, 11(2), 12–21.

Crnkovic, G. (2010). Constructive research and info-computational knowledge generation. In Model-Based Reasoning in Science and Technology (pp. 359–380).

Crutchfield, J. P., Ditto, W. L., & Sinha, S. (2010). Introduction to focus issue: Intrinsic and designed computation: Information processing in dynamical systems-beyond the digital hegemony. Chaos, 20(3), 1–6.

Culha, U., & Saranli, U. (2011). Quadrupedal bounding with an actuated spinal joint. In IEEE International Conference on Robotics and Automation, ICRA 2011 (pp. 1392–1397).

Cully, A., Clune, J., Tarapore, D., & Mouret, J.-B. (2015). Robots that can adapt like animals. Nature, 521(7553), 503–507.

Dahiya, R. S., Metta, G., Valle, M., & Sandini, G. (2010). Tactile sensing-from humans to humanoids. IEEE Transactions on Robotics, 26(1), 1–20.

Dambre, J., Verstraeten, D., Schrauwen, B., & Massar, S. (2012). Information processing capacity of dynamical systems. Scientific Reports, 2, 514.

Danner, S. M., Wilshin, S. D., Shevtsova, N. A., & Rybak, I. A. (2016). Central control of interlimb coordination and speed-dependent gait expression in quadrupeds. Journal of Physiology, 594(23), 6947–6967.

Dasgupta, S., Goldschmidt, D., Wörgötter, F., & Manoonpong, P. (2015). Distributed recurrent neural forward models with synaptic adaptation and CPG-based control for complex behaviors of walking robots. Frontiers in Neurorobotics, 9, 10.

Davies, M., Srinivasa, N., Lin, T. H., Chinya, G., Cao, Y., Choday, S. H., ... Wang, H. (2018). Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. IEEE Micro, 38(1), 82–99.

Davison, A. P., Brüderle, D., Eppler, J., Kremkow, J., Muller, E., Pecevski, D., ... Yger, P. (2008). PyNN: A Common Interface for Neuronal Network Simulators. Frontiers in Neuroinformatics, 2, 11.

Degallier, S., & Ijspeert, A. (2010). Modeling discrete and rhythmic movements through motor primitives: A review. Biological cybernetics, 103(4), 319–338.

Degrave, J., Burm, M., Kindermans, P. J., Dambre, J., & wyffels, F. (2015). Transfer learning of gaits on a quadrupedal robot. Adaptive Behavior, 23(2), 69–82.

Degrave, J., Caluwaerts, K., Dambre, J., & wyffels, F. (2015). Developing an embodied gait on a compliant quadrupedal robot. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015 (pp. 4486–4491).

Degrave, J., Van Cauwenbergh, R., wyffels, F., Waegeman, T., & Schrauwen, B. (2013). Terrain Classification for a Quadruped Robot. In IEEE International Conference on Machine Learning and Applications, ICMLA 2013 (pp. 185–190).

Delvolvé, I., Branchereau, P., Dubuc, R., & Cabelguen, J. M. (1999). Fictive rhythmic motor patterns induced by NMDA in an in vitro brain stem-spinal cord preparation from an adult urodele. Journal of Neurophysiology, 82(2), 1074–1077.

Deng, Q., Wang, S., Xu, W., Mo, J., & Liang, Q. (2012). Quasi passive bounding of a quadruped model with articulated spine. Mechanism and Machine Theory, 52, 232–242.

Dickinson, M. H., Farley, C. T., Full, R. J., Koehl, M. A. R., Kram, R., & Lehman, S. (2000). How Animals Move: An Integrative View. Science, 288(5463), 100–106.

Dominici, N., Ivanenko, Y. P., Cappellini, G., D'Avella, A., Mondì, V., Cicchese, M., ... Lacquaniti, F. (2011). Locomotor primitives in newborn babies and their development. Science, 334(6058), 997–999.

Dosovitskiy, A., & Koltun, V. (2016). Learning to Act by Predicting the Future. arXiv preprint arXiv:1611.01779.

Downing, K. L. (2015). Intelligence Emerging: Adaptivity and Search in Evolving Neural Systems. The MIT Press.

Doya, K. (2000). Reinforcement learning in continuous time and space. Neural Computation, 12(1), 219–245.

Drew, T., Prentice, S., & Schepens, B. (2004). Cortical and brainstem control of locomotion. Progress in Brain Research, 143(03), 251–261.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., & Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In International Conference on Machine Learning, ICML 2016 (pp. 1329–1338).

Duperret, J., & Koditschek, D. E. (2017). Empirical validation of a spined sagittal-plane quadrupedal model. In IEEE International Conference on Robotics and Automation, ICRA 2017 (pp. 1058–1064).

Dzeladini, F., van den Kieboom, J., & Ijspeert, A. J. (2014). The contribution of a central pattern generator in a reflex-based neuromuscular model. Frontiers in Human Neuroscience, 8, 371.

Eckert, P., & Ijspeert, A. J. (2019). Benchmarking Agility for Multi-legged Terrestrial Robots. IEEE Transactions on Robotics, 35(2), 529–535.

Eckert, P., Schmerbauch, A. E., Horvat, T., Söhnel, K., Fischer, M. S., Witte, H., & Ijspeert, A. J. (2018). Towards rich motion skills with the lightweight quadruped robot Serval - A Design, Control and Experimental Study. In (pp. 41–55).

Eckert, P., Spröwitz, A., Witte, H., & Ijspeert, A. J. (2015). Comparing the effect of different spine and leg designs for a small bounding quadruped robot. In IEEE International Conference on Robotics and Automatio, ICRA 2015 (pp. 3128–3133).

Eder, M., Hisch, F., & Hauser, H. (2018). Morphological computation-based control of a modular, pneumatically driven, soft robotic arm. Advanced Robotics, 32(7), 375–385.

Edwards, K. (2002). Compliant mechanisms. Materials & Design, 23(3), 345.

Einevoll, G. T., Destexhe, A., Diesmann, M., Grün, S., Jirsa, V., de Kamps, M., ... Schürmann, F. (2019). The Scientific Case for Brain Simulations. Neuron, 102(4), 735–744.

Ekeberg, O., & Pearson, K. G. (2005). Computer simulation of stepping in the hind legs of the cat: an examination of mechanisms regulating the stance-to-swing transition. Journal of Neurophysiology, 94(6), 4256–68.

Eschenauer, H., Juhani, K., & Osyczka, A. (2012). Multicriteria Design Optimization: Procedures and Applications. Springer Science & Business Media.

Espinal, A., Rostro-Gonzalez, H., Carpio, M., Guerra-Hernandez, E. I., Ornelas-Rodriguez, M., & Sotelo-Figueroa, M. (2016, July). Design of spiking central pattern generators for multiple locomotion gaits in hexapod robots by christiansen grammar evolution. Frontiers in Neurorobotics, 10, 6.

