Hardware, software and control design considerations towards low-cost compliant quadruped robots

THÈSE N^{O} 7458 (2016)

PRÉSENTÉE LE 23 JUIN 2016

À L'ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE BIOROBOTIQUE

ET

À L'INSTITUTO SUPERIOR TÉCNICO (IST) DA UNIVERSIDADE DE LISBOA

PROGRAMME DOCTORAL EN ROBOTIQUE, CONTRÔLE ET SYSTÈMES INTELLIGENTS
ET
DOUTORAMENTO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES (PhD)

PAR

Alexandre TULEU

acceptée sur proposition du jury:

Prof. H. Bleuler, président du jury
Prof. A. Ijspeert, Prof. A. Bernardino, directeurs de thèse
Dr N. Tsagarakis, rapporteur
Prof. C. Santos, rapporteuse
Prof. J. Paik, rapporteuse

Suisse
2016

The path to wisdom does, in fact, begin with a single step.
Where people go wrong is in ignoring all the thousands of other steps that come after it.
— Terry Pratchett, *Hogfather*

Acknowledgments

First and foremost, I want to express my gratitude to both of my thesis directors, Auke Jan Ijspeert and Alexandre Bernardino, for giving me the opportunity to do the work presented in this thesis at BioRob and VisLab. Without their scientific advice, guidance, but also their continuous support I would not have been able to pursue this work. They always helped me to retrieve my confidence and get back on track when I needed to. I am very grateful to both of them.

I would like to also thank the members of my thesis committee, Prof. Hannes Bleuler, Prof. Jamie Paik, Prof. Nikolaos Tsagarakis and Prof. Cristina Peixoto Santos, for their time, remarks and suggestions, that helped me a lot to improve the presentation of this work.

I would like to extend a special note of gratitude to Alexander Spröwitz, my supervisor at BioRob when I was a master student and latter my colleague at the beginning of my PhD. He is the original designer of the Cheetah-Cub and Oncilla robots and without him, and our many discussions, I would not have been able to develop many of the key ideas of this thesis.

I had the opportunity to work in two great laboratories, and to meet there even greater people. I would like to thank:

Jesse van den Kieboom who provided BioRob with so many genuine and great tools, from cȯdγn to the optimization cluster... He was kind enough to give me advice on how to improve my software development skills

Mostafa Ajallooeian for his many discussion on quadruped controller design, and for all of his silly questions during our many trips around the world together.

Peter Eckert, for his colossal work on the quadruped robots' mechanical designs, and to learn me how to design mechanical parts, and to build them with a CNC or a 3D printer.

Alessandro Crespi, for its kindness and smile, its inexhaustible knowledge on Linux and electronics, and to have taught me how to design and build a PCB.

Massimo Vespignani, also for its kindness, the countless time he spent to build the many Cheetah-Cub and Oncilla prototypes and to perform experiments together on these robots. He is also the most awesome beta tester of super-sized musical keyboard.

Jeremy Knüsel as a great office mate and to have presented me the Go language.

Hamed Razavi, who reminded me to always take a step back when modeling a complex system, and to first try to address more simpler problems and to increase the complexity

step-by-step.

Francois Longchamp, which always has good advice on how to practically build anything, who let me mess - just a little - with his tools, and to have taught me how to use a milling machine.

Furthermore, I would also like to thank all of the other past and current members of BioRob: Sébastien Gay, Soha Pouya, Rico Möckel, Stéphane Bonardi, Konstantinos Karakasiliotis, Andrej Bičanski, Andrej Gams, Yannick Morel, Tadej Petrič, Robin Thandiackal, Nicolas Van der Noot, Tomislav Horvat, Salman Faraji, Jessica Lanini, Simon Hauser, Florin Dzeladini, Luca Colasanto, Kamilo Melo, Behzad Bayat, Amy Wu, Mehmet Mutlu, Shravan Tata Ramalingasetty and last but not least Sylvie Fiaux, for making BioRob such a great place to work and live in.

I would like to thank the people of VisLab, especially Prof. Jose Santos-Victor, Prof. José Gaspar, Ricardo Fereira, Lorenzo Jamone, Matteo Taiana, Giovanni Saponaro, Nino Cauli and Luka Lukić. I feel a bit sorry that I could not have the time to integrate more deeply my research with yours, but I could learn many things in Computer Vision and Machine Learning from you, and you made my time in Lisbon much so much nicer.

I am also grateful to the AMARSi consortium for providing me with the opportunity to work with knowledgeable scientists and engineers from many countries, and more particularly Sebastian Wrede and Arne Nordmann, for helping me to provide a very polished API for the Oncilla Robot. I would also like to thank Harshal Arun Sonar for helping me to test my ideas on tactile sensors.

As an inexhaustible source of moral support, moments of relaxation and animation and as a place to find all sort of new crazy ideas, I would like to thank all the people of the Satellite association. Having such a nice chilling place and nice people just next door of your lab, is such a help during a PhD. In the end, it is sometimes a funny and original project, imagined around a shared beer (building a super-sized two meter by one gamepad) that let you find one of the missing piece of the PhD puzzle.

Finally, I want to thank my family and my closest friends for their support during these years: my sister Gabrielle, who I know that I can always count on her — and I hope that she knows that she can always count on me — and also Danny Lambert and Claire Roulin who helped me a lot to go through the hard moments of writing this thesis, and whose kindness and support helped me a lot to go through other difficult times during these past years.

This work has been supported by the grant SFRH/BD/51451/2011 provided by the Portuguese Fundação para a Ciência e a Tecnologia (FCT).

Lausanne, the first of June 2017 A. T.

Abstract

Quadrupedal robots have been a field of interest the last few years, with many new maturing platforms. Many of these projects have in common the use of state of the art actuation and sensing, and therefore are able to handle difficult locomotion tasks very effectively.

This work focuses on another trend of low-cost, quadrupedal robots, that features less-precise actuators and sensors, but overcomes their limitations with strong bio-inspired designs to achieve state of the art locomotion. We aim here to further extend the achievements of this approach to handle more complex tasks and that require anticipation, We would like also to verify to which extent a close synergy between clever mechanics, sensorimotor coordination, and Central Pattern Generator models is able to handle these tasks.

This thesis presents supporting work that was required to pursue this goal. A software architecture for the development of real-time drivers and low-level control for robotic applications, based on a clear separation of concerns is presented. An implementation of this architecture able to handle the specific requirements for small compliant quadruped robots is proposed. Furthermore, the development and integration of a communication protocol for inter-electronic devices communication on the Oncilla robot is discussed. As leg load is a key quantity in some of the sensory-motor coordination this thesis want to explore, a novel tactile sensing approach for its estimation is proposed, based on an Extended Kalman Filter data fusion of static and dynamic tactile sensor information. Then, to support the design of efficient interactions between the control and the bio-inspired mechanics, accurate dynamic modeling of the Advanced Spring Loaded Pantographic leg, equipping all robots considered here, is presented. We propose two approaches to this modeling with the presentation of their benefits and limitations.

Finally, two Central Pattern Generator architectures are proposed, based on biologically inspired foot trajectories. The first is using a well-known method for inter-limb coordination with strong neural coupling, and the second, the Tegotae rule, relies only on limb physical coupling and strong sensory-motor coordination. These two approaches are compared on their capacity to handle dynamic footstep placement and it let to the conclusion that strong sensory-motor coordination is required for this task.

Key words: Quadrupedal Robots, Biologically-inspired Robots, Central Pattern Generators,

Compliant Joint/Mechanism, Control Architectures and Programming, Force and Tactile Sensing, Sensor Fusion, Optimization and Optimal Control

Résumé

La robotique quadrupède à connu un interêt croissant ces dernières années, avec l'émergence de nombreux nouveaux robots. La plupart de ces projets ont en commun d'utiliser un actionnement et des détections évoluées et au sommet de l'état de l'art. Cela leur permet d'aborder des tâches difficile en locomotion de manière particulièrement efficace.

Ce travail se focalise sur une autre tendance, plus bon marché, de la robotique quadrupède. Ces robots utilisent des actuateurs et des senseurs moins précis, mais contournent leur limitations en se fondant sur une conception fortement inspirée de la biologie afin de réaliser des tâches locomotives au sein de l'état de l'art. Notre bût est ici d'étendre les réalisations de cette approche pour aborder des tâches plus complexe que celles précedemment realisées et qui requièrent une anticipation. Nous sommes également interessé de vérifier à quel point une synergie forte entre une méchanique adéquate, une coordination sensori-motrice et des *Central Pattern Generators* est adaptèe pour résoudre ces tâches.

Cette thèse présente plusieurs travaux préliminaires qui sont requis pour pousuivre ce but. Une architecture logicielle dédiée au developpement de pilotes temps réel et d'applications robotique bas niveau est présentée. Une implementation de cette architecture, capable de répondre aux exigences spécifiques aux petits robots quadrupèdes souples, est proposée. Le developpement et de l'intégration d'un protocole de communication pour l'électronique du robot Oncilla est également discuté. Comme les efforts sur la jambe est une grandeur clef dans la coordination sensori-motrice que nous souhaitons explorer, une nouvelle approche pour son estimation fondée sur des capteurs tactile est proposée. Cette approche repose sur la fusion de données entre capteurs statique et dynamique au travers d'un filtre de Kalman étendu. Enfin, afin d'aider la conception d'interactions adéquates entre le contrôle et la méchanique bio-inspirée, une modelisation précise de la jambe "Advanced Spring Loaded Pantographic" (ASLP), équipant tous les robots étudiés, est présentée. Deux approches sont proposées ainsi que leur bénéfices et limitations.

Finallement, deux modèles de *Central Pattern Generator* sont proposés, fondé sur une trajectoire bio-inspirée du pied. Le première utilise une méthode bien connue pour la coordination entre les jambes avec des couplages neuraux forts. La seconde, appelée règle Tegotae, reposent seulement sur le couplage méchanique entre les jambes du quadrupède, et une coordination sensori-motrice forte. Ces deux approches sont comparées et leur capacité à

être utilisée pour le placement dynamique du pied au cours de la locomotion est étudié. Il en a découlé qu'avec ces approches, une coordination sensori-motrice forte est requise pour cette tâche.

Mots clefs : Robot Quadrupèdes, Robots Bio-inspirés, Central Pattern Generators, Articulations/Mechanisme souples, Architecture de contrôle et programmation, Capteurs tactiles et capteurs de forces, Fusion de d'acquisition de données, Optimisation et contrôle optimal

Resumo

Os robots quadrupedes têm sido foco de interesse nos últimos anos com o desenvolvimento de diversos robots funcionais. Muitos destes robots utilizam sensores a actuadores avançados e dispendiosos para conseguir uma locomoção eficaz em terrenos difíceis. O presente trabalho toma uma abordagem diferente, optando por sensores mais económicos e acessíveis, mas inspirados em sistemas biológicos, com capacidades de antecipação. Este objectivo é atingido através de uma "delegação de controlo" para os sistemas de mais baixo nível: sistemas mecânicos com características adequadas, coordenação sensório-motora e geradores de padrões centralizados. Pretende-se estudar até que ponto estes sistemas executar tarefas complexas ao nível do estado-da-arte.

Esta tese apresenta um conjunto de trabalhos que suporta a visão anterior. Inicialmente apresenta-se uma arquitectura de software baseada no princípio da separação de competências, e um protocolo de comunicações entre dispositivos electrónicos, para o controlo de locomoção de baixo-nivel em tempo real. Esta arquitectura foi desenvolvida tendo em conta os requisitos de robots quadrúpedes de pequenas dimensões. De seguida apresenta-se um novo sensor táctil e uma metodologia de estimação das forças na perna do robot baseada num filtro de Kalman extendido que efectua a fusão das cargas estáticas e dinâmicas. Posteriormente, apresenta-se uma modelação detalhada da dinâmica do sistema de locomoção que equipa todos os robots considerados nesta tese: Advanced Spring Loaded Pantographic leg. Apresenta-se duas abordagens para a modelação, comparando os seus benefícios e limitações. Finalmente, duas arquitecturas de geradores de padrões centralizados são propostas, baseadas em trajectórias do pé biologicamente inspiradas. A primeira utiliza um método conhecido para coordenação entre membros com elevado esforço de sincronização (acoplamento neural), enquanto a segunda, a regra Tegotae, baseia-se apenas na percepção local das forças em cada perna (acoplamento físico) e numa maior coordenação sensório-motora. Estas duas abordagens são comparadas na sua capacidade de gerir o posicionamento dinâmico do pé do robot.

Palavras-chave: Robots Quadrúpedes, Robots Inspirados Biologicamente, Geradores de Padrões Centralizados, Mecanismos/Juntas Complacentes, Arquiteturas de Controlo e Programação, Sensores de Força e Tacto, Optimização e Controlo Óptimo.

Contents

1 Introduction

Quadruped robotics has received an increasing amount of attention the last few years, with many new projects (Big Dog, Wildcat, Spot, SpotMini, HyQ, StarlETH, ANYmal or MIT Cheetah) maturing. The common point of all these projects is that they aim to push forward the performances of legged robots in terms of speed, energy efficiency, agility, robustness to perturbation and interactions with uncertain environment. To reach these goals, these projects have used or developed state-of-the-art solutions in terms of actuation bandwidth and power efficiency, employing extremely precise sensors to overcome the technical challenges they faced. In contrast, animals have access to less precise actuators and sensors, although it hardly seems fair to compare the accuracy of a mammal's vestibular system to the precision of an Inertia Measurement Unit (IMU) capable of guiding an aircraft. Likewise, the depth precision of an animal's stereovision system and the precision of a modern laser range scanner are difficult to equate. Yet animals still show performances that exceed nowadays most advanced legged systems.

Another approach, which projects like Super Mini Cheetah (SMC) and Cheetah-Cub have employed, relies on simple, off-the-shelf components and a strong bioinspiration to complete specific, well-defined tasks (e.g. straight-forward, dynamic trotting on flat ground). The Cheetah-Cub robot demonstrated how a bioinspired leg design, the ASLP, coupled with simple feed-forward patterns, could simplify the task of dynamic locomotion over flat terrain and reach one of the fastest dynamic gait for a quadruped robot (6.9 body length per second) [*Spröwitz et al.*, 2013]. The work of *Ajallooeian* [2015] revealed how a platform using the ASLP leg could perform robust locomotion over unperceived terrain. This thesis focuses on how these *low-cost* platforms can be used for locomotion tasks requiring anticipation and terrain perception, without making use of the approaches employed by high-end platforms. For these less expensive platforms, the ability to solve these problems represents a step towards shifting from interesting research projects to tools that could see real-life applications into domains such as search-and-rescue and the exploration of hazardous environments. Using a low-cost quadruped platform — as opposed to a high-end one — has a number of advantages. These platforms: *a)* require fewer resources to operate as only a single person is required;

b) are less dangerous as they use less power than their human-sized counterparts; *c)* are easier to modify and replicate due to lower manufacturing cost; and *d)* can almost be considered disposable which means they can be deployed in larger number. These advantages make them well-suited for education purposes.

The present thesis aims to look at four main topics in relation with the challenge of producing and controlling low-cost and lightweight quadruped robots: a) the modeling of the ASLP leg's mechanical behavior; b) the design of low-level controllers able to perform dynamic footstep placement; c) the conception of a novel tactile-sensing approach to estimate individual leg load; and d) the design and implementation of real-time software that meets the requirement of small quadruped robotic platforms. Before exploring in depth the problematic and outline of this thesis, a more detailed look at the context of legged robotics and bioinspired quadrupeds is due.

1.1 State of the Art: From Walking Machines to Low-Cost Bioinspired Quadruped Robots

1.1.1 Emergence of high-end compliant and dynamic quadruped platforms

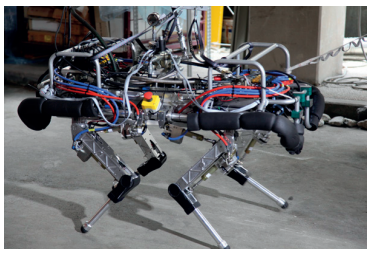
Legged locomotion is not a simple mechanical process. At each step there are impacts with the ground, and this has several implications at various levels: modeling, control and mechanical design. Regarding these first two domains, continuous dynamical systems have only a limited ability to precisely explain the behavior of legged system, and hybrid dynamical systems [*Goebel et al.*, 2009] should be considered. This theory is complex, quite novel and no general results are yet available. At the mechanical level, traditional electric actuation with a gearbox reduction is hard to implement due to the constant presence of impacts throughout the entire gait cycle. This problem is even more difficult when the goal of the legged system is to achieve outstanding performance in terms of dynamic gait, speed and agility, with jumps for example. To overcome these difficulties, in the paste decade, new robotics platforms have tended to use different strategies in terms of actuation and control to move from stiff, kinematically controlled early walking machines, to compliant, dynamic walkers.

1.1.1.1 Hydraulically powered robots

One of the major challenges in achieving versatility in locomotion is simultaneously providing both dynamic locomotion and precise movements. Furthermore, one of the main outcomes of the Defense Advanced Research Projects Agency (DARPA) learning locomotion challenge, was that inverse dynamic model-based control could help to reduce position error feedback gains, and was found very effective for providing precise placements while being compliant in order to compensate for imprecise terrain estimations. This approach improved robust locomotion over challenging terrain [*Buchli et al.*, 2011; *Kalakrishnan et al.*, 2011]. Using traditional electrical high-gear-ratio actuation, this approach is often only accurate at low

(a) Big Dog

(b) Spot



(c) HyQ

Figure 1.1 – Hydraulically powered quadruped robots.

speeds, as these systems tend to be unable to cancel inertia effects due to practical bandwidth limitations. Regardless of using inverse dynamic models, researchers have explored using hydraulic actuation to design platforms capable to have precise and powerful actuation. According to *Boaventura et al.* [2012], hydraulic drives have a substantially higher power-to-weight ratio than comparable electric drives. They are also stiffer, enabling a higher closed-loop control bandwidth, greater accuracy and a better frequency response. Their conception is mechanically simple, which allows for robust design against impact and overload, which are inevitable in highly dynamic locomotion tasks.