Fahmi, S., Focchi, M., Radulescu, A., Fink, G., Barasuol, V., & Semini, C. (2019). STANCE: Locomotion adaptation over soft terrain. IEEE Transactions in Robotics, 1–15.

Farshidian, F., Jelavic, E., Satapathy, A., Giftthaler, M., & Buchli, J. (2017). Real-Time motion planning of legged robots: A model predictive control approach. In IEEE-RAS International Conference on Humanoid Robots, Humanoids 2017 (pp. 577–584).

Fernando, C., & Sojakka, S. (2003). Pattern recognition in a bucket. In European Conference on Artificial Life (pp. 588–597).

Fernández Cara, E., & Zuazua Iriondo, E. (2003). Control theory: History, mathematical achievements and perspectives. Boletìn de la Sociedad Española de Matemática Aplicada, 26, 79–140.

Floreano, D., Ijspeert, A. J., & Schaal, S. (2014). Robotics and neuroscience. Current Biology, 24(18), 910–920.

Focchi, M., Barasuol, V., Havoutis, I., Buchli, J., Semini, C., & Caldwell, D. G. (2013). Local Reflex Generation for Obstacle Negotiation in Quadrupedal Locomotion. Nature-Inspired Mobile Robotics, 443–450.

Focchi, M., Boaventura, T., Semini, C., Frigerio, M., Buchli, J., & Caldwell, D. G. (2012). Torque-control based compliant actuation of a quadruped robot. In International Workshop on Advanced Motion Control, AMC 2012 (pp. 1–6).

Focchi, M., Orsolino, R., Camurri, M., Barasuol, V., Mastalli, C., Caldwell, D. G., & Semini, C. (2020). Heuristic Planning for Rough Terrain Locomotion in Presence of External Disturbances and Variable Perception Quality. In Advances in Robotics Research: From Lab to Market (pp. 165–209). Springer.

Frigerio, M., Barasuol, V., Focchi, M., Caldwell, D. G., & Semini, C. (2017). Validation of computer simulations of the HyQ robot. In International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, CLAWAR 2017 (pp. 415–422).

Fukui, T., Fujisawa, H., Otaka, K., & Fukuoka, Y. (2019). Autonomous gait transition and galloping over unperceived obstacles of a quadruped robot with CPG modulated by vestibular feedback. Robotics and Autonomous Systems, 111, 1–19.

Fukuoka, Y., & Kimura, H. (2009). Dynamic locomotion of a biomorphic quadruped 'Tekken' robot using various gaits: Walk, trot, free-gait and bound. Applied Bionics and Biomechanics, 6(1), 63–71.

Fukuoka, Y., Kimura, H., Hada, Y., & Takase, K. (2003). Adaptive dynamic walking of a quadruped robot 'Tekken' on irregular terrain using a neural system model. In IEEE International Conference on Robotics and Automation, ICRA 2003 (Vol. 2, pp. 2037–2042).

Furber, S. B., Lester, D. R., Plana, L. A., Garside, J. D., Painkras, E., Temple, S., & Brown, A. D. (2013). Overview of the SpiNNaker System Architecture. IEEE Transactions on Computers, 62(12), 2454–2467.

Füchslin, R. M., Dzyakanchuk, A., Flumini, D., Hauser, H., Hunt, K. J., Luchsinger, R. H., ... Walker, R. (2013). Morphological

computation and morphological control: Steps toward a formal theory and applications. Artificial Life, 19(1), 9–34.

Galloway, K. C., Clark, J. E., Yim, M., & Koditschek, D. E. (2011). Experimental investigations into the role of passive variable compliant legs for dynamic robotic locomotion. In IEEE International Conference on Robotics and Automation, ICRA 2011 (pp. 1243–1249).

Garber, L. (2013). Robot OS: A new day for robot design. Computer, 46(12), 16–20.

Gay, S., Santos-Victor, J., & Ijspeert, A. J. (2013). Learning robot gait stability using neural networks as sensory feedback function for Central Pattern Generators. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2013 (pp. 194–201).

Geijtenbeek, T., & Pronost, N. (2012). Interactive character animation using simulated physics: A state-of-the-art review. Computer Graphics Forum, 31(8), 2492–2515.

Geijtenbeek, T., van de Panne, M., & van der Stappen, F. (2013). Flexible muscle-based locomotion for bipedal creatures. ACM Transactions on Graphics, 32(6), 1–11.

Georgiou, P., Triantis, I. F., Constandinou, T. G., & Toumazou, C. (2007). Spiking Chemical Sensor (SCS): A new platform for neurochemical sensing. In IEEE-EMBS International Conference on Neural Engineering (pp. 126–129).

Gewaltig, M.-O., & Diesmann, M. (2007). NEST (NEural Simulation Tool). Scholarpedia, 2(4), 1430.

Geyer, H., & Herr, H. (2010). A Muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 18(3), 263–273.

Ghazi-Zahedi, K., Haeufle, D. F., Montúfar, G., Schmitt, S., & Ay, N. (2016). Evaluating morphological computation in muscle and DC-motor driven models of hopping movements. Frontiers in Robotics and AI, 3, 1–12.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In International Conference on Artificial Intelligence and Statistics (pp. 315–323).

Grandia, R., Farshidian, F., Ranftl, R., & Hutter, M. (2019). Feedback mpc for torque-controlled legged robots. arXiv preprint arXiv:1905.06144.

Grillner, S. (1975). Locomotion in vertebrates: central mechanisms and reflex interaction. Physiological reviews, 55(2), 247–304.

Grillner, S., & Wallen, P. (1985). Central Pattern Generators for Locomotion, with Special Reference to Vertebrates. Annual Review of Neuroscience, 8(1), 233–261.

Ha, D. (2019). Reinforcement learning for improving agent design. Artificial life, 25(4), 352–365.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., ... Levine, S. (2018). Soft Actor-Critic Algorithms and Applications. arXiv preprint arXiv:1812.05905.

Haeufle, D. F., Günther, M., Bayer, A., & Schmitt, S. (2014). Hill-type muscle model with serial damping and eccentric force-velocity relation. Journal of Biomechanics, 47(6), 1531–1536.

Hansen, N. (2006). The CMA Evolution Strategy: A Comparing Review. In Towards a new evolutionary computation (Vol. 192, pp. 75–102).

Hansen, N. (2016). The CMA Evolution Strategy: A Tutorial. arXiv preprint arXiv:1604.00772.

Hassabis, D., Kumaran, D., Summerfield, C., & Botvinick, M. (2017). Neuroscience-Inspired Artificial Intelligence. Neuron, 95(2), 245–258.

Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R., & Maass, W. (2011). Towards a theoretical foundation for morphological computation with compliant bodies. Biological Cybernetics, 105(5-6), 355–370.

Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R., & Maass, W. (2012). The role of feedback in morphological computation with compliant bodies. Biological Cybernetics, 106(10), 595–613.

Hawkins, J., & Blakeslee, S. (2007). On intelligence: How a New Understanding of the Brain will lead to Truly Intelligent Machines. Macmillan.

Haykin, S. S., & Simon. (1996). Adaptive filter theory. Prentice Hall.

Heess, N., TB, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., … Silver, D. (2017). Emergence of Locomotion Behaviours in Rich Environments. arXiv preprint arXiv:1707.02286.

Heess, N., Wayne, G., Tassa, Y., Lillicrap, T. P., Riedmiller, M., & Silver, D. (2016). Learning and Transfer of Modulated Locomotor Controllers. arXiv preprint arXiv:1610.05182.

Heijmink, E., Radulescu, A., Ponton, B., Barasuol, V., Caldwell, D. G., & Semini, C. (2017). Learning optimal gait parameters and impedance profiles for legged locomotion. In IEEE-RAS International Conference on Humanoid Robots, Humanoids 2017 (pp. 339–346).

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. In AAAI Conference on Artificial Intelligence, AAAI 2018 (pp. 3207–3214).

Hermans, M., Schrauwen, B., Bienstman, P., & Dambre, J. (2014). Automated design of complex dynamic systems. PLOS ONE, 9(1), 1–11.

Hill, A. (1938). The heat of shortening and the dynamic constants of muscle. Proceedings of the Royal Society B: Biological Sciences, 126(843), 136–195.

Hines, M. L., & Carnevale, N. T. (1997). The NEURON Simulation Environment. Neural computation, 9(6), 1179–1209.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and

excitation in nerve. Bulletin of Mathematical Biology, 52(1-2), 25–71.

Hoffmann, M., & Müller, V. C. (2017). Simple or complex bodies? trade-offs in exploiting body morphology for control. In Representation and reality in humans, other living organisms and intelligent machines (pp. 335–345). Springer.

Hoffmann, M., & Müller, V. C. (2014). Trade-Offs in Exploiting Body Morphology for Control: from Simple Bodies and Model-Based Control to Complex Bodies with Model-Free Distributed Control Schemes. arXiv preprint arXiv:1411.2276.

Hoffmann, M., & Simanek, J. (2017). The merits of passive compliant joints in legged locomotion: fast learning, superior energy efficiency and versatile sensing in a quadruped robot. Journal of Bionic Engineering, 14(1), 1–14.

Hogan, N. (1984). Impedance Control: An Approach to Manipulation. In IEEE American Control Conference (pp. 304–313).

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. Neural Networks, 2, 985–990.

Hughes, J., Culha, U., Giardina, F., Guenther, F., Rosendo, A., & Iida, F. (2016). Soft Manipulators and Grippers: A Review. Frontiers in Robotics and AI, 3, 1–12.

Hurst, J. W., & Rizzi, A. A. (2008). Series compliance for an efficient running gait. IEEE Robotics and Automation Magazine, 15(3), 42–51.

Hutter, M., Gehring, C., Bloesch, M., Hoepflinger, M., Remy, D. C., & Siegwart, R. (2012). StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In Adaptive mobile robotics (pp. 483–490).

Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, D., Tsounis, V., ... Hoepflinger, M. (2016). ANYmal - A highly mobile and dynamic quadrupedal robot. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016 (pp. 38–44).

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., & Hutter, M. (2019). Learning agile and dynamic motor skills for legged robots. Science Robotics, 4(26), 1–20.

Hyndman, R. J., & Koehler, A. B. (2005). Another look at measures of forecast accuracy. International Journal of Forecasting, 22, 679–688.

Iida, F., & Pfeifer, R. (2004). Cheap rapid locomotion of a quadruped robot: Self-stabilization of bounding gait. In Intelligent Autonomous Systems (pp. 642–649).

Iida, F., & Pfeifer, R. (2006). Sensing through body dynamics. Robotics and Autonomous Systems, 54(8), 631–640.

Ijspeert, A. J. (2008). Central pattern generators for locomotion control in animals and robots: A review. Neural Networks, 21(4), 642–653.

Ijspeert, A. J., Crespi, A., & Cabelguen, J.-M. (2005). Simulation and robotics studies of salamander locomotion. Neuroinformatics, 3(3), 171–195.

Isik, K., He, S., Ho, J., & Sentis, L. (2017). Re-Engineering a high performance electrical series elastic actuator for low-cost industrial applications. Actuators, 6(1), 5.

Ito, M. (2006). Cerebellar circuitry as a neuronal machine. Progress in Neurobiology, 78(3-5), 272–303.

Izhikevich, E. M. (2003). Simple model of spiking neurons. IEEE Transactions on Neural Networks, 14(6), 1569–1572.

Jahn, K., Deutschländer, A., Stephan, T., Kalla, R., Hüfner, K., Wagner, J., ... Brandt, T. (2008). Supraspinal locomotor control in quadrupeds and humans. Progress in Brain Research, 171(08), 353–362.

Jakobi, N., Husbands, P., & Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. Lecture Notes in Computer Science, 929, 704–720.

Johnson, W. L., Jindrich, D. L., Roy, R. R., & Reggie Edgerton, V. (2008). A three-dimensional model of the rat hindlimb: Musculoskeletal geometry and muscle moment arms. Journal of Biomechanics, 41(3), 610–619.

Jonas, E., & Kording, K. P. (2017). Could a neuroscientist understand a microprocessor? PLOS Computational Biology, 13(1), 1–24.

Jordan, L. M., Liu, J., Hedlund, P. B., Akay, T., & Pearson, K. G. (2008). Descending command systems for the initiation of locomotion in mammals. Brain Research Reviews, 57(1), 183–191.

Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., & Schaal, S. (2011). Learning, planning, and control for quadruped locomotion over challenging terrain. International Journal of Robotics Research, 30(2), 236–258.

Kashiri, N., Abate, A., Abram, S. J., Albu-Schaffer, A., Clary, P. J., Daley, M., ... others (2018). An overview on principles for energy efficient robot locomotion. Frontiers in Robotics and AI, 5, 129.

Kawasaki, R., Sato, R., Kazama, E., Ming, A., & Shimojo, M. (2016). Development of a flexible coupled spine mechanism for a small quadruped robot. In IEEE International Conference on Robotics and Biomimetics, ROBIO 2016 (pp. 71–76).

Kenneally, G., De, A., & Koditschek, D. E. (2016). Design Principles for a Family of Direct-Drive Legged Robots. IEEE Robotics and Automation Letters, 1(2), 900–907.