Big Dog [*Playter et al.*, 2006; *Raibert*, 2008], developed during the mid-2000s, is probably one of the best-known hydraulic quadruped platforms. Developed by Boston Dynamics, it is the successor of the MIT Leg Laboratory's quadruped. Moreover, Boston Dynamics recently presented two new quadruped robots: Spot [*Dillet*, 2016] and SpotMini [*Ackerman*, 2016][1]. Big Dog weighs 104 kg and is 1 m tall, 1.1 m long and 0.3 m wide [*Raibert*, 2008]. It is fully hydraulically actuated by a water-cooled two-stroke engine of 17 Hp. Each leg has 4 Degrees of Freedom (DoF), with active hip abduction/adduction, hip protraction/retraction, knee flexion/extension and a passively compliant foot. Each actuator is equipped with precise position and force sensors, as well as aerospace-quality servo-valves, and each foot features distal force sensors. It is also equipped with an aerospace-grade IMU, and depending on the version, with a stereovision camera or a laser scanner for exteroreceptive sensing of the environment.

HyQ [*Semini et al.*, 2011; *Boaventura et al.*, 2012] is a partially hydraulic quadruped developed by the Istituto Italiano di Tecnologia (IIT). While it is lighter than Big Dog, weighing 65 kg, it has a comparable size of 0.98 m tall, 1.0 m long and 0.5 m wide. It features 3 Degree of Freedom (DoF) per leg, but the hip abduction/adduction joint is electrically actuated by a Brushless Direct Current (BLDC) motor with harmonic drives [*Boaventura et al.*, 2012]. Each joint is equipped with relative and absolute position encoders and individual load sensing.

Both of these platforms have demonstrated state-of-the-art robust dynamic locomotion in both indoor and outdoor settings [*Wooden et al.*, 2010; *Bazeille et al.*, 2013; *Barasuol et al.*,

[1] As a private corporation Boston Dynamics publishes little to no academic publications on its work.

2013]. However, they use different control approaches.

Big Dog's controller is built on top of the classical Raibert controller for hopping monopod, bipod and quadruped robot [*Raibert et al.*, 1986; *Raibert*, 1990]. For monopods and bipods, Raibert has proven that by decoupling the equations in simple tasks, the control could become extremely simple to solve. The above-mentioned tasks consist of:

- Maintaining the body altitude by controlling the leg thrust power during the stance phase
- Stabilizing body orientation during the stance phase using the hip torque
- Controlling the foot position during the swing phase to control the acceleration of the main body during the *next* stance phase.

This control mechanism, which is easily applicable for the hopping monopods and bipeds, can be extended to quadrupeds via the concept of the virtual leg control [*Raibert and Tello*, 1986]. By ensuring that the orientation of the trunk is remains parallel to the support surface, and by maintaining an exact symmetry between diagonally opposite legs, the quadrupedal case becomes mathematically similar to the bipedal case. However, this approach is limited to symmetric trot, and there are no academic publication explaining how this approach could be extended to outdoor environment. HyQ is using another control framework for both blind and visually guided rough terrain locomotion [*Barasuol et al.*, 2013]. One of the *Barasuol et al.'s* [2013] key ideas is to use a novel reference frame to generate foot trajectories. This "horizontal frame" is placed at the geometric center of the robot, but its horizontal plane is defined as parallel to the ground. Its purpose is to replace the robot reference frame, so as to decouple the foot trajectories from the trunk orientation. In detail, this approach consist to:

- Use a Central Pattern Generator (CPG) to create the foot trajectory in the aforementioned frame. These trajectories take into account the robot's state (height) and desired speed.
- The foot trajectories are kinematically adjusted from the horizontal frame to the robot trunk frame to avoid weak contact with the ground.
- *Buchli et al.'s* [2011] low feedback Proportional–Derivative (PD) controller with and inverse dynamics model is used to generate the desired torque and position joint profile from the inverse kinematics of the desired foot trajectories.
- The desired joint torques are also modulated to correct the trunk posture, again using the robot's inverse dynamic model.
- The foot trajectories are modulated to dynamically alter the desired footstep positions to track the instantaneous capture point. *Pratt et al.* [2006] has demonstrated the utility of this method in recovering from unexpected thrusts of the robot. The required trunk velocities are obtained through state estimation.

By using a stereovision system and modulating the desired step height for each foot, HyQ was able to locomote over perceived terrain [*Havoutis et al.*, 2013; *Bazeille et al.*, 2013].

(a) StarlETH (b) ANYmal

Figure 1.2 – StarlETH and ANYmal, Series-Elastic Actuator (SEA) based quadrupeds.

1.1.1.2 Series-Elastic Actuator- (SEA-) based platforms

Another method to circumventing the problems caused by impacts and shock, is the use of Series-Elastic Actuators (SEAs) [*Pratt and Williamson*, 1995]. This method maintains a high torque density and a precise torque control. Two quadruped robots, developed by Eidgenössische Technische Hochschule Zürich's (ETHZ) autonomous Systems Lab, that use this actuation paradigm are StarlETH [*Remy et al.*, 2012] and its successor ANYmal [*Hutter et al.*, 2016]. StarlETH aims to be fast, versatile and efficient. It weighs 23 kg and is 0.58 m tall, 0.71 m long and 0.64 m wide. Each leg features 3 DoF, all of them actuated by SEAs made of a 200 W Maxon 4-Pole BLDC motor, a 1:100 harmonic reduction and a linear precompressed spring. All of the SEAs are mounted proximally, and power is transmitted to the distal joints using chain and cable pulley systems, which keeps the leg inertia as low as possible. This setup aims to ensure robustness against impacts, permits energy storage to improve efficiency, and achieves full torque controllability [*Remy et al.*, 2012].

The StarlETH control framework operates as follow *Gehring et al.* [2013]:

- A *gait-pattern* module[2] generates the timing for each leg at the scale of the robot. For each legs, it determines the point in time within the gait cycle this leg should be.
- During the swing phase, an appropriate desired footstep position is computed for each leg using the desired robot speed and a predictive linear inverted pendulum model. The legs are then position controlled to track a desired trajectory to reach those points.
- In the stance phase, the desired torques for the leg are computed using Virtual Model Control (VMC) [*Pratt et al.*, 2001]. This approach aims to maintain a desired orientation and acceleration for the robot's trunk.
- A module maintains an estimation of the robot's state, especially the robot's height. These estimations are required to generate the swing-leg trajectories.

[2] In this work, we would have named this module a Central Pattern Generator (CPG) without sensorimotor coordination, as we will define later.

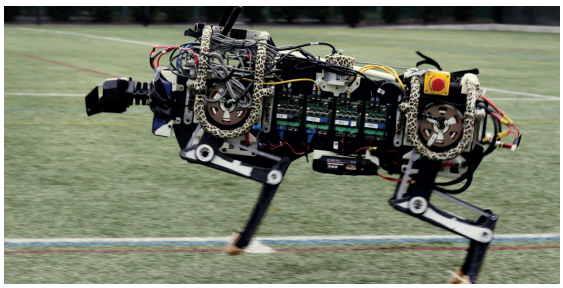


Figure 1.3 – MIT Cheetah

1.1.1.3 High torque-density electric drive platforms

Another recent quadruped platform that appeared recently that aims to push forward the limits of speed and efficiency is the MIT Cheetah (see figure 1.3) which is developed by the MIT Biomimetics Laboratory [*Seok, Wang, Chuah, Hyun, Lee, Otten, Lang and Kim*, 2014]. Its design relies on four principles:

- The use of high torque-density electric drives. These custom-designed motor are based on a large band gap design, which reduces the required stator electrical current associated with high torque applications, and therefore reduces Joule losses in the stator.
- Implementation of energy regeneration to increase efficiency. These motors use a dedicated electronic to recover the braking energy at the beginning of each stance phase.
- The use a low-impedance mechanical transmission (i.e. a custom designed single stage planetary gear) to reduce the transmission losses and the load inertia reflection.
- The a dual coaxial motor design allows for a low-inertia leg, concentrating actuator mass around the trunk.

On the control side, the MIT's approach is simpler than those employed by the previously mentioned projects:

- A gait-pattern generator computes the foot trajectories for each leg with the desired phase differences. Those trajectories use the desired robot speed to determine the duration of the swing and stance phases for each leg.
- The desired leg trajectories serve as the equilibrium point for the leg impedance controller. This controller makes the leg's mechanical behavior mimics as a spring and damper system positioned between the hip and the foot.

This control framework has many open parameters. For example, the foot trajectory and the impedance controller gains, were hand-tuned by *Seok, Wang, Chuah, Hyun, Lee, Otten, Lang and Kim* [2014]. The MIT Cheetah demonstrated an outstanding performance in terms of speed, $6\,\mathrm{m\,s^{-1}}$ while maintaining a low — for a legged robot — Cost of Transport (CoT) of 0.5.

1.1.2 Similarities and differences with animal legged locomotion

All of these platforms claim to be bioinspired to a certain extend. All of these mobile robots solve the problem of ground locomotion via limbs, which is a more bioinspired solution than wheels or tracks. However, whether they use the same principles as animals is more subject to debate. The notion of bioinspiration is complex and is often confused with biomimetics. One appropriate example of bioinspired engineering is the airplanes. Humans do not travel across the world on the back of giant, metallic birds flapping their wings. Instead, understanding some of the general principles of bird flight (e.g. the lift force generated by the wing shape and the relative wind speed) made possible the design of flying machines. However, to generate thrust power, airplanes employs solutions that do not exist in nature, as they are better suited to the problem that humans want to solve with planes, i.e. the fast transport of large masses over long distances.

1.1.2.1 The performance gap between robotics and animals

Regarding bioinspired locomotion, animals continue to outperform their bionic counterparts. For example, the fastest legged are Boston Dynamics Cheetah ($48\,\mathrm{km\,h^{-1}}$) and the MIT cheetah ($21.6\,\mathrm{km\,h^{-1}}$). This is still at best only half of what real cheetahs could achieve, i.e. 110-$120\,\mathrm{km\,h^{-1}}$. Yet animals would seem to posses less efficient components than their robotic counterparts. For example, high-end platforms use quality, even aerospace-grade, IMUs to measure their orientation relative to the Earth's acceleration. Animals also have the vestibular apparatus that would fulfill the same role, although it is significantly less precise. Modern IMUs have a very fast response speed and high precision: a resolution of $0.008\,^\circ\mathrm{s^{-1}}$ and a frequency response up to 250 kHz for a middle-end Microstrain 3DM-GX-4-25 IMU. In comparison, the vestibular system in macaque monkeys (*Macaca Fascicularis*) has a maximal frequency response of 20 Hz and a neural detection threshold of 3-$4\,^\circ\mathrm{s^{-1}}$ [*Sadeghi et al.*, 2007]. Furthermore, robots potentially have access to very reliable and stable actuators, whereas animals rely on muscles, which are subject to fatigue and cannot work indefinitely with the same efficiency and capabilities [*Bigland-Ritchie and Woods*, 1984]. Finally, on the level of control, animals have intrinsic limitations which raise doubts concerning the applicability of implementing the control algorithm described by *Barasuol et al.* [2013] or *Gehring et al.* [2013] as part of a neural system. Both of these approaches rely on an inverse dynamic model for the entire robot. Their algorithm requires a substantial amount of information on the entire system: all current joint positions and velocities, orientation of the main body, estimation of the Ground Reaction Forces (GRFs), among others. To properly function in practice, such approaches demands extensive parameter identification for their models, and are also sensitive to experimental noise. It was recently reported that for such models, rapid integration time was critical to their success [*Johnson et al.*, 2015]. For example, *Herzog et al.* [2016] reported that hierarchical inverse dynamic control could be used in order to implement a fast control loop for the entire robot at a rate of 1 kHz, thus efficiently dealing with the experimental noise of the sensors and model inaccuracies of their platform. However, in

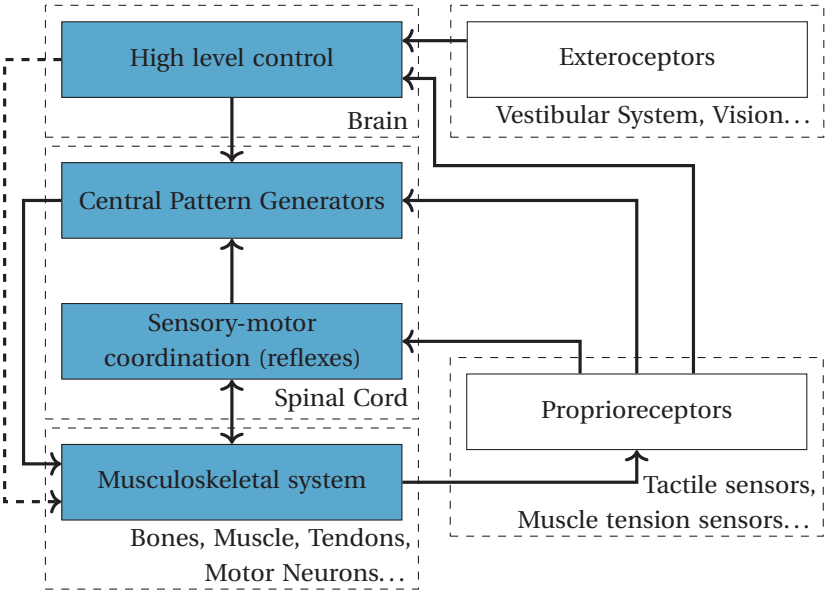


Figure 1.4 – Analysis of the architecture in vertebrate of the locomotor system, inspired from *Rossignol et al.* [2006]. The four principal components of animal locomotion are highlighted in blue.

the case of animals, nerve conductivity would be a limiting factor if a neural circuit were to implement a similar model. For the fastest nerves in horses or human, neural information is bounded to a speed of approximately $60\,\mathrm{m\,s^{-1}}$ [*Jones et al.*, 1982; *Zarucco et al.*, 2010]. This means that for a human, sensory information could take almost 30 ms to travel from the foot to the brain. Any dynamical model implemented in animals would require, to employ another strategy than fast integration time for dealing with their sensor noise and model inaccuracies. The latter are particularly prone to emerge in animals due to rapid morphological changes such as muscle fatigue.

1.1.2.2 Architecture of the vertebrate locomotor system.

If animals still outperform legged robots with, hypothetically, a more complex problem to solve, the question of which principles and paradigms of animal locomotion would be useful for engineering legged machines remains relevant. On the basis of *Rossignol et al.* [2006], locomotion in animals could be interpreted as consisting of four key components (see figure 1.4).

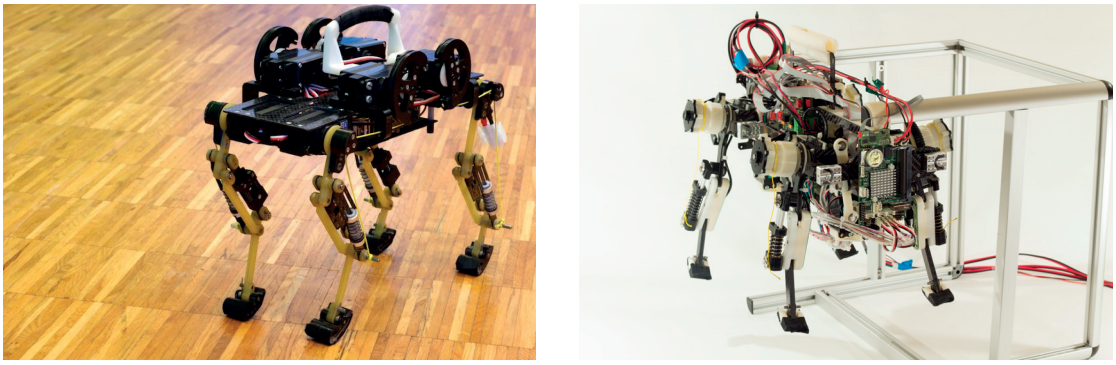
The musculoskeletal system not only provides the physical ability to locomote, but also implements clever mechanisms that help to directly solve some locomotion tasks on their owns. In guinea fowls, *Daley et al.* [2009] demonstrated that the musculoskeletal system alone has self-stabilization properties. They compelled these bipedal birds to step into a hidden pothole, and have identified that stabilization for the first step was only led by feed-forward control and the dynamics of the leg. Furthermore, one of the most common models explaining

the behavior of a leg, namely the Spring-Loaded Inverted Pendulum (SLIP) model [*Geyer et al.*, 2006], shows similar properties in some specific situations. This model could explain for bipedals the basics of walking and running, by considering a simple spring with punctual mass acting as an inverted pendulum. It has also been extended to explain some quadruped behaviors [*Shahbazi and Lopes*, 2016]. In the case of running, the leg retraction criterion [*Seyfarth et al.*, 2003], a non-zero horizontal speed of the foot relative to the ground at touch-down, induces some self-stabilization properties. When this criterion is fulfilled, the model yields a basin of attraction towards its limit cycle. To summarize, the mechanical system's intrinsic design and properties plays a significant role in improving and simplifying certain aspects of the locomotion.

Sensory-motor coordination through spinal reflexes is also a key element of the vertebrate locomotion. Positive force feedback in muscles plays an important role in load-bearing [*Prochazka et al.*, 1997], and it can also generate compliant behavior in a leg [*Geyer et al.*, 2003]. *Geyer and Herr* [2010] went further, proposing a model composed solely of a musculoskeletal structure and a set of spinal reflexes. This model is able to stabilize in walking patterns, which highlight the important role of reflexes in locomotion. Furthermore *Ajallooeian* [2015] found that the implementation of stumbling reflexes on a quadruped robot would help it locomote over rough terrain.

Central Pattern Generators (CPGs) are neural networks capable of producing coordinated patterns of rhythmic activity without any rhythmic inputs from sensory feedback or from higher control centers [*Ijspeert*, 2008]. Their existence was controversial throughout the beginning of the twentieth century, but as *Ijspeert* [2008] mentioned, there is now clear evidence that in vertebrates, rhythms can be centrally generated without the requirement of sensory information. However sensory information does play an important role in modulating the CPGs outputs to coordinate them with the mechanical behavior of the body. *Rossignol et al.* [2006] presents a more in-depth review of these interactions between sensorimotor information and CPGs in the field of neurobiology. Computational CPG models have largely been used in robotics to solve the problem of locomotion for various types of robots, including fish-like swimming [*Zhao et al.*, 2006], anguiliform swimming [*Crespi et al.*, 2005], amphibious locomotion [*Ijspeert et al.*, 2007], bipedal locomotion [*Liu et al.*, 2008], quadrupedal locomotion [*Rutishauser et al.*, 2008; *Righetti and Ijspeert*, 2008; *Spröwitz et al.*, 2013], and multi legged locomotion [*Spröwitz et al.*, 2008].