Khoramshahi, M., Jalaly Bidgoly, H., Shafiee, S., Asaei, A., Ijspeert, A. J., & Nili Ahmadabadi, M. (2013). Piecewise linear spine for speed-energy efficiency trade-off in quadruped robots. Robotics and Autonomous Systems, 61(12), 1350–1359.

Kim, D., Di Carlo, J., Katz, B., Bledt, G., & Kim, S. (2019). Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. arXiv preprint arXiv:1909.06586.

Kim, S., Laschi, C., & Trimmer, B. (2013). Soft robotics: A bioinspired evolution in robotics. Trends in Biotechnology, 31(5), 287–294.

Kim, Y., Park, J., Yoon, B., Kim, K. S., & Kim, S. (2014). The role of relative spinal motion during feline galloping for speed performance. Journal of Bionic Engineering, 11(4), 517–528.

Kohl, N., & Stone, P. (2004). Policy gradient reinforcement learning for fast quadrupedal locomotion. In IEEE International Conference on Robotics and Automation, ICRA 2004 (Vol. 3, pp. 2619–2624).

Kolter, J. Z. J. Z., Abbeel, P., & Ng, A. Y. (2007). Hierarchical Apprenticeship Learning with Application to Quadruped Locomotion. Advances in Neural Information Processing Systems, 1, 1–8.

Kota, S., & Ananthasuresh, G. (1995). Designing compliant mechanisms. Mechanical Engineering-CIME, 117(11), 93–97.

Larger, L., Soriano, M. C., Brunner, D., Appeltant, L., Gutiérrez, J. M., Pesquera, L., ... Fischer, I. (2012). Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing. Optics Express, 20(3), 3241.

Lazebnik, Y. (2002). Can a biologist fix a radio?—or, what i learned while studying apoptosis. Cancer cell, 2(3), 179–182.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.

LeCun, Y. A., Bottou, L., Orr, G. B., & Müller, K.-R. (2012). Efficient backprop. Lecture Notes in Computer Science, 9–48.

Lee, C., Kim, M., Kim, Y. J., Hong, N., Ryu, S., Kim, H. J., & Kim, S. (2017). Soft robot review. International Journal of Control, Automation and Systems, 15(1), 3–15.

Lee, J. H., Delbruck, T., & Pfeiffer, M. (2016). Training Deep Spiking Neural Networks Using Backpropagation. Frontiers in Neuroscience, 10, 508.

Lee, Y. H., Lee, Y. H., Lee, H., Phan, L. T., Kang, H., Kim, Y. B., & Choi, H. R. (2017). Development of torque controllable leg for running robot, AiDIN-IV. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017 (pp. 4125–4130).

Legg, S., & Hutter, M. (2007). A collection of definitions of intelligence. Frontiers in Artificial Intelligence and Applications, 157, 17–24.

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2001). End-to-End Training of Deep Visuomotor Policies. The Journal of Machine Learning Research, 17(1), 1334–1373.

Lewis, M. A., Fagg, A. H., & Solidum, A. (1992). Genetic Programming Approach to the Construction of a Neural Network Control of a Walking Robot. In IEEE International Conference on Robotics and Automation, ICRA 1992 (pp. 2618–2623).

Li, T., Nakajima, K., Calisti, M., Laschi, C., & Pfeifer, R. (2012). Octopus-inspired sensorimotor control of a multi-arm soft robot. In IEEE International Conference on Mechatronics and Automation, ICMA 2012 (pp. 948–955).

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., … Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Lipson, H., & Pollack, J. P. (2010). Automatic design and manufacture of robotic life forms. The Nature of Life: Classical and Contemporary Perspectives from Philosophy and Science, 406, 260–267.

Lizier, J. T., Pritam, S., & Prokopenko, M. (2011). Information dynamics in small-world Boolean networks. Artificial Life, 17(4), 293–314.

Lukoševičius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. Computer Science Review, 3(3), 127–149.

Lungarella, M., & Sporns, O. (2006). Mapping information flow in sensorimotor networks. PLOS Computational Biology, 2(10), 1301–1312.

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. Neural Networks, 10(9), 1659–1671.

Marblestone, A. H., Wayne, G., & Kording, K. P. (2016). Toward an Integration of Deep Learning and Neuroscience. Frontiers in Computational Neuroscience, 10, 94.

Markman, A. B., & Brendl, C. M. (2014). Constraining Theories of Embodied Cognition. Psychological Science, 16(1), 6–10.

Marreiros, A. C., Daunizeau, J., Kiebel, S. J., & Friston, K. J. (2008). Population dynamics: Variance and the sigmoid activation function. NeuroImage, 42(1), 147–157.

Martin, L., Sándor, B., & Gros, C. (2016). Closed-loop robots driven by short-term synaptic plasticity: Emergent explorative vs. limit-cycle locomotion. Frontiers in Neurorobotics, 10, 1–11.

Martins, L. T., Arend Tatsch, C. A., Maciel, E. H., Gerndt, R., & Da Silva Guerra, R. (2015). A Polyurethane-based Compliant Element for Upgrading Conventional Servos into Series Elastic Actuators. IFAC-PapersOnLine, 48(19), 112–117.

Martius, G., & Lampert, C. H. (2016). Extrapolation and learning equations. arXiv preprint arXiv:1610.02995.

Matsuda, T., & Murata, S. (2006). Stiffness distribution control locomotion of closed link robot with mechanical softness. In IEEE International Conference on Robotics and Automation, ICRA 2006 (pp. 1491–1498).

Matsuoka, K. (1985). Sustained oscillations generated by mutually inhibiting neurons with adaptation. Biological Cybernetics, 52(6), 367–376.

McGeer, T. (1990). Passive walking with knees. In IEEE International Conference on Robotics and Automation, ICRA 1990 (pp. 1640–1645).

McMahon, T. A., & Cheng, G. C. (1990). The mechanics of running: how does stiffness couple with speed? Journal of Biomechanics, 23, 65–78.

Minassian, K., Hofstoetter, U. S., Dzeladini, F., Guertin, P. A., & Ijspeert, A. J. (2017). The Human Central Pattern Generator for

Locomotion: Does It Exist and Contribute to Walking? Neuroscientist, 23(6), 649–663.

Miriyev, A., Stack, K., & Lipson, H. (2017). Soft material for soft actuators. Nature Communications, 8(1), 1–8.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. arXiv preprint arXiv:1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529–533.

Morton, S. M., & Bastian, A. J. (2004). Cerebellar control of balance and locomotion. The Neuroscientist, 10(3), 247–259.

Morton, S. M., & Bastian, A. J. (2006). Cerebellar contributions to locomotor adaptations during splitbelt treadmill walking. Journal of Neuroscience, 26(36), 9107–9116.