In addition, CPG computational models have been developed either at a very fine level, using actual model of individual or group of neurons with excitatory and inhibitory coupling, or at more abstract level, using dynamical systems, and more particularly networks of coupled oscillators. This thesis employs the latter approach. These more abstract CPG models are well-suited to perform interlimb coordination, as they make it very simple to impose phase differences between the different oscillators [*Ijspeert*, 2008]. Moreover, *Ajallooeian et al.*



(a) Cheetah-Cub (b) Oncilla

Figure 1.5 – Cheetah-Cub and Oncilla robots

[2013] proposed a mathematical framework for building well-behaving dynamical systems for almost any desired limit cycle. A recent interesting results in CPGs applied to robotics, and their hypothetical role in biology on interlimb coupling would be the work of *Owaki et al.* [2012, 2013]. By using strong sensorimotor coupling between each independent leg and each single oscillators driving them, and by removing any neural coupling (i.e. synchronization) between these oscillators, they could produce a variety of different gaits — lateral or diagonally sequenced walk, trot, pace. Moreover, they linked the emergence of a particular gait to the robot's morphology. These relationships between gaits and morphologies were similar to the correspondences between morphologies and preferred gait patterns for various species. Finally, it has recently been shown on an anguiliform swimming robot that the use of similar strong local coupling produced efficient swimming patterns, even if in the absence of feedback, the underlying CPG was set to produce inefficient open-loop patterns [*Knüsel*, 2013].

High-Level brain control is the last element that modulates the CPGs, on the basis of higher-level signals, such as information from the visual or the vestibular system. In this context, we are interested not only in the conscious part of the brain that makes decisions but also in the role of the *cerebellum*, which is the center of balance control and coordination. For example, studies have revealed that optic flow, the apparent motion of objects, surfaces and edges, plays an important role in regulating the locomotion [*Gibson*, 1958, 1979; *Warren Jr.*, 1998]. As regards this research project, we do not claim that no possibility exists for the brain and the cerebellum to directly control the musculoskeletal system, bypassing the CPGs. However as mentioned earlier, many studies have highlighted the important roles that the three lower-level elements, the musculoskeletal system, the sensorimotor coordination and the CPGs play in locomotion. This thesis limits itself to exploring the interactions between these three components, placing less emphasis on a global integrated control. In the robotic field, this means avoiding global, very rapid and complex inverse dynamics model-based algorithms that maintain balance of the robot and control each joints directly. Instead we prefer a more modular architecture in which some lower-level, simpler controller controls the robot's joints,

and body wide balancing algorithms are held at an higher level and modulate, at a much slower integration rate, this lower-level controller.

1.1.3 Bioinspired low-cost quadruped robot: Cheetah-Cub et al.

The primary robotic platforms studied in this thesis is the family of Cheetah-Cub and Oncilla robots [*Spröwitz et al.*, 2013, 2011]. These are lightweight, compliant, bioinspired quadrupeds. Regarding the mechanics, their bioinspiration comes from the design of their legs, the Advanced Spring Loaded Pantograph (ASLP) leg. *Witte et al.'s* [2000] observations had a significant influence on this design, in particular the suggestion of the pantograph leg template. This template is based on findings regarding mammal leg kinematics during cycle locomotion: their legs are three-segmented, and the proximal and distal segments keep their relative orientation constant during most of the locomotion cycle. This fixed relationship only deviates during the end of the stance phase, near the toe-off event. The ASLP leg uses a pantograph mechanism to achieve this relationship between these two segments (see figure 1.6). This pantograph is passively extended by a diagonal spring, and a cable-pulley mechanism is actively performs the knee flexion. This design is at the heart of *Spröwitz et al.'s* [2013] control strategy. At a high-momentum gait, the leg length is mainly determined by momentum dynamics, as the cable is loosened and the knee actuator is mechanically isolated from the knee. During the swing phase, the cable mechanism is in tension and serves to maintain sufficient foot ground clearance. At low-momentum gaits however, the cable remains tensioned during the stance phase, and the leg continues to be primarily position controlled. *Spröwitz et al.* [2013] indicated that, for highly dynamic gaits, the active control of the leg length could be significantly simplified during the stance phase. Finally serial compliance is present in the ASLP leg by opening one of the sides of the pantograph and by the addition of a small foot passively actuated with a rotational spring. *Spröwitz et al.* [2013] demonstrated that this serial compliance enhanced both the speed and the CoT of the robot.

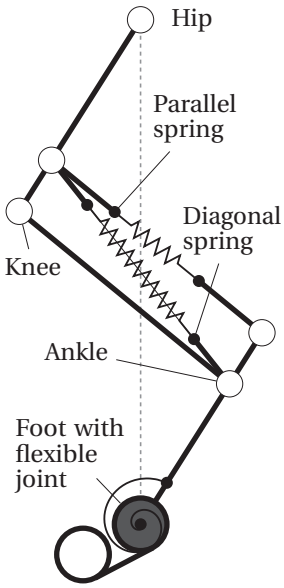


Figure 1.6 – Advanced Spring Loaded Pantograph (ASLP) leg mechanical design, from *Spröwitz et al.* [2013]

Cheetah-Cub (figure 1.5a) is a small robot, weighing 1.1 kg and measuring 15.8 cm tall, 20.1 cm long and 10 cm wide. All its components are off-the-shelf parts. It is actuated by 8 Kondo KRS2350 ICS servomotors driven by a Roboard RB-110 embedded computer. The version presented in *Spröwitz et al.* [2013] has no sensors. Oncilla (figure 1.5b) was designed by the Adaptive Modular Architecture for Rich Motor Skills (AMARSi) FP7 European project [*Soltoggio and Steil*, 2012] as a fully sensorized and more powerful version of Cheetah-Cub. It features the same ASLP leg, but with the addition of a hip abduction/adduction movement for each leg,

for a total of 3 DoF per leg. For the hip protraction/retraction and the knee flexion/extension joint the servomotors were replaced by 90 W BLDC Maxon motors with a custom-designed electronic systems capable of position control with virtual compliance. All of the active joints are relatively position-sensed at the actuator side by optic encoders, and absolutely position-sensed directly on the leg by custom-made magnetic encoders. Each leg features a 3 axis force sensors, available in two forms: one with a distally mounted semi-spherical OptoForce sensor and one with proximally mounted custom designed load cells. Some versions of the Oncilla robot also features a Microstrain 3DM-GX3-35 IMU. This robot also features its own power management electronics that making capable of running autonomously on a single 3S 4500 mA h Li-Po battery for more than 30 minutes with a single charge.

On the control side, in *Spröwitz et al.* [2013], the CPG model had a feed-forward form, without any sensorimotor coordination, in order to investigate almost exclusively the effectiveness of the mechanical behavior of the robot. Furthermore Cheetah-Cub displayed some self-stabilization properties. With a success rate of more than 80%, it could effectively blindly go over a down-step of 2 cm ($\approx$ 12% of its hip height), without any adjustment of its joint trajectories. Furthermore, it could reach a maximal speed of 1.42 m s^{-1} or 6.9 body lengths per second. Thus, in terms of speed relative to its size, Cheetah-Cub is comparable to MIT Cheetah which achieved approximately 6 body lengths per second *Seok, Wang, Chuah, Hyun, Lee, Otten, Lang and Kim* [2014]. These results, which are somewhat comparable to those for high-end platform, with the use of low-cost, off-the-shelf components, were part of the motivation that grew up over the years for this thesis to focus exclusively on these small, low-cost, robots. A motivation that was recently followed by the designer of the MIT Cheetah with the publication of their SMC robot [*Bosworth et al.*, 2016]. Low-cost approaches to quadruped locomotion have several advantages over high-end ones:

- They are inherently easier to modify and more appropriate to test new idea. *Bosworth et al.* [2016] claimed that for high dynamic gaits and unpredictable environment, since there is a bigger gap between simulation and real robot due to hard-to-model impacts and friction, there is a requirement to test hypothesis on real robots. Less expensive robots mean getting to run more tests, and the empirical knowledge acquired in that manner could potentially reduce this gap between reality and simulation.

- Since these robots are much smaller, they require less mechanical power, even while performing highly dynamic gaits. As a consequence, misoperations would be potentially less harmful for the operator. The financial risk in case of damage to the robot is also drastically reduced.

- They require fewer human resources to operate. As size, power and cost increase, robot requires more people to operate them safely and reduce the associated risks. Cheetah-Cub and Oncilla are usually operated by a single person.

- When robots are inexpensive they are also more disposable, as well as prone to use in large numbers. This could be very advantageous for search-and-rescue missions, where the goal is to explore a given geographic zone as quickly as possible to find one or several

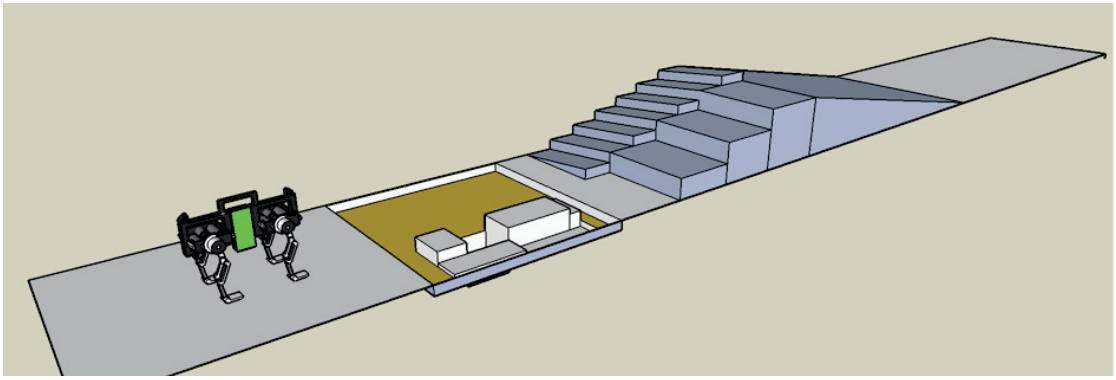


Figure 1.7 – A complex locomotion scenario requiring anticipation. This thesis addresses those scenarios with a bioinspired locomotion architecture similar to that depicted in figure 1.4

people in danger.

As a consequence, low-cost legged robots are better suited to teaching as it is far more easier to allow student train, make mistakes, and test their new ideas. Likewise research unit are able to easily employ several of these robots to perform work in parallel, as is the case at the Biorobotics Laboratory.

1.2 Problem Statements

This thesis focuses on a variety of aspects related to bioinspired quadruped robots, including the design of simulated models, software, hardware and controls. As such, it supported a number of other research projects that have been pursed by others, such as the exploration of Cheetah-Cub's performances and behavior [*Spröwitz et al.*, 2013; *Spröwitz, Ajallooeian, Tuleu and Ijspeert*, 2014], an investigation into achieving gait stabilization via optical flow coordination [*Gay*, 2014], achieving blind rough terrain locomotion [*Ajallooeian*, 2015] and ongoing studies on agility and morphology [*Eckert et al.*, 2015; *Heim et al.*, 2015; *Weinmeister et al.*, 2015; *Eckert and Ijspeert*, 2016]. In addition to these very specific development topics, this thesis also focuses on a more long-term and high-level research question. It aims to address locomotion problems that require anticipation (i.e. perceived rough terrain locomotion; see figure 1.7) within the context of a bioinspired control architecture, as previously described.

Specifically, this addresses the following questions:

To which extent, can the combination of CPG, sensorimotor coordination and bioinspired mechanics achieve dynamic footstep placement? This question addresses a sub-problem related to complex locomotion scenarios. For example, should an obstacle be identified by the high-level controller, how to ensure the feet will be placed correctly in order to avoid this obstacle. This raises the question as to whether modulating the CPG alone, would be sufficient for dynamically adjusting the step position, or whether it requires a regulation in the

form of a feedback of the higher level controller. We try here to investigate if the approaches using complex inverse dynamic model, which are already successful to address this task, are absolutely required, or if simpler, more robust, model free approaches could also be used. If we were to extend the approach of *Spröwitz et al.* [2013], would an open-loop approach be adequate, or would more involved sensorimotor coordination be required, and if so, of what kind? Once identified, what are the kind of sensors required to acquire this information?

How can we accurately model the ASLP leg? As discussed above, we are interested in potential interactions between the mechanical structure of the robot and the low-level control. Such interactions are complex and currently not well understood. It could be technically difficult to gather the data required to explain these interactions and mechanical behaviors, or even to repeat an experiment in the exact same conditions. A simulated model would be particularly helpful for investigating these interactions. Furthermore, as stated by *Spröwitz et al.* [2013], a critical aspect of the ASLP leg design is the tuning of the stiffness of its springs. This aspect is criticized by other researchers [*Bosworth et al.*, 2016], as it could lead to a large number of trials and design iterations to empirically find the correct value. The ability of predicting accurately the mechanical behavior of the ASLP leg would help us reduce this number of iterations when designing any new robots, to adapt those values to the new robot's weight and desired optimal gait characteristics. This leads to other sub-questions, such as which ASLP leg characteristics are the most important to model, and what trade-offs could lead to an efficient model in terms of power computation.

How can we develop generic software capable of addressing the specificity of a variety of robots? The Cheetah-Cub and Oncilla came with their own specificity in terms of control and hardware. But, as one of the main motivations to design low-cost robot is the possibility to iterate rapidly new design or prototypes, many robots with different morphologies were built alongside this thesis. Therefore, the need for a generic control framework, that would solve generic problems of the real-time control of small compliant robot, without requiring a complete new controller for each new robot, was soon made evident. As for highly dynamic gait, the choice of a smaller size and weight imposed even tighter timing constraints, new technical achievements in terms of electronic real-time communications over standard serial protocol were required; therefore these developments will also be detailed.

1.3 Thesis Outline

This thesis is organized as follows:

Chapter 2 describes the modeling of the ASLP leg, and it also presents more deeply the aspects of the ASLP leg. It compares the accuracy of several model, but also their computational power efficiency. It shows, for each model what are their extents and limitations.

Chapter 3 addresses the issue of gait modulation by the low-level CPG model, and it also explore the extent to which this approach could achieve dynamic footstep modulation. It compares two approaches, one with strong sensorimotor coordination and one without.

Chapter 4 describes a novel tactile sensor for leg load estimation in the Cheetah-Cub robot. This estimation is required by the approach detailed in chapter 3 that had the best results. Thus, integrating such a sensor is necessary for implementing this control in the real robot.

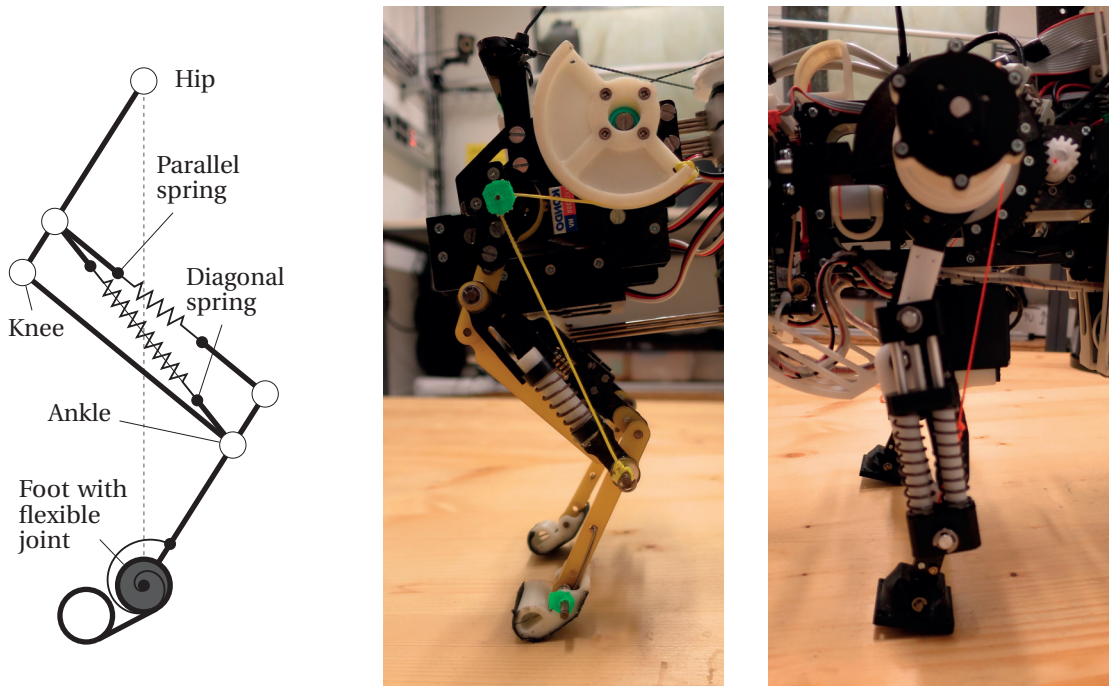
Chapter 5 discusses the development and integration of the Cheetah-Cub and Oncilla robots' firmware and internal software, and it also addresses related technical issues. It presents `robo-xeno`, a modular framework for the fast prototyping of embedded low-level real-time controllers, the result of such developments.

2 Modeling of the Dynamics of the ASLP Leg

All of the robots considered in this thesis feature the Advanced Spring Loaded Pantograph (ASLP) leg, as it simplifies the low-level control task, and adds self-stabilization properties to the robot. As stated by *Spröwitz et al.* [2013], for any new design of the ASLP leg, there is an important part of empirical tuning of the spring stiffness of the leg that need to be done. The ability of predicting with a good certainty the behavior of the ASLP mechanism from its design parameters (e.g. leg segment sizes, spring stiffness, motor choice, trunk weight) would provide lots of benefits in the design of new robots. It would for example give us the possibility to make trade-off between these design criterion. The hand-tuning task is also difficult and cumbersome, and is much harder to know if the any solution would be optimal regarding to a desired outcome (e.g. speed, agility). Using accurate models we might be able to reduce the number of hardware tests required to ensure any optimal criterion. Furthermore as we focus on the strong interaction between low-level controller and mechanics, the prototyping of new controller in simulation suffers a lot from the gap between simulation and reality.

Regarding how we could implement such models, there is generally two approaches in Rigid Body Dynamics to formulate the equation of motions: the use of *maximized* or *generalized* coordinates. At the beginning of our work on Cheetah-Cub, we were using exclusively the first approach. First, because of its widespread use in robotic simulation software packages, such as Gazebo, Webots and VRep, and secondly, because it is more difficult to model closed kinematic loops with the second. We will present here our own approach to model the ASLP leg in *generalized* coordinates with an analytical resolution of the closed loop, and compare it to the *maximized* approach in term of numerical stability and computational efficiency.

This chapter will first discuss the key properties of the ASLP leg we would like to model. Then we will present more deeply Rigid Body Dynamics methods and the mathematical details of the two approach. Finally, after the discussion of the extents, limits and validity of these two models, we propose early work on a new model that aims to solve limitations of both approaches.



(a) Schematic (b) Cheetah-Cub S (c) Oncilla

Figure 2.1 – ASLP leg description. a: Schematics from *Spröwitz et al.* [2013]. b: Cheetah-Cub AL front leg. c: Oncilla front leg. The leg presents two kinematic closed loop. Implementations differ in size, but also on the structure of the knee actuation, i.e. with (b) or without (c) a pulley.

2.1 Context

2.1.1 ASLP leg presentation

The ASLP leg structure is depicted in figure 2.1a. It consists of an opened pantograph, with two springs: one maintaining it open (the diagonal spring), and the second maintaining the distal and proximal segment parallel (the parallel spring). The last foot segment is also passively actuated using a rotational spring. The hip protraction/retraction is directly actuated by the motor, but the knee flexion is actuated by the mean of a cable. This opened pantograph presents two closed kinematic loops, both of them passively actuated by the diagonal and/or parallel spring.

Several robots implements this leg mechanism with a few different changes. The Oncilla design, without the use of the pulley, adds another nonlinear transformation between the motor axis and the leg length when the cable is under tension (see figure 2.1c).

2.1.2 Model specifications: ASLP leg key properties

We present here a list of features we would like to add to our model. We build this list mainly

based on the work described in *Spröwitz et al.* [2013].

Property 1: High body-over-leg mass ratio One of the first design properties of the ASLP leg is the use of the cable mechanism in order to deport the knee actuation from the limb to the trunk of the robot. This is a common properties seen in animals where the main body mass is more concentrated around the trunk and less in the limbs. There is an immediate benefit as it lowers down a lot the inertia of the leg, allowing faster retraction and protraction (swing) time of the leg. As an example in Cheetah-Cub the total mass of the leg is approximately 50 g and the mass of the trunk of the robot is around 850 g. This properties is quite important to model as it then relates closely to the dynamicity of the gait reachable by the robot.

Property 2: Non-linear spring and damping behavior As previously mentionned *Spröwitz et al.* [2013] highlighted some self-stabilization properties of Cheetah-Cub. In order for the robot to return to a stable gait limit cycle without any modification of the open-loop patterns, the energy added by the perturbation should be dissipated in some way. One assumption we could make is that this dissipation come from the nonlinear spring dynamic, and the internal friction in the leg mechanism. Therefore this an important properties we would like to take into account in our models to test this assumption.

Property 3: Kinematic constraints Design principle of the ASLP leg comes from *Witte et al.* [2000]. One of these design rules comes from the observation that in most of the small mammals, they legs should be considered as having three segment, and that during locomotion, the first and third segment are mostly parallel. This property is replicated in the ASLP leg with the pantograph mechanism. This property of maintaining almost parallel these two segments is important and should be retained in our model.

Property 4: Ground contact friction Finally, empirical knowledge on the Oncilla robot, especially experiments made by *Ajallooeian* [2015], shows that high foot friction coefficient are not desirable for these kind of robot. Indeed the parallel and toe spring adds a lot of serial compliant element, and some of them cannot be easily sensed. Some control architecture, likes the one described in *Barasuol et al.* [2013]; *Ajallooeian* [2015] or the approach described in chapter 3, relies on inverse kinematic to control the trajectory of the foot. This serial element makes it difficult to generate kinematically plausible trajectories, i.e. having each foot in contact with the ground following the same relative motion relative to the trunk. These inaccuracy are sources of jitter in the Oncilla robot and are easily reduced by allowing the foot to slip a bit on the ground, to deal with the inaccuracy of our methods. Therefore the possibility to modulate this effect is desirable, in order to replicate or to prototype new control methods that would be able to deal with those inaccuracies.

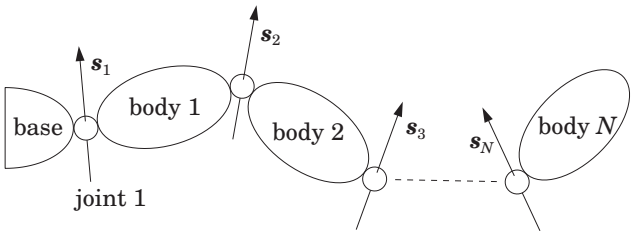


Figure 2.2 – Illustration of a kinematic chain in Rigid Body Dynamics (RBD). Graphics from *Featherstone* [2008]

Property 5: Asymmetric knee actuation The actuation of the pantograph is asymmetric. As shown in *Spröwitz et al.* [2013], for highly dynamic gait — with a high Froude number, see section 4.1.1 and [*Alexander*, 1989] — the behavior of the leg is mainly determined by momemtum dynamics as during stance phase, the cable is loosened and the knee motor is mechanically isolated from the pantograph. Furthermore *Spröwitz et al.* [2013] showed that for gait with a high Froude numbers, the active control of the knee in the stance phase could be forgotten, simplifying the control. This assymetric cable actuation is another important property we would like our model to have.

2.1.3 Rigid Body Dynamics

Rigid Body Dynamics (RBD) is widely used in robotics to model mechanical behavior [*Siciliano and Khatib*, 2008, Chapter 2]. It consist to see a robot mechanism as a set of rigid bodies, whose relative motion is constrained by perfect joints, as illustrated by figure 2.2. Rigid Body Dynamics (RBD) simulation is an old problem, and computer were used as early as 1970 to compute RBD models and there is a variety of software packages that could be used. We present here the mathematical formalism that could be used to describe RBD and a classification of some software frameworks that could be used for the ASLP leg modeling.

2.1.3.1 Equation of motion and unilateral constraints

Under this assumption, the general formulation of the dynamics of the system would be [*Featherstone*, 2008; *Siciliano and Khatib*, 2008, Chapter 3]:

$$H(q)\,\ddot{q} + C(q,\dot{q}) = \tau \tag{2.1}$$

Where q is a set of variables describing the pose of the bodies of the system, $\dot{q}$ a set of variables describing the velocities of the system, and $\ddot{q}$ their acceleration. $H(q)$ is called the mass matrix, and has a role similar to the mass term in Newton's second law of motion. $C(q,\dot{q})$ is a term that collect all gravitational forces and nonlinear quantities such as Coriolis effect,

and could even be augmented to account for all viscous-elastic effects. Finally τ accounts for all external forces applied to the system.

The equation (2.1) alone would not be sufficient to model a legged system. Indeed those system consist of a floating base joint that interact with the environment through dynamic contacts. Additionally robotics system may have joint limits that reduces the range of motion between two bodies. These constraints have two interesting properties: they are unilateral constraints, and they can be seen complementary to the acceleration $\ddot{q}$ [*Featherstone*, 2008, Section 11.3], i.e. equation (2.1) could be rewritten:

$$H(q)\,\ddot{q} + C(q,\dot{q}) = \tau + K(q)\,\lambda \tag{2.2}$$

Where λ are solutions to the Linear Complementary Problem (LCP):

$$\dot{\xi} = M\lambda + d, \quad \dot{\xi} \geq 0, \quad \lambda \geq 0, \quad \dot{\xi}^{\mathsf{T}}\lambda = 0 \tag{2.3}$$

λ are Lagrange multipliers that express the effect of the constraints on the system. ξ is a vector of linear quantities of the system, that describe the constraints and which are complementary to these multipliers. For example as seen in *Featherstone* [2008], ξ are chosen to be the separation velocities of the active contact points. The section 2.3.1 also illustrates how we can formulate joint end-limit constraints with this formalism.

2.1.3.2 Differences between RBD Framework

Maximized vs generalized coordinates framework The first differences between RBD frameworks is the choice of the coordinate system. In the maximal coordinates case, each body dynamics is considered independently. The vector q is the aggregation of all coordinates representing the poses of each bodies. The restriction by the joints on the relative motion between bodies is expressed as bilateral constraints on the relative acceleration and velocities of these bodies. On the other hand generalized coordinates frameworks use an optimal set of variables that fully describes the system, e.g. by using the relative joint angles between each bodies. This difference has a lot of consequences in the structure of equation (2.2), as it is illustrated by the structure of $H(q)$, see figure 2.3.

The advantages of the maximized coordinates approach is to be generally faster for small systems, as no global quantities needs to be computed (i.e. $M(q)$ in (2.1) is block-diagonal), and the joint constraints could be computed relatively quickly. Models, especially those presenting closed kinematic loops, are also simpler to build. We just have to specify one by one all relative constraints between bodies as LCP constraints and let the solver ensure the consistency of the system. It is therefore easier to provide libraries that automate this simple process.

(a) Maximized (b) Generalized

Figure 2.3 – Structural differences in inertia matrices between maximized and generalized RBD framework. The topologies of these matrices would in this example correspond to a quadruped robot with two segments per leg. In the maximized case, the matrix is block-diagonal, each body inertia being described by a 6-by-6 block. In the generalized case the matrix is sparse and inertial relations can be made between main and distal bodies.

Although more complex to use, generalized coordinates framework relies on the kinematic structure of the mechanism (figure 2.3b). By using $H(q)$ we could determine inertial relation between bodies. This as a lot of advantages for legged locomotion, where impacts and Ground Reaction Force (GRF) are applied on the most distal segment and their effect is propagated to the main floating body through the kinematic chain. With generalized coordinates, the use of the jacobian of the system would be sufficient to determine the effect of this forces on the whole system, but with maximized coordinates, this relation is determined through $K(q)$ and the numerical resolution of λ, see equation (2.2). This last process could be error-prone, especially when using software libraries developed for the video game industry, that prefers their solver to be numerically stable — to avoid bad-looking, jittering effect of bodies —rather than physically precise. For example, Open Dynamics Engine (ODE) address this issue, by letting its solver not strictly meet the relative motion constraints, and adds error reduction parameters that adds nonphysical forces to reduce the drift between bodies. This makes the joints non-rigid, but furthermore, it may be difficult to tune correctly ODE's correction parameters, namely the Error Reduction Parameter (ERP) and the Constraint Force Mixing (CFM).

Symbolic framework Another main difference between frameworks is if they provide a pure numerical derivation of the equation of motion, i.e. given a vector state $(q, \dot{q})$ and a description of the system, to provide a function that compute the next state value, or if they provide a symbolic representation of the equation of motion. This symbolic representation could be used to integrate the equations of motion, but also be optimized for their numeric resolution.

Table 2.1 shows how some of the relevant framework for our modeling could be characterized.

Name	Coordinate system	Constraint Formulation	Constraint Solver
ODE	Maximized	LCP	Projected Gauss-Seidel
Bullet	Maximized	LCP	Iterative
Dartsim	Generalized	LCP	Projected Gauss-Seidel (from ODE)
Robotran	Generalized	Soft constraints	N.a.
còdγn	Generalized	Event-based modification of equations (hybrid dynamical system)	analytical

Table 2.1 – Characterization of RBD framework.

2.2 Comparison Between Maximized and Generalized Coordinate Model of the ASLP Leg

Historically, our first approach to build models of Cheetah-Cub and Oncilla was to use a maximized coordinate approach. However we are here interested to compare it to generalized coordinate approach, as it may yield more accurate models.

There exist many algorithms to solve closed kinematic loops, but those could be complex to use, and would erequire more computational power. However we found that we could solve analytically the closed kinematic loops as shown in appendix A. This analytical solutions implies that:

- The compression forces of the diagonal and parallel spring applies a torque on respectively q_2 and q_3 joints. These torques can be expressed as nonlinear function of q_2 and (q_2, q_3), see section A.4.
- The joint end-limit of q_3 is not constant and depends non-linearly on the joint q_2, see section A.3.

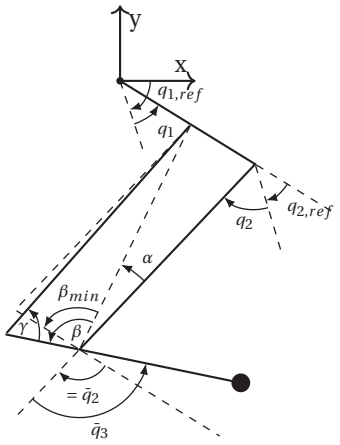


Figure 2.4 – ASLP leg joint and angle notation.

2.2.1 Method requirements

Concerning the RBD framework suitable to use with the maximized approach, any RBD simulator with a frictional contact model and an interface to build closed kinematic loop would be sufficient. This is the case for most popular, commercial or open-source robotic

simulator based on ODE or Bullet, such as Webots, Gazebo or VRep. The second approach however needs a direct access to the constraint solver due to the unusual joint end-limit formulation. Most of the available framework completely makes the constraint solving opaque: we cannot provide our own K,M and d matrices in equations (2.2) and (2.3). One of the easiest solution will be to provide our own LCP solver implementation and use a symbolic framework that does not impose its own LCP solver, such as Robotran or cȯdγn.

We summarize in table 2.2 the two approaches and all the requirements we need to implement the features described in section 2.1.2. Section 2.3 will present our extension of the classical constant joint end-limit constraints as a LCP to the state dependent case.

	Maximized Coordinates	**Generalized Coordinates**
Properties		
Kinematic structure	Closed kinematic chains	Serial chain
Joint Limit	Constant	State dependent
Asymmetric Knee actuation	Diagonal joint variable stop limit	Unilateral Constraint
Requirements		
Framework	end-limit constraints frictional contact model closed kinematic chain solver	Access to RBD equation terms Access to LCP solver frictional contact model
Inputs	Model data Reference angles	Model data linear length to angle mapping linear force (spring, cable) to joint torque mapping Reference angles
Suitable Frameworks		
	Webots Gazebo (ODE) Gazebo (Bullet) Gazebo (Dartsim) Dartsim	Robotran + LCP solver SDFast Codyn + LCP solver

Table 2.2 – Description of the two considered approaches and their requirements. References angles are described in section A.2, spring to joint torque relation in section A.4 and state dependent joint end limit LCP formulation in section 2.3.3

2.3 State Dependent Joint End-limit Constraint LCP Formulation

We will first present, as an illustration and introduction to constraint formulation as a LCP, the constant case, and then propose our method for the state dependent case.

2.3.1 Constant end limit formulation

When a joint i (identified by q_i) reaches its limit, we would need to add a torque or a force that is forbidding the joint to go further on that direction. Let us consider p joint that have reached a limit at any given time. In the case of an upper limit, respectively lower, this force should be negative, respectively positive, and act only that particular joint. $K(q)$ is therefore a $n-by-p$ constant matrix with ∓ 1 coefficient selecting which constraint force act on which joint, and its direction.

$$K(q) = \begin{pmatrix} 0 & 0 \\ \vdots & \vdots \\ \mp 1 & 0 \\ \vdots & \vdots \\ 0 & \mp 1 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix} \tag{2.4}$$

Furthermore, when a joint hits a limit, it cannot go further beyond that limit, but could only have an acceleration in the other direction. We can therefore choose for that joint $\xi_i = \mp \dot{q}_i$ or in matrix form:

$$\xi = K^\intercal \dot{q} \tag{2.5}$$

This choice of variable is suited for a LCP (2.3). Indeed if a contact is maintained ($\dot{\xi}_i = 0$) there could exist a force that would maintain this contact $\lambda_j > 0$. On the opposite, if there is separation $\dot{\xi}_i > 0$ no constraint forces should act on the system ($\lambda_j = 0$).

Now to finish the formulation of the LCP, we need to express $\dot{\xi}$ linearly in function of lambda, which can be done using (2.2):

$$\dot{\xi} = K^\intercal \ddot{q} = K^\intercal H(q)^{-1} K\lambda + K^\intercal H^{-1}(q)\left(\tau - C\left(q, \dot{q}\right)\right) \tag{2.6}$$

Which led to the following LCP parameters:

$$M = K^\intercal H(q)^{-1} K \tag{2.7}$$

$$d = K^\intercal H^{-1}(q)\left(\tau - C\left(q, \dot{q}\right)\right) \tag{2.8}$$

2.3.2 Impact and error correction

There is two problems not covered in the example above. The first one is the computation of the impact when the end limit is reached. For complex constraint like contact constraints, this is not trivial and this impact would lead to a discrete change of $\dot{q}$. In the case of the joint end limit, this computation is much simpler. The activation of the constraint will put the velocity of the joint to zero.

The second issue is related to the integration scheme we use for resolving equation (2.2). Indeed at each time-step, we will detect which joint have reached their end limit, and compute K accordingly. When a fixed time-step is used, a constraint could be activated with the joint position already beyond or over its limit. Two approaches could be used. The first one would be to use a variable time-step, where we reduce the integration time-step until all newly activated constraint are not violated more than a specific threshold. Another solution would be to add a correction term to reduce that error over the next integration steps.

The constraint on the separation acceleration will become $\dot{\xi}$:

$$\mp\ddot{q}_i \geq \mp\left(K_{corr}\left(q_i - q_{i,\pm bound}\right) - \nu_{corr}\left(\dot{q}_i\right)\right) \tag{2.9}$$

We can easily see that for the LCP parameters M will be left unchanged, but d becomes:

$$d = K^{\mathsf{T}} H(q)^{-1}\left(\tau - C\left(q,\dot{q}\right)\right) + K^{\mathsf{T}}\left(K_{corr}\left(q_{i,\pm bound} - q\right) + \nu_{corr}\dot{q}\right) \tag{2.10}$$

One can notice that the correction stiffness and damping term are directly expressed on the generalized coordinate q. Their dynamic is dependent of the dynamic properties of the mechanism ($H(q)$ and $C(q,\dot{q})$). Their equation is of the form of a second degree oscillator, so we can relate them to the resonance frequency and the quality of this oscillator.

$$\tau_{corr} = \frac{1}{\sqrt{K_{corr}}},\ Q_{corr} = \frac{\sqrt{K_{corr}}}{\nu_{corr}} = \frac{1}{\tau_{corr}\nu_{corr}} \tag{2.11}$$

To have numerically stable integration, we should choose $Q_{corr} < 0.5$ (over damped regime), and τ_{corr} relatively larger than the integration time-step.