Murai, A., & Yamane, K. (2011). A neuromuscular locomotion controller that realizes human-like responses to unexpected disturbances. In IEEE International Conference on Robotics and Automation, ICRA 2011 (pp. 1997–2002).

Murai, A., Yamane, K., & Nakamura, Y. (2010). Effects of nerve signal transmission delay in somatosensory reflex modeling based on inverse dynamics and optimization. In IEEE International Conference on Robotics and Automation, ICRA 2010 (pp. 5076–5083).

Murphy, M. P., Saunders, A., Moreira, C., Rizzi, A. A., & Raibert, M. (2011). The LittleDog robot. International Journal of Robotics Research, 30(2), 145–149.

Müller, V. C., & Hoffmann, M. (2017). What Is Morphological Computation? On How the Body Contributes to Cognition and Control. Artificial Life, 23(1), 1–24.

Nakai, H., Kuniyoshi, Y., Inaba, M., & Inoue, H. (2003). Metamorphic robot made of low melting point alloy. In IEEE/RSJ International

Conference on Intelligent Robots and Systems, IROS 2003 (Vol. 2, pp. 2025–2030).

Nakajima, K. (2020). Physical reservoir computing—an introductory perspective. Japanese Journal of Applied Physics, 59(6), 060501.

Nakajima, K., Hauser, H., Li, T., & Pfeifer, R. (2015). Information processing via physical soft body. Scientific Reports, 5, 1–11.

Nakajima, K., Li, T., Hauser, H., & Pfeifer, R. (2014). Exploiting short-term memory in soft body dynamics as a computational resource. Journal of the Royal Society Interface, 11(100).

Nakamura, T. (2011). Real-time 3-D object tracking using Kinect sensor. In IEEE International Conference on Robotics and Biomimetics, ROBIO 2011 (pp. 784–788).

Narioka, K., Rosendo, A., Spröwitz, A., & Hosoda, K. (2012). Development of a minimalistic pneumatic quadruped robot for fast locomotion. In IEEE International Conference on Robotics and Biomimetics, ROBIO 2012 (pp. 307–311).

Nassour, J., Hénaff, P., Benouezdou, F., & Cheng, G. (2014). Multi-layered multi-pattern CPG for adaptive locomotion of humanoid robots. Biological Cybernetics, 108(3), 291–303.

Omurtag, A., Knight, B. W., & Sirovich, L. (2000). On the simulation of large populations of neurons. Journal of Computational Neuroscience, 8(1), 51–63.

Orlovsky, G. N. (1972). Activity of vestibulospinal neurons during locomotion. Brain Research, 46, 85–98.

Owaki, D., & Ishiguro, A. (2017). A quadruped robot exhibiting spontaneous gait transitions from walking to trotting to galloping. Scientific Reports, 7(1), 1–10.

Owaki, D., Kano, T., Nagasawa, K., Tero, A., & Ishiguro, A. (2012). Simple robot suggests physical interlimb communication is essential for quadruped walking. Journal of The Royal Society Interface, 10(78).

Paine, N., Oh, S., & Sentis, L. (2014). Design and control considerations for high-performance series elastic actuators. IEEE/ASME Transactions on Mechatronics, 19(3), 1080–1091.

Papadopoulos, D., & Buehler, M. (2000). Stable running in a quadruped robot with compliant legs. In IEEE International Conference on Robotics and Automation, ICRA 2000 (pp. 444–449).

Park, H.-W., Wensing, P. M., & Kim, S. (2017). High-speed bounding with the MIT Cheetah 2: Control design and experiments. International Journal of Robotics Research, 36(2), 167–192.

Paugam-Moisy, H., & Bohte, S. (2012). Computing with spiking neuron networks. In Handbook of Natural Computing (pp. 335–376).

Paul, C. (2006). Morphological computation. A basis for the analysis of morphology and control requirements. Robotics and Autonomous Systems, 54(8), 619–630.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11), 559–572.

Pearson, K. (1976). The control of walking. Scientific American, 235(6), 72–87.

Peng, X. B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In IEEE International Conference on Robotics and Automation, ICRA 2018 (pp. 3803–3810).

Peng, X. B., Berseth, G., & van de Panne, M. (2016). Terrain-adaptive locomotion skills using deep reinforcement learning. ACM Transactions on Graphics, 35(4), 1–12.

Peng, X. B., Berseth, G., Yin, K., & Van De Panne, M. (2017). DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. ACM Transactions on Graphics, 36(4), 1–13.

Pfeifer, R., Lungarella, M., & Iida, F. (2007). Self-organization, embodiment, and biologically inspired robotics. Science, 318(5853), 1088–1093.

Poincare, H. (1881). Mémoire sur les courbes définies par une équation différentielle (I). Journal de Mathématiques Pures et Appliquées, 7, 375–422.

Polani, D. (2011). An informational perspective on how the embodiment can relieve cognitive burden. In IEEE Symposium Series on Computational Intelligence (pp. 78–85).

Poli, R., Langdon, W. B., & McPhee, N. F. (2009). A field guide to genetic programming. Genetic Programming and Evolvable Machines, 10(2), 229–230.

Pollack, J. B., & Lipson, H. (2000). Automatic design and manufacture of robotic lifeforms. Nature, 406(6799), 974–978.

Poulakakis, I., Smith, J. A., & Buehler, M. (2005). Modeling and experiments of untethered quadrupedal running with a bounding gait: The scout II robot. International Journal of Robotics Research, 24(4), 239–256.

Pouya, S., Khodabakhsh, M., Spröwitz, A., & Ijspeert, A. J. (2017). Spinal joint compliance and actuation in a simulated bounding quadruped robot. Autonomous Robots, 41(2), 437–452.

Pratt, G. A., & Williamson, M. M. (1995). Series elastic actuators. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2002 (pp. 399–406).

Pratt, J. E., & Krupp, B. T. (2004). Series elastic actuators for legged robots. In Unmanned Ground Vehicle Technology VI (Vol. 5422, pp. 135–144).

Prilutsky, B. I., Klishko, A. N., Weber, D. J., & Lemay, M. A. (2016). Computing Motion Dependent Afferent Activity During Cat Locomotion Using a Forward Dynamics Musculoskeletal Model. In Neuromechanical Modeling of Posture and Locomotion (pp. 273–307).

Pusey, J. L., Duperret, J. M., Haynes, G. C., Knopf, R., & Koditschek, D. E. (2013). Free-standing leaping experiments with a power-autonomous elastic-spined quadruped. In Unmanned systems technology xv (Vol. 8741).

Raibert, M. (1986). Legged robots that balance. The MIT Press.

Raibert, M., Nelson, K. B. G., & Playter, R. (2008). Bigdog, the rough-terrain quadruped robot. IFAC Proceedings Volumes, 41(2), 10822–10825.

Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2018). Evolutionary Algorithms and Reinforcement Learning : A Comparative Case Study for Architecture Search. In ICML 2018 AutoML Workshop.

Richalet, J., Rault, A., Testud, J., & Papon, J. (1978). Model predictive heuristic control. IFAC Automatica, 14(5), 413–428.

Righetti, L., & Ijspeert, A. J. (2006). Programmable central pattern generators: An application to biped locomotion control. In IEEE International Conference on Robotics and Automation, ICRA 2006 (pp. 1585–1590).

Righetti, L., & Ijspeert, A. J. (2008). Pattern generators with sensory feedback for the control of quadruped locomotion. In IEEE International Conference on Robotics and Automation, ICRA 2008 (pp. 819–824).

Rossignol, S., Dubuc, R., & Gossard, J.-P. (2005). Dynamic Sensorimotor Interactions in Locomotion. Physiological Reviews, 86(1), 89–154.

Rückert, E. A., & Neumann, G. (2013). Stochastic optimal control methods for investigating the power of morphological computation. Artificial Life, 19(1), 115–131.

Rusu, A. A., Vecerik, M., Rothörl, T., Heess, N., Pascanu, R., & Hadsell, R. (2016). Sim-to-Real Robot Learning from Pixels with Progressive Nets. arXiv preprint arXiv:1610.04286.

Saab, W., Rone, W. S., & Ben-Tzvi, P. (2017). Robotic Modular Leg: Design, Analysis, and Experimentation. Journal of Mechanisms and Robotics, 9(2).

Schaal, S. (2009). The SL simulation and real-time control software package. University of Southern California, 1–93.

Schemmel, J., Brüderle, D., Grübl, A., Hock, M., Meier, K., & Millner, S. (2010). A wafer-scale neuromorphic hardware system for large-scale neural modeling. In IEEE International Symposium on Circuits and Systems, ISCAS 2010 (pp. 1947–1950).

Schubert, B. E., & Floreano, D. (2013). Variable stiffness material based on rigid low-melting-point-alloy microstructures embedded in soft poly(dimethylsiloxane) (PDMS). RSC Advances, 3(46), 24671–24679.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015). Trust region policy optimization. In International Conference on Machine Learning, ICML 2015 (pp. 1889–1897).

Semini, C., Barasuol, V., Boaventura, T., Frigerio, M., Focchi, M., Caldwell, D. G., & Buchli, J. (2015). Towards versatile legged robots through active impedance control. International Journal of Robotics Research, 34(7), 1003–1020.

Semini, C., Tsagarakis, N. G., Guglielmino, E., Focchi, M., Cannella, F., & Caldwell, D. G. (2011). Design of HyQ -A hydraulically and electrically actuated quadruped robot. Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering, 225(6), 831–849.

Seo, K., Chung, S. J., & Slotine, J. J. E. (2010). CPG-based control of a turtle-like underwater vehicle. Autonomous Robots, 28(3), 247–269.

Seoane, L. F. (2019). Evolutionary aspects of reservoir computing. Proceedings of the Royal Society B: Biological Sciences, 374(1774), 45–48.

Seok, S., Wang, A., Chuah, M. Y., Otten, D. M., Lang, J. H., & Kim, S. (2013). Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot. In IEEE International Conference on Robotics and Automation, ICRA 2013 (pp. 3307–3312).

Seok, S., Wang, A., Otten, D. M., & Kim, S. (2012). Actuator design for high force proprioceptive control in fast legged locomotion. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012 (pp. 1970–1975).

Shannon, C. E. (1949). Communication in the Presence of Noise. Proceedings of the IRE, 37(1), 10–21.

Shen, H., Yosinski, J., Kormushev, P., Caldwell, D. G., & Lipson, H. (2012). Learning fast quadruped robot gaits with the RL PoWER spline parameterization. Cybernetics and Information Technologies, 12(3), 66–75.

Siciliano, B., & Khatib, O. (2017). Springer Handbook of Robotics, 2nd Edition. Springer International Publishing AG.

Sigvardt, K. A., & Williams, T. L. (1992). Models of central pattern generators as oscillators: the lamprey locomotor CPG. Seminars in Neuroscience, 4(1), 37–46.

Singla, A., Bhattacharya, S., Dholakiya, D., Bhatnagar, S., Ghosal, A., Amrutur, B., & Kolathaya, S. (2019). Realizing learned quadruped locomotion behaviors through kinematic motion primitives. In IEEE International Conference on Robotics and Automation, ICRA 2019 (pp. 7434–7440).

Song, S., & Geyer, H. (2015). A neural circuitry that emphasizes spinal feedback generates diverse behaviours of human locomotion. Journal of Physiology, 593(16), 3493–3511.

Sprowitzt, A., Tuleut, A., Khoramshahi, M., Spröwitz, A., Tuleu, A., Ahmadabadi, M. N., & Ijspeert, A. J. (2013). Benefits of an Active Spine Supported Bounding Locomotion With a Small Compliant Quadruped Robot. In IEEE International Conference on Robotics and Automation, ICRA 2013 (pp. 3329–3334).

Spröwitz, A., Tuleu, A., Vespignani, M., Ajallooeian, M., Badri, E., & Ijspeert, A. J. (2013). Towards dynamic trot gait locomotion: Design, control, and experiments with Cheetah-cub, a compliant quadruped robot. International Journal of Robotics Research, 32(8), 932–950.

Spröwitz, A. T., Tuleu, A., Ajallooeian, M., Vespignani, M., Möckel, R., Eckert, P., ... Ijspeert, A. J. (2018). Oncilla robot: a versatile open-source quadruped research robot with compliant pantograph legs. Frontiers in Robotics and AI, 5, 67.

Sreenivasa, M., Ayusawa, K., & Nakamura, Y. (2016). Modeling and Identification of a Realistic Spiking Neural Network and Musculoskeletal Model of the Human Arm, and an Application to the Stretch Reflex. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 24(5), 591–602.

Steeves, J. D., & Jordan, L. M. (1980). Localization of a Descending Pathway in the Spinal Cord which is Necessary for Controlled Treadmill Locomotion. Neuroscience Letters, 20, 283–288.

Steil, J. J., & Reinhart, F. R. (2009). Reaching movement generation with a recurrent neural network based on learning inverse kinematics. In IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009 (pp. 323–330).

Sussillo, D., & Abbott, L. F. (2009). Generating Coherent Patterns of Activity from Chaotic Neural Networks. Neuron, 63(4), 544–557.

Sutton, R. S., & Barto, A. G. (2013). Reinforcement Learning: An Introduction. The MIT Press.

Szita, I., Gyenes, V., & Lörincz, A. (2006). Reinforcement Learning with Echo State Networks. In International Conference on Artificial Neural Networks, ICANN 2006 (pp. 830–839).

Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., ... Corke, P. (2018). The limits and potentials of deep learning for robotics. International Journal of Robotics Research, 37(4-5), 405–420.

Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., ... Vanhoucke, V. (2018). Sim-to-Real: Learning Agile Locomotion For Quadruped Robots. arXiv preprint arXiv:1804.10332.

Tanaka, G., Yamane, T., Héroux, J. B., Nakane, R., Kanazawa, N., Takeda, S., ... Hirose, A. (2019). Recent advances in physical reservoir computing: A review. Neural Networks, 115, 100–123.

Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., & Maida, A. (2019). Deep learning in spiking neural networks. Neural Networks, 111, 47–63.

Thijssen, J. (2007). Computational physics. Cambridge university press.

Tokic, M. (2010). Adaptive ϵ-greedy exploration in reinforcement learning based on value differences. In Annual Conference on Artificial Intelligence (pp. 203–210).

Torricelli, D., Gonzalez-Vargas, J., Veneman, J. F., Mombaur, K., Tsagarakis, N. G., Del-Ama, A. J., ... Pons, J. L. (2015). Benchmarking Bipedal Locomotion: A Unified Scheme for Humanoids, Wearable Robots, and Humans. IEEE Robotics and Automation Magazine, 22(3), 103–115.

Trivedi, D., Rahn, C. D., Kier, W. M., & Walker, I. D. (2008). Soft robotics: Biological inspiration, state of the art, and future research. Applied Bionics and Biomechanics, 5(3), 99–117.

Tsianos, G. A., Rustin, C., & Loeb, G. E. (2012). Mammalian muscle model for predicting force and energetics during physiological behaviors. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 20(2), 117–133.

Tucker, V. A. (1975). The Energetic Cost of Moving About: walking and running are extremely inefficient forms of locomotion. Much greater efficiency is achieved by birds, fish - and bicyclists. American Scientist, 63(4), 413–419.

Tuleu, A., Ajallooeian, M., Spröwitz, A., Löpelmann, P., & Ijspeert, A. J. (2011). Trot Gait Locomotion of a Cat Sized Quadruped Robot. In International Workshop on Bio-Inspired Robots (pp. 1–4).

Valle, G., Mazzoni, A., Iberite, F., D'Anna, E., Strauss, I., Granata, G., ... Micera, S. (2018). Biomimetic Intraneural Sensory Feedback Enhances Sensation Naturalness, Tactile Sensitivity, and Manual Dexterity in a Bidirectional Prosthesis. Neuron, 100(1), 37–45.

Van Essen, D. C., & Felleman, D. J. (1991). Distributed Hierarchical Processing in the Primate Cerebral Cortex. Cerebral Cortex, 1, 1–47.

Van Ham, R., Vanderborght, B., Van Damme, M., Verrelst, B., & Lefeber, D. (2007). MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot. Robotics and Autonomous Systems, 55(10), 761–768.

Vanderborght, B., Tsagarakis, N. G., Semini, C., Ham, R. V., & Caldwell, D. G. (2009). MACCEPA 2.0: Adjustable compliant actuator with stiffening characteristic for energy efficient hopping. In IEEE International Conference on Robotics and Automation, ICRA 2009 (pp. 544–549).

Vanderborght, B., Van Ham, R., Lefeber, D., Sugar, T. G., Hollander, K. W., Ham, R. V., … Hollander, K. W. (2009). Comparison of Mechanical Design and Energy Consumption of Adaptable, Passive-compliant Actuators. International Journal of Robotics Research, 28(1), 90–103.

Vandesompele, A., Urbain, G., Mahmud, H., wyffels, F., & Dambre, J. (2019). Body randomization reduces the sim-to-real gap for compliant quadruped locomotion. Frontiers in Neurorobotics, 13, 1–9.

Vandesompele, A., Urbain, G., wyffels, F., & Dambre, J. (2019). Populations of spiking neurons for reservoir computing: Closed loop control of a compliant quadruped. Cognitive Systems Research, 58, 317–323.

Vandoorne, K., Mechet, P., Van Vaerenbergh, T., Fiers, M., Morthier, G., Verstraeten, D., … Bienstman, P. (2014). Experimental demonstration of reservoir computing on a silicon photonics chip. Nature communications, 5, 3541.

Villarreal, O., Barasuol, V., Camurri, M., Franceschi, L., Focchi, M., Pontil, M., … Semini, C. (2019). Fast and Continuous Foothold Adaptation for Dynamic Locomotion Through CNNs. IEEE Robotics and Automation Letters, 4(2), 2140–2147.

Villarreal, O., Barasuol, V., Wensing, P., & Semini, C. (2019). MPC-based Controller with Terrain Insight for Dynamic Legged Locomotion. arXiv preprint arXiv:1909.13842.

Vu, H. Q., Yu, X., Iida, F., & Pfeifer, R. (2015). Improving energy efficiency of hopping locomotion by using a variable stiffness actuator. IEEE/ASME Transactions on Mechatronics, 21(1), 472–486.

Vukobratovi, M., & Borovac, B. (2004). Zero-Moment Point — Thirty Five Years of Its Life. International Journal of Humanoid Robotics, 1(01), 157–173.

Wang, C., & Wang, S. (2016). Bionic Control of Cheetah Bounding with a Segmented Spine. Applied Bionics and Biomechanics, 2016.

Wang, C., Zhang, T., Wei, X., Long, Y., & Wang, S. (2017). Dynamic characteristics and stability criterion of rotary galloping gait with an articulated passive spine joint. Advanced Robotics, 31(4), 168–183.

Wang, J. M., Hamner, S. R., Delp, S. L., & Koltun, V. (2012). Optimizing locomotion controllers using biologically-based actuators and objectives. ACM Transactions on Graphics, 31(4), 1–11.

Wang, Z., Merel, J. S., Reed, S. E., de Freitas, N., Wayne, G., & Heess, N. (2017). Robust Imitation of Diverse Behaviors. In Advances in Neural Information Processing Systems, NIPS 2017 (pp. 5320–5329).

Wei, X., Long, Y., Wang, C., & Wang, S. (2015). Rotary galloping with a lock-unlock elastic spinal joint. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 229(6), 1088–1102.

Weidel, P., Djurfeldt, M., Duarte, R., & Morrison, A. (2016). Closed loop interactions between spiking neural network and robotic simulators based on MUSIC and ROS. Frontiers in Neuroinformatics, 10(31), 1–19.

Willems, B., Degrave, J., & Dambre, J. (2017). Quadruped Robots Benefit from Compliant Leg Designs. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017.