2.3.3 State dependent extension

As seen in the resolution of the ASLP leg, we have the joint limit of q_3 that is not constant, but depend on the value of q_2:

$$q_3 \geq q_{3,-bound} = -q_2 \tag{2.12}$$
$$q_3 \leq q_{3,+bound} = -q_2 + B(q_2) \tag{2.13}$$

For the upper bound, it means that we cannot use anymore $K^\intercal$ in equation (2.5) for expressing the separation velocity of the joint ξ_3, since it should take into account the value of q_2. The new separation speed is now in the upper case:

$$\xi_3 = \dot{q}_3 + \left(1 - \frac{\partial B}{\partial q_2}\right)\dot{q}_2 \tag{2.14}$$

That should be differentiated to give the constraint on the separation acceleration:

$$\dot{\xi}_3 = \ddot{q}_3 + \left(1 - \frac{\partial B}{\partial q_2}\right)\ddot{q}_2 - \frac{\partial^2 B}{\partial^2 q_2}\dot{q}_2^2 \tag{2.15}$$

From this last equation, we can create a new matrix $\bar{K}$, which is identical to K for the other joints but its column corresponding to q_3 constraints becomes:

$$\bar{K} = \begin{pmatrix} \vdots & \vdots & \vdots \\ \vdots & 1 - \frac{\partial B}{\partial q_2} & \vdots \\ \vdots & 1 & \vdots \\ \vdots & \vdots & \vdots \end{pmatrix} \tag{2.16}$$

and we should use now the following LCP terms:

$$M = \bar{K}^\intercal H(q)^{-1}\bar{K} \tag{2.17}$$
$$d = \bar{K}^\intercal\left(q\right)\left(\tau - C\left(q,\dot{q}\right)\right) + \bar{L} \tag{2.18}$$

where $\bar{L}$ is the correction term that now depends on both q_2 and q_3:

$$\bar{L} = -\frac{\partial^2 B}{\partial^2 q_2}\dot{q}_2^2 + \bar{K}^\intercal\left(K_{corr}\left(B(q_2) - q_3\right) + \nu_{corr}\left(\frac{\partial B}{\partial q_2} - \dot{q}_3\right)\right) \tag{2.19}$$

2.4 Implementation details

Maximized Coordinates The model has been implemented with Webots. Most of the mechanical properties of the ASLP leg were simply modeled using Webots standard interface.

The most complicated feature was the implementation of the asymmetrical actuation of the knee, which required to circumvent Webots Application Programming Interface (API) and address directly the ODE API. Instead of using a standard actuator that will add force on one of the joint of the robot, the end-limit of the diagonal prismatic joint l_d was changed accordingly to the desired position. A speed limitation was added on how fast this end-limit position could be changed. First the torque $\tau_c$required to pull the cable was estimated by computing the compression force applied by the diagonal spring. Actually this estimation could be considered as an upper-bound of the actual required torque, as the robot weight and momentum tends to help the flexion of the leg. Then the maximal achievable speed of the end-limit is computed as:

$$\omega_{\max} = \omega_{\text{noload}}\left(1 - \frac{\tau_c}{\tau_{\text{stall}}}\right) \tag{2.20}$$

with ω_{noload} the maximal speed achievable by the motor and τ_{stall} the stall torque of the motor. This relation corresponds to a first order approximation of the limitation of a Direct Current (DC) motor. It is worth to note, than during voluntary flexion of the knee, the end-limit joint will at each integration step be outside of its admissible range. This requires the error reduction mechanism of ODE to be active, and fine tuning of its ERP and CFM parameters are required.

Generalized Coordinates We choose to rely on a combination of the cȯdγn framework to symbolically generate the term of equations (2.2) and (2.3). This task was quite easy through cȯdγn generative syntax and the fact that all the RBD algorithm are coded in cȯdγn specific language. We also modified cȯdγn RBD physic library to add the required modifications on the computation of the LCP terms to account for state dependent joint end-limits (section 2.3). Then a C++ program was used to numerically evaluate these generated functions and to solve the LCP problem, by using a Projected Gauss-Seidel algorithm with over-relaxation [*Mangasarian*, 1977]. Then the resulting λ were sent back to cȯdγn, which was still responsible to integrate the state $(q, \dot{q})$ using explicit Euler integration, and to maintain quaternion normalization. Since the LCP is solved externally to cȯdγn, no other integration strategy such as Leap Frog or Runge-Kutta could be used, as they require different LCP problem to be solved for a given iteration. In order to use these other strategies, the LCP solver should be available as an internal component of cȯdγn.

2.5 Efficiency and Stability Comparison

When implementing this generalized coordinates approach, we discovered that if we were not using small integration time-step, we could not have a numerically stable simulation. We assume that this small time-step are required because the models displays some high-frequency mechanical oscillators. This is directly an outcome of the fact that we model a system with a high body-over-leg mass ratio. There is a ratio of almost 1:85 regarding the

mass of one leg's segment ($\approx 10\,g$) and the weight of the trunk (850 g). However the springs in the leg are tuned to sustain a quarter of the robot mass. Furthermore the lever effect in the pantograph, requires these spring to even be stiffer than those required in a prismatic design — see section A.4 equation (A.24) and (A.25). Making these stiff springs act on light masses constitutes oscillators with a high resonance frequency ($\sqrt{k/m}$ for a linear system). If we integrate such oscillators using explicit Euler integration, we should use a time-step smaller than this frequency.

2.5.1 Methodology

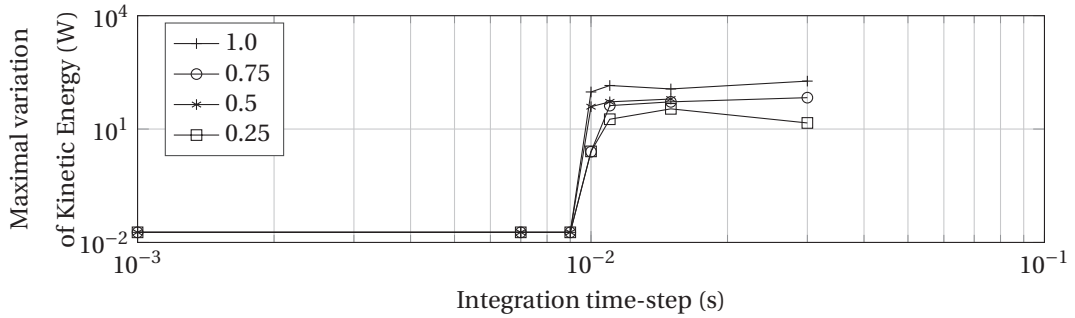
To test this last assumption and make a fair comparison of both models, we decided to use a toy simulation:

- The base of the robot was set to be fixed.
- Four ASLP legs are simulated, without any ground interactions or active actuation, only passive dynamics from springs and viscous friction.
- Both model were setup with the inertia and stiffness parameters extracted from CAD data of the latest version of Cheetah-Cub AL.
- The starting position was set to have a small amount of potential energy: the starting hip angle was set to 10° in front of the rest position and the knee 10% flexed.
- The simulation was left to iterate until the rest position of the system is met.

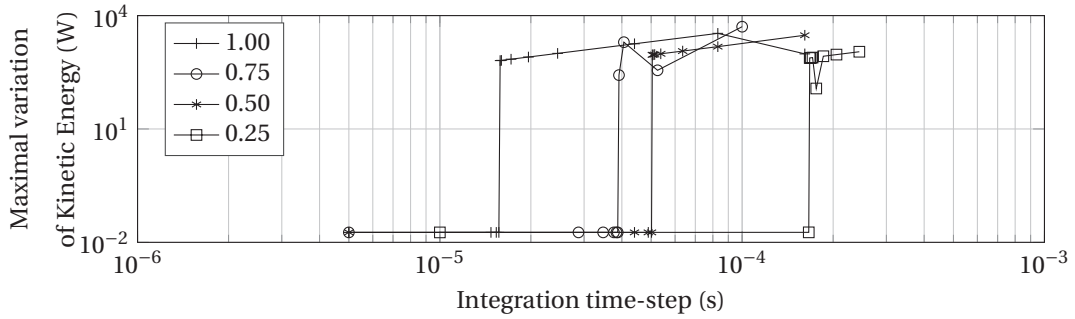
Then we could iterate this simulation several times while changing the springs' stiffness values to be a ratio of the value extracted from the CAD data. For each of these stiffness levels we could look for the numerical stability of the simulation: we monitor the kinetic power output of the system, which in this passive scenario, should go asymptotically to zero. Numerically unstable simulations will however have this quantity rapidly increase towards infinity. Then we perform a binary search on the integration time-step, to find the highest one yielding numerically stable simulation.

2.5.2 Results

Results for each of the modeling are shown in figure 2.5. We also performed a comparison of the execution time for a single time-step on a standard desktop computer in figure 2.6. Here we only considered stable simulations for computing the average computational time, as both Webots and our simulation with cȯdγn uses an iterative LCP solver. When the simulation becomes numerically unstable, these solvers cannot find suitable solutions for the constraints and soon hit their allowed maximal number of iterations for each integration step. In turn, this increase significantly the computation time for both approaches. We see from figure 2.5b that for the generalized coordinates approach we see an increase in the stable simulation time-step from approximatively 15 µs to 166.1 µs when a quarter of the desired spring stiffness is used. This correspond to a simulation speed of the system of approximatively 25 times down to 2



(a) Maximized coordinate stability binary search (Webots)



(b) Generalized coordinate stability binary search (cȯdγn)

Figure 2.5 – Comparison of the numerical stability of both modeling approach. The stable simulation time-step integration value are found by binary search for different stiffness value of the springs, expressed as a ratio of the actual robot original values ($\{1.0, 0.75, 0.5, 0.25\}$).

slower than the real-time. For the maximized coordinates approach, varying the stiffness did not change the value of the stable integration time-step which is 9 ms and correspond to a simulation speed of 2.7× faster than the real-time.

2.6 Discussion

2.6.1 Validity and limitations of the Webots simulation

The maximized coordinate implementation in Webots was done for two robots, Cheetah-Cub and Oncilla and was used for various projects. The most notable ones being the search for dynamic trot gait locomotion with Particle Swarm Optimization (PSO) in *Spröwitz et al.* [2013] (see figure 2.7), the prototyping of a controller stabilized with optical-flow detection [*Gay*, 2014], and the prototyping of a robust locomotion controller [*Ajallooeian et al.*, 2014].

One nice outcome from *Spröwitz et al.* [2013] is that we see similar results in terms of behavior of the Cheetah-Cub robot and its model. Both on hardware and simulation, the robot could display dynamic trot, i.e. with a small flight phase at each ronot's steps, where none of its legs touch the ground. Both simulation and hardware displayed a similar relationship

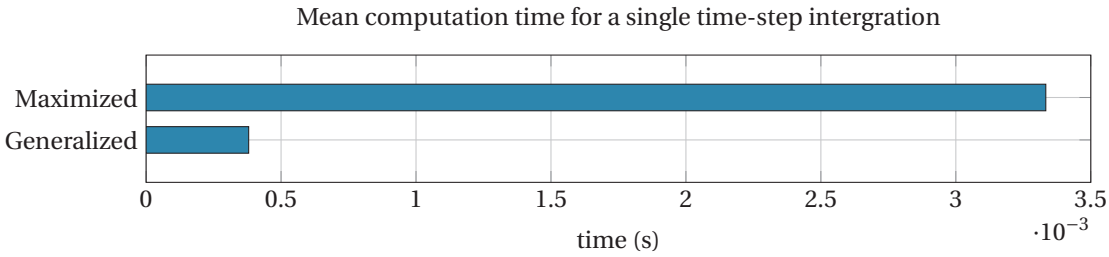


Figure 2.6 – Comparison of execution speed of both approaches. Only numerically stable simulation are compared.

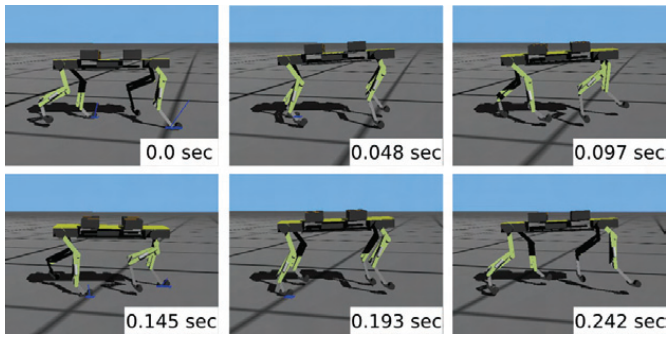


Figure 2.7 – Snapshots of Webots simulation of Cheetah-Cub from *Spröwitz et al.* [2013], performing a 3.5 Hz dynamic trot gait.

between speed, gait frequency, swing leg amplitude and desired duty ratio of the gait (see figure 9 of *Spröwitz et al.* [2013]). However this simulation had also unexpected results: the energy consumption of the simulated model is up to 15 times smaller than the real robot. It is questionable that all this loses comes only from the gear reduction and the servomotors energy conversion on the real robot. For the model, it may come from the fact that we often (at each active flexion) hit the joint limit of the diagonal spring mechanism, and ODE's error reduction technique might add unrealistic energy to the system. Furthermore figure 2.5a shows that there are no variations of the numerical stability of the model in regards to the springs' stiffness values, as one may expect to see. Again, since the model is most of the time hitting the diagonal spring joint end-limit, unrealistic forces may be added to this joint by the constraint solver. It is known that ODE's LCP solver is optimized to ensure simulation stability (i.e. avoiding fast acceleration or jitter of the bodies) over physical accuracy. The general behavior of the robot is quite similar to the real one, as we could have tested, but those model should not be used for purely quantitative estimations, such as estimating motor torques to choose a motor, or estimating the optimal stiffness of the ASLP leg spring.

2.6.2 Numerical instabilities of the cȯdγn simulation

The figure 2.5b concurs with our assumption that high mechanical resonance frequencies of the mechanical system plays a role on its numerical stability. Indeed the more stiff the springs are, the smaller the integration time-step should be to kept the simulation stable. However, systems expressed in generalized coordinates tends to be dimensionally smaller than their maximized coordinates counterparts. In the Cheetah-Cub case, 19 variables are required in the generalized case against 198 in the maximized one. This difference is well reflected in the average time to compute a single step, as it is one order of magnitude higher for Webots compared to cȯdγn. Furthermore, since cȯdγn is a symbolic framework, it might be that a numerical framework would be even be faster. However due to the numerical instabilities even if only $\approx 380\,\mu s$ are required to compute a single step, these time-step account for only $15\,\mu s$ which make the simulation 25 times slower than real-time. This impractically small time-steps made us reconsider performing a parameter identification of the cȯdγn model to have accurate simulation. However since we identified that high resonance frequencies are accountable for this unstability, we propose a third approach to circumvent this particular issue by neglecting the mass of the leg segment.

2.7 Generalized Coordinate Extension With Mass-less Leg Segments

The first thing with such approach is that we should not consider a completely mass-less leg. Indeed as shown by *Or and Moravia* [2016], if the leg is completely mass-less, as in their case, is the original Spring-Loaded Inverted Pendulum (SLIP) model, coulomb friction with the ground could not be modeled. As soon as the foot will start to slip — which is always the case when near the toe-off and touch-down events — it will gain an infinite speed, there will be no possible equilibrium of forces at the foot. Therefore the foot need to have a small mass, to allow the bounding of horizontal components of GRFs. For that purpose we choose to make the approximation to concentrate all of the leg mass in the foot.

Now we parametrize the relative motion between the trunk and the foot by a planar joint, i.e. a motion parametrized by three variable, a rotation q_1 around the hip axis, an extension L_{leg} and a final rotation θ, see figure 2.8. Algorithm from *Featherstone* [2008] are still able to compute symbolically the matrices $H(q)$ and $C(q,\dot{q})$ for such choice of generalized coordinates.

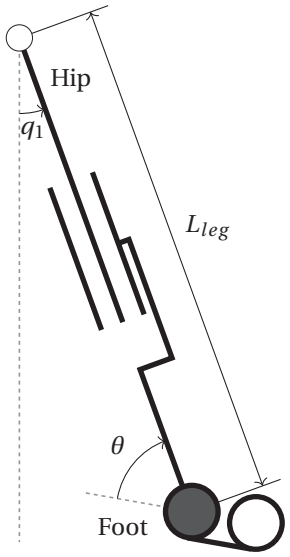


Figure 2.8 – Mass-less ASLP leg model parametrization.

However the efforts transmitted by such a joint is a bit particular for the ASLP leg, to take into account the dynamics of the springs. The hip torque is still controlled by the motor, however the linear force between the foot and the trunk and the torque transmitted to the foot is defined solely by the passive elements.

2.7.1 Numerical resolution of passive element contribution

Since the leg segments are mass-less, we consider that at each time-steps the springs should reach their equilibrium point, since the leg segments should instantly jump to that state, as they have no masses. We suggest here to pursue a numerical approach to compute this equilibrium, by considering the minimum of the potential energy of the system:

$$E_p = \frac{1}{2}K_D\left(L_{D,0} - l_d\right)^2 + \frac{1}{2}K_p\left(L_{P,0} - l_p\right)^2 + \frac{1}{2}K_F\left(q_{f,0} - q_f\right)^2 \tag{2.21}$$

Furthermore, using relations between different triangles in the leg, we can express the leg length L_{leg} and the foot orientation θ from the same variable $\{l_d, l_p, q_f\}$ (see section A.5). This is equivalent to tell that the system should respect the forward kinematic:

$$\begin{bmatrix} L_{leg} & \theta \end{bmatrix}^T = f(l_d, l_p, q_f) \tag{2.22}$$

(2.21) and (2.22) forms a quadratic optimization problem with non-linear equality constraints, that we could numerically solves using Matlab for different values of L_{leg} and θ, taking into consideration end-joint limits for $\{l_d, l_p, q_f\}$.

Then by writing the static force equilibrium of the segment l_3, we could retrieve the force

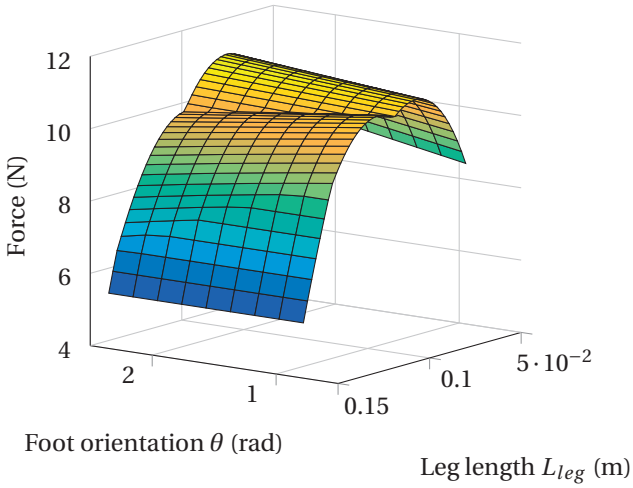


Figure 2.9 – Computation of the force between the foot and the trunk as a result of a quadratic optimization under non-linear constraints. The maximal leg stiffness is here $\approx 253.27\,\mathrm{N\,m^{-1}}$ which is one order of magnitude less than the corresponding diagonal spring stiffness of $4400\,\mathrm{N\,m^{-1}}$.

$||F_{leg}||$ transmitted by the leg and the value of the force $||F_2||$ passing through the l_2-l_3 joint:

$$\begin{bmatrix} \cos(\alpha) & \cos(\gamma) & \cos(q_3) & \cos(\beta) & 0 \\ -\sin(\alpha) & \sin(\gamma) & -\sin(q_3) & -\sin(\beta) & 0 \\ 0 & -l_3\sin(\gamma) & (l_3-l_\Delta)\sin(q_3) & (l_3-l_\Delta)\sin(\beta) & 1 \end{bmatrix} \times \begin{bmatrix} F_{leg} \\ F_p \\ F_2 \\ F_d \\ \tau_f \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.23}$$

Here again the various angles $\{\alpha, \beta, \gamma, q_3\}$ can be expressed in terms of $\{l_d, l_p, q_f\}$ using the laws of cosines and sines in several triangles (see section A.5).

The result of the offline optimization for various leg lengths and foot orientations is shown in figure 2.9, where the force F_{leg} is plotted against the state of the leg (L_{leg}, θ). Firstly, the system shows a strong non-linear stiffness that may indeed play a role in the self-stabilization properties of the ASLP leg. Second we see that the maximal stiffness value on these two dimensional curve along the leg length dimension is $\approx 253.27\,\mathrm{N\,m^{-1}}$ which is one order of magnitude less than the diagonal and parallel spring stiffness in this model which are respectively set to $4400\,\mathrm{N\,m^{-1}}$ and $1800\,\mathrm{N\,m^{-1}}$. Therefore we expect a simulation which will be much easier to integrate numerically.

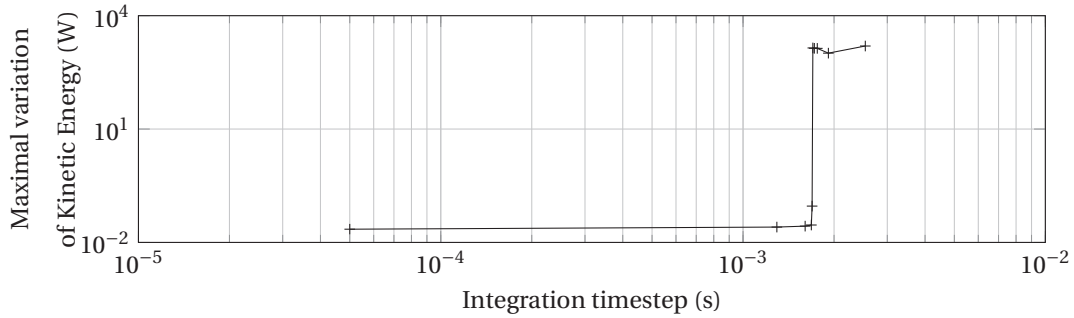


Figure 2.10 – Binary search of the highest integration time-step which is numerically stable for the mass-less leg segment model. We gain almost two orders of magnitude in comparison with the model considering the segment mass.

2.7.2 Numerical stability analysis

To integrate this approach with the cȯdγn framework, since the shape displayed in figure 2.9 is quite smooth, the force values have been integrated in a look-up table in the simulator, with a linear interpolation between the computed values.

Then the experiment described in section 2.5.1 was conducted on the new model to see if effectively, the largest stable integration time-step limit was increased for this model. Results are depicted in figure 2.10.

2.7.3 Discussion

We see an increase of two orders of magnitude in term of numerical stability of the model, since the highest stable integration time-step is now at 1.69 ms against 15.7 µs for the previous model. It is also worth to note that the mean computational time of the system on a standard desktop computer is still ≈ 350µs. Therefore this mass-less model can be computed almost five time faster than the real-time, which would be very efficient compared to all other methods we showed here.

By removing the mass of the intermediary segment, we confirmed than the high stiffness of the spring and their configuration was the cause of instabilities in our first method. A problem that will be the same considering any formulation and implementation in generalized coordinates, regardless of using a numerical or analytical resolution of the closed kinematic loops. However this model still lacks many of the features we required in section 2.1.2. There is no asymmetrical knee actuation, there is no way to specify viscous friction forces between leg joints, and it still lacks a coulomb friction model for the contacts. The first feature would require to make our off-line optimization more complex, by adding new input variables (the cable position and maximal tension) and more constraints such as specifying thate the cable should always be in tension and never in compression. If the system becomes too complex, we may not be able to use a look-up table, and this solution may not be more efficient computationally.

Finally one of the great benefit of this new approach, is we do not require to solve complex joint end-limit ourselves. We could use any RBD software that support planar joints. We just then need to plug the results of our offline optimization to specify the mechanical efforts that the passive dynamics of the leg induces between the foot and the hip joint.

3 Modulation of Gait Pattern with Low-Level Controller

This chapter is certainly the one the most in relation to our main goal, i.e. to use a bioinspired control architecture to handle complex locomotion scenario. This is an ambitious goal, and if we would achieve it in its globality, we would require the robot to scan its surroundings, plan and then execute a trajectory. For a legged robot, executing a defined path would be to move towards a goal, while placing its feet at designated absolute positions. This kind of problematic, i.e. of crossing a very challenging, well-defined and known terrain was the kind of problematic the Defense Advanced Research Projects Agency (DARPA) locomotion learning challenge was focusing on. Since the environments were there very challenging, the robots required to make complex, difficult foot placement to be cross the terrains. Often only static slow gait were used. The winning approach, proposed by *Buchli et al.* [2011]; *Kalakrishnan et al.* [2011], relied on inverse dynamic model-based control, i.e. a very fine balancing control of the robot. Considering these results and our control architecture (figure 1.4), we assume that addressing these very challenging scenarios without this higher-level control may indeed not be possible, and for sure, too challenging to address directly without adressing first simpler scenario.

We would like here to address the problem of dynamic footstep placements (figure 3.1). Here each of the footsteps is not required to take place at very precise and narrow location. There is an obtstacle in front of the robot, and without a fine adjustement of a few of its expected footstep location, it would fall. We known from *Buchli et al.* [2011], that any model-based approach would easily deal with this situation but with our approach relying solely on the modulation of a lower-level controller, this is not yet proven.

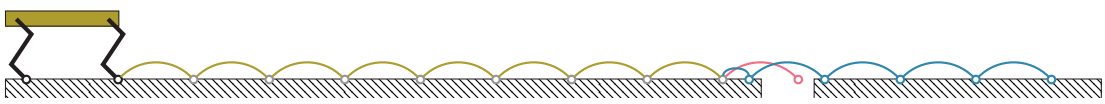


Figure 3.1 – Obstacle cross-over requiring dynamic footstep placement scenario.

3.1 Problem Statement

Since our initial approach *Spröwitz et al.* [2013] was very sucessfull for flat terrain locomotion, we want to see if it would be able to address this problem. However the Central Pattern Generator (CPG) model described there is using a parametrization in the joint space and footstep modulation is an operational space task. Therefore our first requirement was to move the original CPG model to a task space parameterization via a bioinspired foot locus (section 3.2).

As an open-loop approach may not be sufficient, we also would like to see if the use of strong sensorimotor coordination could help to this task space modulation. For that purpose we propose to use here the Tegotae feedback rule proposed by *Owaki et al.* [2012, 2013] and adapt it to Cheetah-Cub specifities.

3.2 Central Pattern Generator (CPG) with Bioinspired Kinematic

3.2.1 Foot locus generation

We can define the foot locus for quadruped locomotion as the trajectory of the foot relatively to the hip in the parasagital plane of the robot. A simplified representation of such a trajectory would be to use an half-ellipsis, permitting the foot to leave the ground between steps and have a flat trajectory during stance phase. For example, *Barasuol et al.* [2013] suggests to morph the phase portrait of an Hopf oscillator to generate such trajectory. However, as they contract the vertical axis of the oscillator to make an ellipsis, if the Hopf oscillator has a linear phase, the foot will not follow the trajectory linearly. It will spend more time near the touch-down and take-off event. We would like to generate trajectories to use another strategy that will preserve the phase of the oscillators to keep more control of the timing of the trajectory tracking.

To generate this trajectory two approaches have already been performed on the Cheetah-Cub robot: a) use simple parametrized function in joint space, such as sine and bumps as shown in *Spröwitz et al.* [2013], and let the foot locus emerge from systematic search b) use kinematic data from animals that are scaled to the size of the robot and use inverse kinematic to map it to the joint space as shown in *Moro et al.* [2013].. The latter as the advantages to offer trajectory closer to what is seen in animal locomotion. Under the assumption that animal locomotion is optimized as the result of evolution, it may results in better performing gaits. However it could be argued that the scaling effect to a different morphology may have an effect, and as there are little possibilities to modify easily these trajectories, it is difficult to verify this assumption on the optimality regarding any criterion. On the other hand the first approach could be used to optimize the overall gait as it is a fully parametrized approach. However combination of these parameters can be chosen that produce unstable gait patterns. Furthermore, when someone want to modify a certain aspect of the robot gait, for example the step length or step height, combination of these parameters have to be modified, as for example the amplitude of the swing movement of

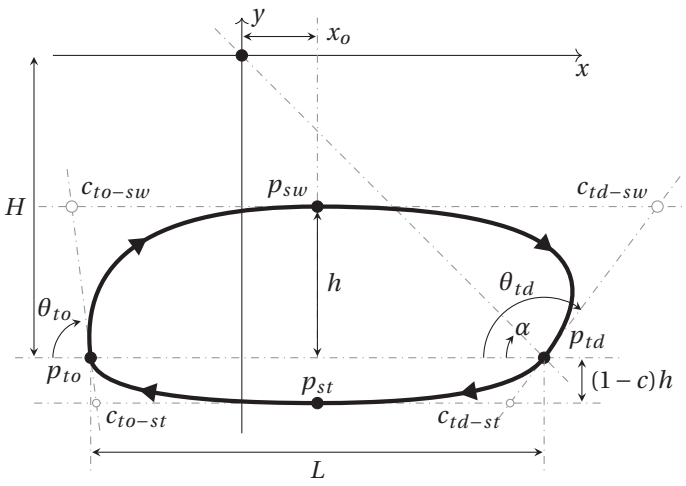


Figure 3.2 – Generic foot locus parametrization. We aim to parametrize a trajectory for the foot that is close to the one found in animals. With such parametrization we can relate more easily the parameters of our controller to variable of other model such as SLIP. The curve is itself a quadratic Bézier curve [*Farin*, 1990] defined by the four points p_{sw}, p_{td}, p_{to} and p_{st} and their corresponding control points c_{to-sw}, c_{td-sw}, c_{to-st} and c_{td-st} (See table 3.2). The origin is chosen to be the center of the hip axis.

the leg have an effect on both the step length and the angle of attack. This approach is in practice unfeasible for footstep modulation.

We propose here an intermediary approach: we define explicitly in the sagital our desired foot locus, and we rely on inverse kinematic to generate the joint trajectory. Instead of a fixed trajectory we parametrize it with variables that are meaningful for locomotion.

If we look at one of the most elegant model for legged locomotion, the SLIP model [*Geyer et al.*, 2006], three relevant kinematic quantities qualifies a locomotion pattern: the leg length, the angle of attack and the Center of Mass (CoM) height. Furthermore it have been shown that two strategies plays a role to the self-stabilization of a running legged system: the leg retraction [*Seyfarth et al.*, 2003] and the leg extension strategies [*Daley et al.*, 2009]. The former

Symbol	Description
H	Desired Hip height
h	Desired Step height
L	Desired Step Length
x_o	Mid stance offset, related to desired angle of attack α
θ_{td}	Touch down incidence angle
θ_{to}	Take off incidence angle
c	stance phase trajectory compression. If $c \to 1$ then the stance trajectory is completely flat, the lower, the more bumpier. $c \in [0;1[$

Table 3.1 – Listing of the parameters of our foot locus parametrization (see figure 3.2).

Point	Description	Sagital Plane Coordinates	
		x	y
p_{to}	Take-Off	$x_o - \frac{L}{2}$	$-H$
p_{sw}	Mid-swing	x_o	$h - H$
p_{td}	Touch-down	$x_o + \frac{L}{2}$	$-H$
p_{st}	Mid-stance	x_o	$-H - (1-c)h$
c_{to-sw}	Bézier control point between take off and mid-swing	$x_o - \frac{L}{2} + h\tan\left(\theta_{to} - \frac{\pi}{2}\right)$	$-H + h$
c_{td-sw}	Bézier control point between mid-swing and touch-down	$x_o + \frac{L}{2} + h\tan\left(\theta_{td} - \frac{\pi}{2}\right)$	$-H + h$
c_{td-st}	Bézier control point between touch-down and mid-stance	$x_o + \frac{L}{2} - (1-c)h\tan\left(\theta_{td} - \frac{\pi}{2}\right)$	$-H - (1-c)h$
c_{to-st}	Bézier control point between mid-swing and touch-down	$x_o - \frac{L}{2} - (1-c)h\tan\left(\theta_{to} - \frac{\pi}{2}\right)$	$-H - (1-c)h$

Table 3.2 – Foot locus key point definition and parametrization in the sagital plane, related to figure 3.2 and table 3.1. Origin is taken at the hip joint axis.

shows that when the leg starts its retraction a bit before the contact with the ground (touch-down), it increases the size of the stability region of the SLIP model. In the latter, it is shown that Guinea Fowls, when running over an hidden hole, extend their leg in order to limit the drop of the CoM height and to avoid to fall. It is assumed that they command their foot to follow an ellipsis and they rely on the compliance of their leg for their feet to follow a flat trajectory during stance phase. This strategy increase their robustness as they do not require to perceive or modulate their gait wen running over a small hole.

Following these observations, the foot locus depicted in figure 3.2 is proposed. It is a Quadratic Bézier Curve [*Farin*, 1990], defined by the four key point of the gait cycle (see table 3.2), i.e. the touch-down point where the foot enter in contact with the floor, the mid-stance point, the take-off point, at which the foot leaves the ground and the mid-swing point. These points are parametrized with the variable described in table 3.1. H,x_o and L relates to the leg length and angle of attack α of the leg. The depth of p_{st} is also parametrized by the c parameter to leverage the leg extension strategy. By setting this parameter to 1 there will be no leg extension during stance phase and the foot will be commanded to have a flat trajectory. By setting it to 0, the foot locus will display a more ellipsoidal shape. Furthermore with the help of the control points and the θ_{td} and θ_{to} angles, respectively the incidence angle at touch-down and toe-off angle the leg retraction strategy could be implemented and tuned. It is just needed to put a value of θ_{td} higher than $\pi/2$.

3.2.2 Trajectory timing

The previous section described the desired shape we would like our foot to follow. However its still lacking time information to build an actual trajectory. This timing is quite important

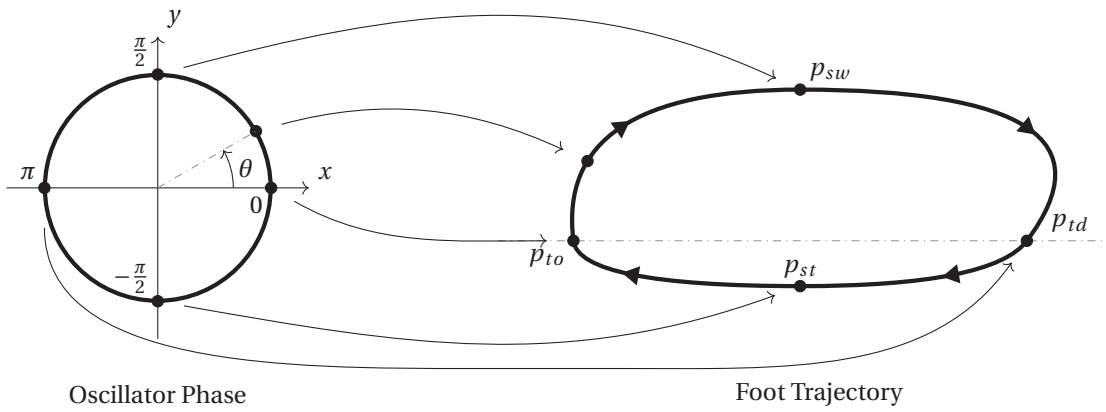


Figure 3.3 – Mapping of the phase of the oscillator to the the foot locus. The phase of the oscillator is used as the curvilinar abscissa of the bezier curve to specify the current tracked point. $[0;\pi]$ corresponds to the swing phase and $[-\pi;0]$ to the stance phase.

in locomotion, as to regulate their speed, animals changes their duty ratio [*Alexander*, 1989]: i.e. the ratio of the duration of the stance phase over the duration of one gait cycle. If the position of the figure 3.2 would be taken linearly in time, we would not have any control over this important quantity.