Wolf, S., & Hirzinger, G. (2008). A new variable stiffness design: Matching requirements of the next robot generation. In IEEE International Conference on Robotics and Automation, ICRA 2008 (pp. 1741–1746).

wyffels, F., D'Haene, M., Waegeman, T., Caluwaerts, K., Nunes, C., & Schrauwen, B. (2010). Realization of a passive compliant robot dog. In IEEE/RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob 2010 (pp. 882–886).

wyffels, F., Li, J., Waegeman, T., Schrauwen, B., & Jaeger, H. (2014). Frequency modulation of large oscillatory neural networks. Biological Cybernetics, 108(2), 145–157.

wyffels, F., & Schrauwen, B. (2009). Design of a central pattern generator using reservoir computing for learning human motion. In Advanced Technologies for Enhanced Quality of Life, AT-EQUAL 2009 (pp. 118–122).

Xi, W., Yesilevskiy, Y., & Remy, D. C. (2016). Selecting gaits for economical locomotion of legged robots. International Journal of Robotics Research, 35(9), 1140–1154.

Xie, Z., Berseth, G., Clary, P., Hurst, J., & Van De Panne, M. (2018). Feedback Control for Cassie with Deep Reinforcement Learning. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018 (pp. 1241–1246).

Yamanaka, Y., Yaguchi, T., Nakajima, K., & Hauser, H. (2018). Mass-spring damper array as a mechanical medium for computation. In Lecture Notes in Computer Science (pp. 781–794).

Yoshida, H. (1990). Construction of higher order symplectic integrators. Physics Letters A, 150(5), 262–268.

Yu, J., & Eidelberg, E. (1983). Recovery of locomotor function in cats after localized cerebellar lesions. Brain Research, 273(1), 121–131.

Zahedi, K., & Ay, N. (2013). Quantifying morphological computation. Entropy, 15(5), 1887–1915.

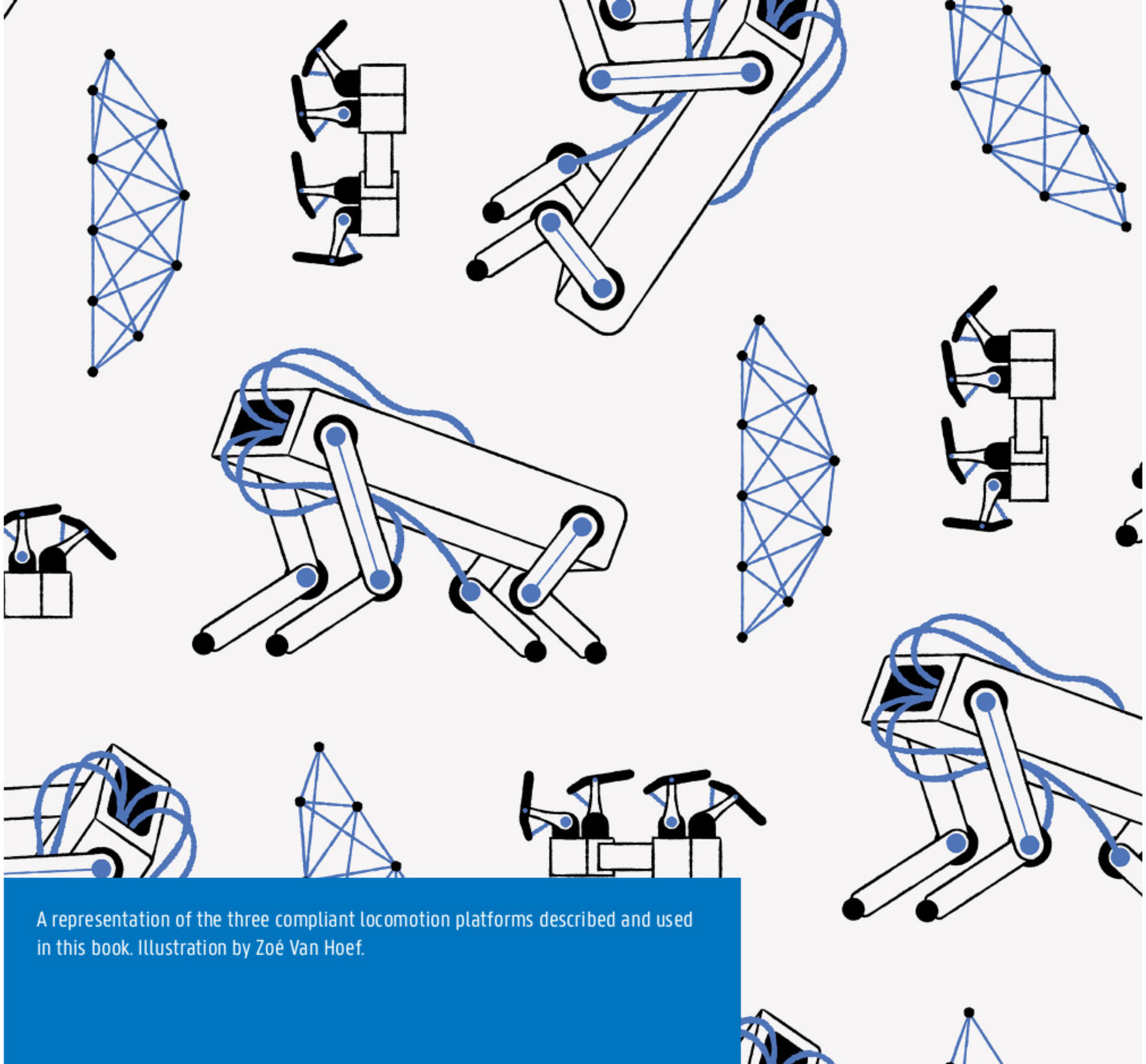
Zhang, M., Geng, X., Bruce, J., Caluwaerts, K., Vespignani, M., Sun-Spiral, V., ... Levine, S. (2017). Deep Reinforcement Learning for Tensegrity Robot Locomotion. In IEEE International Conference on Robotics and Automation, ICRA 2017 (pp. 634–641).

Zhao, Q., Nakajima, K., Sumioka, H., Hauser, H., & Pfeifer, R. (2013). Spine dynamics as a computational resource in spine-driven quadruped locomotion. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2013 (pp. 1445–1451).

Zhong, Y., Wang, R., Feng, H., & Chen, Y. (2019). Analysis and research of quadruped robot's legs: A comprehensive review. International Journal of Advanced Robotic Systems, 16(3), 1–15.

Ziegler, J. G., & Nichols, N. B. (1942). Optimum settings for automatic controllers. Transactions of the American Society of Mechanical Engineers, 64(11).

Ziegler, M., Iida, F., & Pfeifer, R. (2006). Cheap underwater locomotion: roles of morphological properties and behavioural diversity. In International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, CLAWAR 2006.



A representation of the three compliant locomotion platforms described and used in this book. Illustration by Zoé Van Hoef.