We propose to use a non-linear phase oscillator to encode this timing information, and map the phase of this oscillator desired foot trajectory, as depicted in figure 3.3. The swing phase of the gait cycle is mapped to $[0;\pi]$ and the stance phase to $[-\pi;0]$, while 0 and π correspond to the take-off and touch-down event of the gait cycle. The non-linear phase oscillator is defined as a piecewise linear transformation of the phase of a phase oscillator using the following function:

$$\Theta = F(\theta) = \begin{cases} \dfrac{\theta}{1-d} & \text{if} \quad \theta < \pi(1-d) \\ \dfrac{\theta + 2\pi d}{1+d} \end{cases} \tag{3.1}$$

where d is the *desired* duty factor. This function simply maps the touch-down event time that occurs at a time π to occur at time $(1-d)\pi$, while keeping the take-off point at the same timing. With $d < 0.5$ the *expected* stance phase will be shorter. It is worth to note that those variable are expectation. Due to the serial compliance of the ASLP leg, there are delays that occurs when for example we control the leg to leave the ground, making the actual touch-down and take-off event not occur exactly when the oscillator reaches $\Theta = \pi$ or $\Theta = 0$. For inter-limb

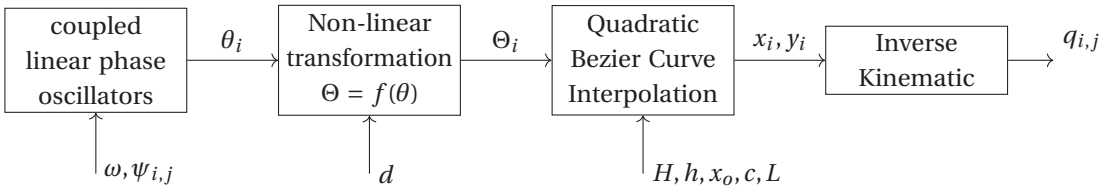


Figure 3.4 – Overview of the CPG architecture. Parameter for each block are displayed on the bottom. Coupled linear phase oscillator encodes global gait frequency and inter-limb coupling. Non-linear phase transformation encodes *desired* duty ratio. Bézier curve interpolation encodes the desired foot trajectory. Finally inverse kinematic is producing the desired joint trajectories.

synchronization, we can simply use as our CPG model a network of coupled phase oscillators:

$$\theta_i = \omega + \sum_j w_{i,j} \sin\left(\theta_j - \theta_i - \psi_{i,j}\right) \tag{3.2}$$

It is worth to note that the coupling is performed on the linear phase θ and not the desired non-linear phase Θ to insure phase locking of the network.

We have now synchronized patterns of desired foot positions. To transform them to desired joint angle, we can use inverse kinematics. For the case of the Cheetah-Cub and Oncilla models, these functions are described in appendix A.1.

3.2.3 Summary of the architecture

The whole CPG model built without sensorimotor coordination and a bioinspired foot locus is described in figure 3.4. All the control parameters of this low-level controller are listed in table 3.3. For most of the experiments, we do however fix some of these parameters. For example the phase difference is always chosen to make the network of coupled oscillators to converge towards a specific gait such as diagonal or lateral walk, trot, bound or diagonal or rotary gallop. These differences barely changes during the same experiment as we do not investigate gait transitions. The compression parameter c is also often fixed to a value close to 1, to enforce the foot to have a flat trajectory during stance phase. This is not necesarly required with the ASLP leg for highly dynamic gait [*Spröwitz et al.*, 2013], but this is required for slow static gait, when the robot is gaining momentum to reach high dynamic gait.

When implemented on the Cheetah-Cub robot, this control approach was very sucessful as we could hand-tune all of these parameters rapidly within a few minutes. The parameter were linked with important locomotion quantities, e.g. step length, step height, robot height, angle of attack, amount of leg retraction. For example, if the front limb would need to be put more in front of the robot — effectively reducing the angle of attack α — one would just

Symbol	Description	Range	Remarks
ω	pulsation/frequency of the gait ($\omega = 2\pi f$)	$]-\infty;+\infty[$	same for all legs
$\psi_{i,j}$	Desired phase difference between leg i and j	$[-\pi;\pi]$	coherent phase difference between all legs required for each desired gait
d	Desired duty factor	$]0;1[$	different for fore and hind legs
H	Desired hip height	$]h;H_{\text{robot}}[$	different for fore and hind legs
h	Desired Step height	$[0;H]$	different for fore and hind legs
L	Desired Step Length	$[0;+\infty[$	same for all legs
x_o	Mid stance offset	$[0;+\infty[$	different for fore and hind legs
θ_{td}	Touch down incidence	$[0;\pi]$	different for fore and hind legs
θ_{to}	Take off incidence	$[0;\pi]$	different for fore and hind legs
c	stance phase compression	$]0;1[$	different for fore and hind legs

Table 3.3 – Listing of the control parameter of the open-loop approach with bioinspired foot locus.

need to change the corresponding x_o parameter. In tbe controller described by *Spröwitz et al.* [2013], we should have tuned three parameters: the hip amplitude, hip offset, and the phase difference between hip and knee trajectories.

3.2.4 Gait parameters optimization

Looking at the table 3.4 we have still 14 to 16 opened parameters (depending if we fix c or not). Even if both in simulation and on the real hardware we could rapidely find nice looking gait. With such an high number of opened parameters, we still require optimization algorithms. We rely here on Particle Swarm Optimization (PSO) [*Kenndy et al.*, 1995]. We would like the objective function to consider multiple criteria (in order of importance): a) The robot should reach a limit cycle and not fall b) The phase mapping for each leg should be respected c) The feet should not slip on the ground d) The amplitude of pitch and roll movement should not be too large e) It should be fast.

Putting all those criteria in a single function could be tedious, as the PSO may not be able to optimize all these criteria at once. One solution is to use a fitness function defined by stages as shown in *Van den Kieboom* [2014], as PSO does not require the actual fitness value of each particle, but just an weak ordering of the particles. Therefore we can define a hierarchy of different fitness function, or stages, each of them with a threshold. If a particle mets the threshold of the first stage, it is evaluated using the next stage and so on. If two particle are compared and are at the same stage of evaluation, the corresponding fitness function will be used. Otherwise the particle with the higher stage is considered the best. This stage fitness function is still a weak ordering of the particle and can be used by a PSO algorithm.

With such an objective function, we can use the most simple and important criteria as the first stage and by small step makes the objective function more and more complex (see

Stage	Description	Value	Threshold
0	Reach a limit cycle	$V_{com}e^{t_r-1}$	$t_r > 0.5$
1	The phase mapping is respected	$V_{com}e^{-10c_{mis}^2}$	$c_{mis} > 0.3$
2	The feet are not slipping	$V_{com}e^{-10c_{mis}^2}e^{-l_{slip}^2/500}$	$c_{mis} > 0.3$
3	Pitch and roll are smalls	$V_{com}e^{-10c_{mis}^2}e^{-l_{slip}^2/500}e^{-(\alpha+\beta)/0.3}$	—

Table 3.4 – Staged fitness function used for optimization of the CPG architecture. V_{com} is the average speed of the robot. t_r is the ratio of the simulation completed before the robot falls. c_{mis} is the average time per leg where the foot is in contact while the phase is in $]0;\pi[$ or the foot is not in contact and the phase is in $]\pi,2\pi[$. l_{slip} is the total distance the feet slipped over the ground

table 3.4).

Typically each PSO was set to use a swarm size of 120 to 160 and 400 to 800 iterations, requiring 48000 to 128000 runs to fully complete. These runs were performed on a cluster of 157 nodes, made of of 87 2.0 GHz nodes with 1.6GB RAM each (22 Intel Xeon E5504 quadcore processors) and 70 2.2 GHz nodes with 1.3GB RAM each (14 Intel Xeon E5-2430 hexacore processors). Typically optimizations were completed in 5 to 20 hours in real-time.

3.2.5 Application to footstep modulation

3.2.5.1 Methodology

In order to investigate if this approach could be successful for dynamic footstep modulation, we propose a preliminary experiment. Given a gait resulting from a previous optimization, we would observe how a change of the L parameter will affect the footstep absolute position on the ground. The footstep of an optimized gait are recorded in our simulation. Then the simulation is re-run with the same parameters and pseudo random generator seed. Indeed, pseudo random generator are used both in the LCP solver used by Webots, and in our CPG model to avoid phase locking in a unstable fixed point of the network. With this approach we ensure strict repeatability of the simulation. On these new runs, at a random point in the gait cycle after the robot as reached its limit cycle, the L parameter is reduced from ≈ 12.6 cm by a fixed value between 2.5 mm and 4 cm. The sub-sequent footstep positions are then recorded and compared to the original run. Each of the step length variations is repeated at ten different point in the limit cycle. The results are shown in figure 3.5.

3.2.5.2 Discussion

The figure shows, as expected, a negative displacement of the footstep for larger step length variation. However smaller step variations could have a positive foot displacement for the first steps which is quite unexpected. One explanation for this behavior is that the CPG model

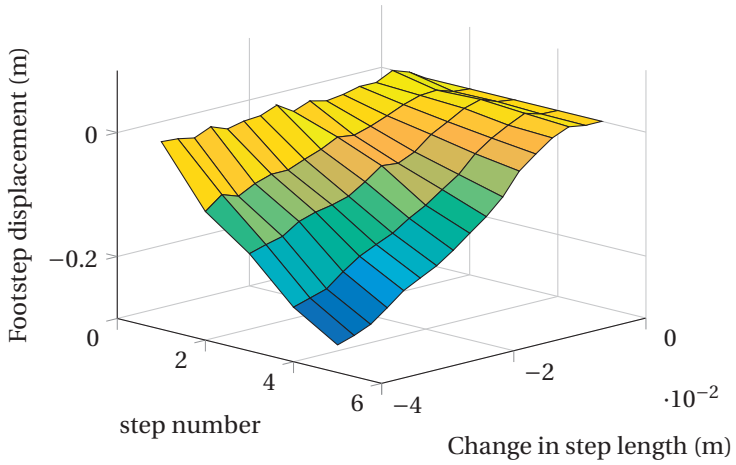


Figure 3.5 – Step length L modulation with open-loop CPG. Each steps change are triggered at a random point of the gait cycle and the experiment is repeated ten times for each variation. The surface shows the mean value of the footstep displacement. For very small step size variations, we see unpredictable behavior and even an increase of the footstep displacement even if the step length is reduced.

we choose has no feedback and is not synchronized with the dynamic of the robot. When we change the step length during the stance phase, the accurate trajectory tracking may add some energy to the robot that will be stored in the leg springs. At the end of the stance phase this energy will be released producing a small jump and therefore a positive leg displacement. Such an open-loop CPG does not manages any transient, and this could be a reason why it is hard to use as this in practice for dynamic footstep placement. Using a strong coupling between the mechanical and the CPG could eventually helps the management of this transient, and a promising methods is the one proposed by *Owaki et al.* [2012].

3.3 The Tegotae Rule: Bottom-Up Limb Coordination

3.3.1 Presentation

CPGs modeled by a network of coupled oscillators could be seen as a top down approach for inter-limb coordination. By example it can produce different gait, that can be classified using based on the phase differences between legs [*Alexander*, 1984]. A traditional approach with CPGs to produce a given gait, would be to enforce this phase difference between legs using oscillator coupling. This would correspond to an enforced neural coupling enforced between each smaller CPGs controlling each limb. However there is some limitation of such approach. First, animal does show transition from one gait to another. For horses the reason why these transition is still unclear as it could either results for metabolism cost optimization or reducing musculoskeletal peak forces [*Wickler et al.*, 2003]. It is the same in robot driven by coupled oscillators. Such gait transition could be easily triggered by changing the desired

phase coupling between oscillator, but how and when to trigger this changes could still be considered an open question. Furthermore, with compliant robots with complex mechanics such as cheetah-cub, even if we command the robot with strictly pure trot gait, the resulting gait diagram are not symmetric and there is a large difference between *desired* and *actual* duty factor [*Spröwitz et al.*, 2013].

The approach proposed by *Owaki et al.* [2012, 2013] is to remove completely this neural coupling between each limb CPGs, and replace it by a strong coupling between the dynamics of each leg and its local CPGs. Then inter-limb coupling will only be created by physical coupling. Such approach could be seen as a bottom-up approach for gait generation: *Owaki et al.* [2012] shows that lateral and diagonal walk pattern arise with the same control strategy from the robot morphology, by displacing the CoM of the robot between the back or the front of the trunk. Similarly *Owaki et al.* [2013] shows that pace and trot could arise from the same control strategy by lowering or increasing the height of the CoM of the robot. Furthermore this approach was successful to demonstrate walk to trot transition when the CPG model is asked to produce faster pattern.

We are interested to apply the same approach on our robots and see how effective this approach is successful for footstep modulation. It is worth to note that the work described in *Owaki et al.* [2012, 2013] is using robot with one Degree of Freedom (DoF) per leg, and therefore intra-limb coordination was not investigated. We will first present how this approach, that we name internally the Tegotae rule, could be applied to two or more DoF per leg re-using part of our aforementioned CPG architecture, and finally we will look how effective this rule could be applied to footstep modulation.

3.3.2 Application with intra-limb coordination

We will first present more formally the Tegotae rule first presented in *Owaki et al.* [2012, 2013], and then our own extension with fixed point compensation in order to apply it on a robot able to make large step, like Cheetah-Cub.

3.3.2.1 Mathematical derivation with bioinspired foot locus

Given the phase mapping described in figure 3.3, and the leg load estimation F, one can define a discrepancy function:

$$I(\theta) = \sigma F \sin(\theta) \tag{3.3}$$

Where σ is a simple scaling constant. This function express illogical states. For example, during the swing phase, i.e. $\theta \in [0;\pi]$ we expect the leg load to be null. If the leg is in contact with the ground $F > 0$ and therefore $I(\theta) > 0$. Similarly, during the stance phase ($\theta \in [-\pi;0]$), we expect $F > 0$ and therefore $I < 0$, if the leg would not touch the ground I will be null. To ensure a strong coupling between the leg and the oscillator, we would like to build a system

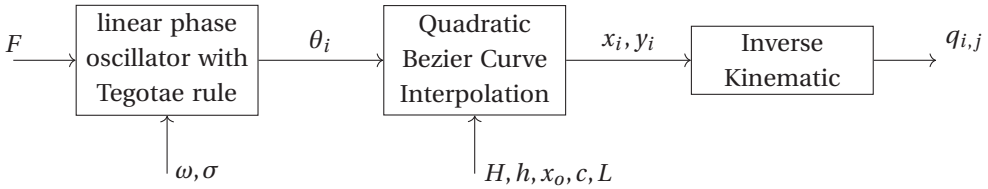


Figure 3.6 – CPG model using the Tegotae rule as feedback. The joint trajectories are computed from the leg load F.

that minimize this quantity. If we take a linear phase oscillator, it gives:

$$\dot{\theta} = \omega - \frac{dI}{d\theta}(\theta) = \omega - \sigma F \cos(\theta) \tag{3.4}$$

It is worth to note that because of the presence of F (that will fluctuate during locomotion), this oscillator will not display a linear phase. Compared to the approach described in section 3.2, there is no need to specify a desired duty ratio d. The duty ratio will emerge from the body dynamics itself. Instead of calling ω the pulsation of our oscillator, we call it excitation.

It is worth to note that this dynamical system shows a bifurcation if $\omega > \sigma F$ there is no fixed point of the system, but if $\omega < \sigma F$, two fixed point appear, one stable and one unstable:

$$\begin{cases} f(\theta) = & 0 \\ \dfrac{df}{d\theta}(\theta) < & 0 \end{cases} \iff \theta = -\arccos\left(\frac{\omega}{\sigma F}\right) \tag{3.5}$$

This nice behavior makes the robot motion to stop for one leg when the excitation is too low. This nice property also solves the problem of startup condition that we have with traditional CPGs, as for the robot to start from a standstill position, one just have to increase ω from 0 to the desired value.

To apply this rules to a robot with more than two DoF, we simply uses this non-linear phase as the Θ in our previous CPG architecture (see figure 3.6). This approach adds a new parameter to the controller (in comparison with table 3.3): the amount of feedback σ. On the other hand, we do not use the virtual duty ratio as different duty ratio emerge from the interaction of self-excitation ω, feedback amount σ and the dynamics of the robot.

3.3.2.2 Variable fixed point compensation

When we use the previous rules with robots able to achieve large leg retraction and protraction, such as Cheetah-Cub, we encounter static stability problems. When we look at the fixed point

value (3.5), it is not fixed to a constant phase value. When F approaches ω/σ:

$$\lim_{F \to \frac{\omega}{\sigma}} -\arccos\left(\frac{\omega}{\sigma F}\right) = 0 \tag{3.6}$$

which maps to the take-off position p_{to} (figure 3.2). If we use a large step length L, then this point may lay horizontal far away from the hip joint axis, putting the robot in an unstable position. This condition could arise during normal locomotion and could lead the robot to fall. We propose here to modify the original rule to ensure that the stable fixed is always $-\frac{\pi}{2}$, i.e. the mid-stance point which is at a fixed horizontal distance x_o from the hip. For that purpose we propose to use a dynamical system modified from (3.4):

$$\dot{\theta} = g(\theta) = f(\theta + x_c) = \omega - \sigma F \cos(\theta + x_c) \tag{3.7}$$

With x_c a compensation term. To ensure the stable fixed point is always $-\frac{\pi}{2}$, we need to solve:

$$\begin{cases} g(-\frac{\pi}{2}) = & \omega - \sigma F \cos\left(x_c - \frac{\pi}{2}\right) = 0 \\ \frac{dg}{d\theta}\left(-\frac{\pi}{2}\right) = & \sigma F \sin\left(x_c - \frac{\pi}{2}\right) < 0 \end{cases} \iff x_c = \arcsin\left(\frac{\omega}{\sigma F}\right) \tag{3.8}$$

To extend it to case where $|\omega| > |\sigma F|$, we just use the constant $\pm\frac{\pi}{2}$ as boundary conditions:

$$x_c(\sigma, F, \omega) = \begin{cases} \arcsin\left(\frac{\omega}{\sigma F}\right) & \text{if} \quad |\omega| < |\sigma F| \\ \text{sgn}(\omega\sigma F)\frac{\pi}{2} & \text{else} \end{cases} \tag{3.9}$$

It gives this formulation for our dynamical system in the case $|\omega| < |\sigma F|$:

$$\dot{\theta} = \omega - \sigma F \cos\left(\theta + \arcsin\left(\frac{\omega}{\sigma F}\right)\right) \tag{3.10}$$

With this novel Tegotae rule, we could practically apply this novel approach to Cheetah-Cub.

3.3.2.3 Emerging gait transitions

To validate our approach, the new CPG architecture was tested in simulation. However, instead of optimizing the excitation ω value, it is defined to increase every 10 s by a step of $\frac{\pi}{2}$, from 0 to 8π (corresponding to 0 Hz to 4 Hz). The PSO was again used to optimize the foot locus parameters and this time the feedback parameter σ. The fitness function was left unchanged, and the optimization was repeated several times.

All optimized gait showed a very nice property: at some point in time a gait transition was emerging as shown in figure 3.7. The robot is changing from a trot gait, where diagonal leg

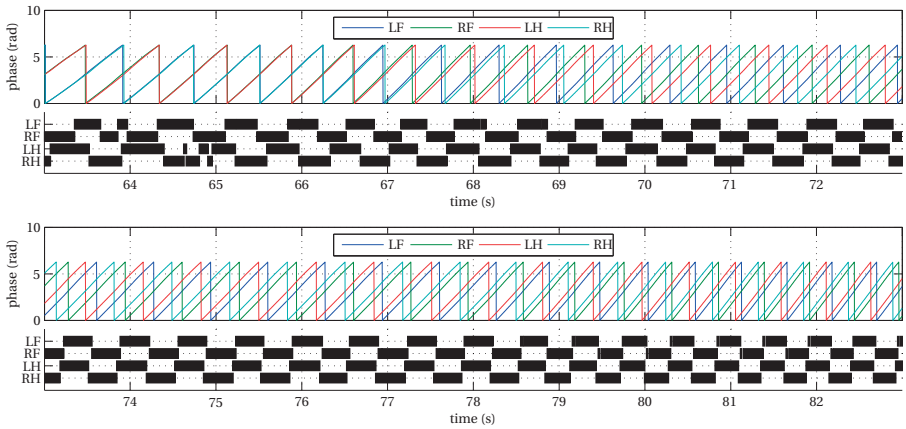


Figure 3.7 – Transition from trot to pace. At t=63 s, the increase of the ω excitation triggers a gait transition from pace to trot, which after a period of large foot slipping (64 s to 66 s) establishes to a regular pace at $t \approx 70$ s. In the range of 70 s to 74 s the phase pattern is similar to a walk, but the footfall pattern is one of a pace.

are synchronized, to a pace gait, where lateral limbs are in phase, the transition taking a few seconds. Such a transition is emerging here form our control strategy, and would have been in practice be hard to produce if only strong coupling between phase oscillator would have been used.

3.3.3 Application to footstep modulation

The experiment described in section 3.2.5 has been reproduced with our new CPG model and results are reported in figure 3.8. The result are much nicer than in the previous case (figure 3.5) as even small step length decreases makes a negative footstep displacement as soon as the first step after the change. Furthermore the relation between the step length changes, the number of step taken after the change and the footstep displacement seems to be linear. Such an approach would be more suitable for dynamic footstep placement to avoid an obstacle (figure 3.1).

3.4 Conclusion

In order to provide footstep modulation for anticipatory control of legged robot trajectory, the original CPG model used for the Cheetah-Cub and Oncilla robot has been modified to use a bioinspired foot locus parameterization. This novel approach, made all the parameters of the low-level controller represents quantities in the task space of the robot. This enabled us to easily setup the measurement of the footstep displacement when we modify the desired step length parameter.

However an open-loop CPG seems unsufficient to perform dynamic footstep modulation

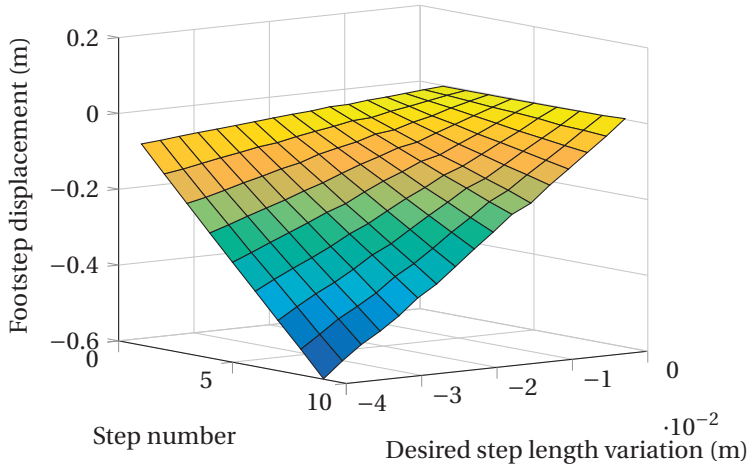


Figure 3.8 – Step length variation comparison.

in simulation. As shown by figure 3.5, the displacement of the footstep is erratic. We assume that the open-loop CPG lacks the ability to control the transient trajectories between two limit cycles, and it takes several steps to return to a limit cycle after the change.

In comparison, an approach with a strong sensorimotor coordination, the Tegotae rule, is much more successful for modulating the footstep placement. Our main contribution is to adapt the original rule to Cheetah-Cub, by forcing the fixed point of the CPG model (after bifurcation) to have a constant value corresponding to the mid-stance timing. Regarding dynamic footstep placements, figure 3.8 shows that the displacement occurs as soon as the next step after the parameter change. The footstep displacement is also both linearly related to the number of steps taken and to the variation of the step length parameter. Such a relation would be easy to use in any higher-level controller to change the position of the next footstep given their current estimation. This footstep modulation experiment is only considering change of the step length by the same amount for all of the legs. However, to avoid a real obstacle in front of the robot, such as an hole, the footstep may need to be displaced at a different distance for each of the leg. We should test next if the CPG model we propose is still able to manage the transient states that are required in this more complex situation.

4 Leg Load Estimation with Tactile Sensors

4.1 Introduction

With the promising result on a simulated model of the Tegotae approach, we would like now to implement it in the Cheetah-Cub to see if the results remains the same with the real robot. However Cheetah-Cub was not originally designed with any sensors. The Oncilla robot was an attempt to build a sensorized version of Cheetah-Cub, and features three axis force sensors on each foot. However the added mass of the sensors and Brushless Direct Current (BLDC) motor driving electronic, made this robot less suited for dynamic gaits than the smaller Cheetah-Cub. The optimized gaits found on Oncilla have very small stride lengths, which makes the measurement of footstep displacement quite difficult in practice.

There have been previous attempt to equip Cheetah-Cub with force sensing on each leg. First by using strain gauges on the distal segments. These were very fragile and were quickly destroyed during manipulation of the robot. Furthermore, building a protective envelope around them to protect these gauges would have put too much inertia on the legs.

Proximal sensing was also considered and implemented on Cheetah-Cub. But for highly dynamical gaits, there is an mechanical low-pass filtering of the Ground Reaction Forces (GRFs) due to the compliance and damping in the leg. We would like to have these sensor distally mounted.

One last attempt was using commercial tactile force sensors on the sole of the foot to estimate the normal of the GRFs. However these sensors did not yield good results, and in practice could only measure boolean information, i.e. if the foot touches the ground or not.

We aim in this chapter to provide Cheetah-Cub with our own design of a tactile sensor, in order to be able to estimate the mechanical load on each leg. We will first list the desired speci-

fications for such a sensor, then present a brief review of available tactile sensing technologies. Then we propose our own piezoresistive based sensor design. Finally due to overcome the limitation of this sensor, we propose a novel approach by fusing its information with a dynamic tactile sensor using an Extended Kalman Filter.

4.1.1 Desired sensor specifications

The specifications of the sensor derives from the work on the Cheetah-Cub robot and the research interests in our lab on quadruped locomotion, and especially dynamic gaits. Dynamic similarities between two different legged systems, especially running systems, can be qualified using two dimensionless numbers: the Froude number ($F_r = v^2/gl$), and the Strouhal number ($S_{tr} = fl/v$) [*Alexander*, 1989; *Delattre and Moretto*, 2008; *Villeger et al.*, 2014]. Here v is a characteristic speed of the system (i.e. the speed of the CoM), g the earth acceleration, l a characteristic length (i.e. the height of CoM), and f a characteristic frequency (i.e. the stride frequency). These numbers could be seen as respectively the ratios between the importance of the Inertia forces over the Gravitational ones, and between the Elastic Forces and the Inertia ones. They are used to characterize dynamic aspect of the locomotion, and relate different gait used by different animals, independently of the scale. Therefore any hardware modification should have a very small impact on these two numbers.

4.1.1.1 Low inertia impact

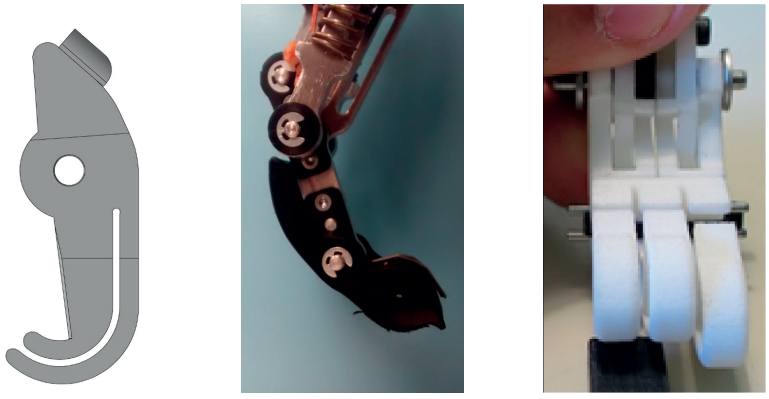
As mentionned in section 2.1.2 one of the key property for ASLP legged robot is to keep the inertia of the leg as low as spossible. Therefore the sensor should add as little as possible weight on the toe segment. Another consideration is the overall impact of the sensors on the total robot weight. Another formulation of the Strouhal number, using the characteristic frequency of a spring mass system ($f = \sqrt{k/m}$)

$$S_{tr}^2 = \frac{f^2 l^2}{v^2} = \frac{k l^2}{m v^2} \tag{4.1}$$

One can see that the mass of the robot has an impact on the dynamicity of the gait it could achieve. Therefore the sensors and their amplification circuitry should also increase as little as possible the total robot weight.

4.1.1.2 Response of the sensor

The product of the Strouhal and Froude number ($F_r S_{tr}^2 = f^2 l/g$) show that similar gaits, for smaller animal, would shows a larger stride frequency. In this case, Cheetah-Cubs robots typically have stride frequencies for running gait around 3 Hz and stance phase duration that could goes down to 85 ms. The sensor must be able to measure load changes over this



(a) Cylinder (b) Claw (c) Adaptive

Figure 4.1 – Different foot designed for the Cheetah-Cubs robots: (a) original cylinder shape, (b) claw shaped and (c) adaptive compliant toe shaped. Designing our own customized sensor would make us able to cover all of these foot shapes.

relatively short period. Furthermore, the aimed control approach (see chapter 3) does not rely on a precise estimation of the load, but rather one the load differences between the legs, from standstill (static load) to the dynamic case stated above. Therefore nonlinearities in the sensor response are acceptable, but we will require a good sensitivity of the sensor to measure those load differences. Finally *Spröwitz et al.* [2013] shows that during steady state locomotion, the load on legs could build up to 13 N, and to ensure the possibility of measuring extraordinary conditions, the sensor should measure load up to 20 N.

4.1.1.3 Adaptability to different shape

As depicted in figure 4.1, the sensor could be used on a variety of different foot shapes. Furthermore, empirical observations, shows that the contact could be made either on the soles of the feet or, at a certain point of the stance phase, on the front tip of the foot. Therefore such load sensor would need to be flexible to cover the whole shape of the foot or toes. The variety of foot design makes the use of commercial solutions much harder, as they limit the possible achievable design for the foot and the toe.

4.1.1.4 Requirements for leg load estimation

A summary of the desired sensor specifications can be found in table 4.1.

4.1.2 Available tactile transduction

Any tactile sensor measures the deformation of a known material under the effort that needs to be estimated. The monitoring of physical quantities such as electrical capacitance, resistance charge or optical properties of these materials to estimate these deformations is called trans-

Specification	Value
Load Range	0 to 20 N
Frequency Response	> 4 Hz
Sensitivity	< 200 mN
Sampling Frequency	< 10 ms
Weight on the Tip	< 2 g
Total weight on the robot	< 20 g

Table 4.1 – Leg Load Sensor Specification.

duction of the contact information. Pro and cons of several types of different transduction are presented below, and a summary of these descriptions could be found in table 4.2. A more complete review could be found in *Kappassov et al.* [2015].

4.1.2.1 Piezoresistive sensor

Piezoresistivity is the change of electrical resistance of a material when it is subject to a deformation. Material displaying this property exists in many different forms, such as rubber, fabric, plastic sheet or foam. Force Sensitive Resistor (FSR) based on piezoresistive material are often found in Human Interaction Devices, such as Joysticks [*Dahiya and Valle*, 2008]. One of the key advantages of such sensors is that they are easy to manufacture, and therefore many commercial solutions are available, such as Interlink FSR [*Interlink*, 2016], Tekscan flexiforce [*Tekscan*, 2016], Weiss Robotics[*Weiss Robotics Gmbh*, 2016], Eeonyx[*Eeonyx*, 2016]. Compared to other transduction types they are more robust to electromagnetic noise.

However piezoresistive sensors suffer from serious drawbacks. When elastic rubbers are used as piezoresistive material, their elasticity can introduce large hysteresis. They often present large nonlinearities in their responses (see calibration plot of *Tekscan* [2016]). Finally material used in the constitution of the sensor can change properties due to moistness or temperature [*Fraden*, 2016].

It is worth to note that the FSR provided by Interlink are not a direct application of the piezoresistive effect. Indeed, in order to improve their response, they use a combination of piezoresistivity and mechanical deformation of a single electrode layer to transduct contact forces: the two electrodes are placed on the same layer, and separated from the piezoresistive material with an electrically isolated annular spacer. Contact forces in the center of the sensor deform that layer and create contact between electrode when they touch the piezoresistive layer. The drawback is that if the external contact surface is bigger than the FSR itself, all contact forces go through the spacer, and there is no deformation of the electrode layer. Therefore there is no contact and the resistance remains infinite. This is certainly a reason why these FSR are more used as internal part of a device, and less as a direct exterio-receptive sensor, where the contact surface could not be easily controlled.

Transduction Type	Sensitivity	Non-linearities	Frequency Response	Dynamic Sensing	EMI Robustness	Simplicity
piezoresistive	+	−−	−−	−−−	+	+++
capacitive	++	−	+	+	−−−	++
piezoelectric	+++	+++	+++	+++	−	−
optical	+++	+++	+	+	+++	−−−

Table 4.2 – Advantages and disadvantages of different types of transduction, summary of the review of *Kappassov et al.* [2015]

4.1.2.2 Capacitive sensors

Capacitive sensors are constituted of two conductive plates separated by a dielectric material. When mechanical efforts are applied to the sensor, distance between the plates is reduced, and the capacitance increases. By varying the thickness or changing the compressibility of the dielectric material, higher sensitivities could be achieved. This approach is very popular in robotics, and has been applied in the iCub tactile skin, PR2 grippers or the Allegro hand. Integrated Circuit (IC) for measuring small capacitance, such as the AD7147 from Analog Device, helps the integration of such sensors in new projects.

Their main advantages over piezoresistive sensors are their larger frequency response, and the widespread use of this technology in every day life application (for instance in touch screens). However they are more prone to electromagnetic noise as the exact number of charges in the sensor has to be estimated. Furthermore they still present nonlinearities and hysteresis [*Maiolino et al.*, 2013].

4.1.2.3 Piezoelectric sensors

Piezoelectricity is the electrical charge generation that occur in certain crystalline material caused by deformation due to applied forces or pressure. This effect could be seen in quartz, or human-made ceramics and polymers, such as Polyvinylidene fluoride (PVDF) [*Seminara et al.*, 2011] and Lead Zirconate Titanate (PZT) [*Acer et al.*, 2015]. Due to the nature of the piezoelectric effect, sensors based on this transduction are only reserved for measuring dynamic forces, i.e. fast changing forces such as vibration or sudden changes [*Cutkosky and Ulmen*, 2014]. This is due to the fact that if a static load is applied to such material only a fixed quantity of electrical charges would be generated by the material. Any electronic amplification circuitry would need to continuously consume these charges to estimate them. In a static scenario, these charges would ultimately be depleted no more signal could be read.

In order to deal with these limitations *Sonar and Paik* [2016] proposed to couple this sensor with Soft Pneumatic Actuator (SPA) for vibrotactile feedback. Here the actuation is used as both a feedback mechanism for human device interaction, but also as a source of dynamical forces, making the piezoelectric sensor always measure dynamic forces, even if

only static forces would be applied to the sensor. *Sonar and Paik* [2016] shows how external forces could be estimated using peak detection of the signal generated by the PZT material.

The main advantages of piezoelectric transduction is their frequency bandwidth that can range up to 7 kHz, however they are still sensitive to temperature, and limited to dynamic measurement only.

4.1.2.4 Optical sensor

Optical sensors are based on measuring how a material light reflection is affected by its mechanical deformation. One technique would be to use two materials with different refraction indices, traversed by a a beam of light emitted by a light. A photo-diode would measure the scattered light through the material. Any external pressure would modify the amount of scattered light in the material, and that changes would be measured by the photo-diode.

Another approach is using a reflective material of a known shape, such as cylindrical or hemispheric, and to measure how the reflections of light beams emitted by a LED are affected by the deformations of this shape. Sensor based on this technique are commercialized by OptoForce [*OptoForce*, 2016].

The main advantages of these type of sensor are their spatial resolution, sensitivity, high repeatability, immunity to electromagnetic noise, range of forces that could be measured (from a few newtons to thousand of newtons). However they are relatively big and have high-power consumption as they should perform complex processing in their package.

4.1.2.5 Multimodal approaches

There exist several approaches in robot tactile sensing that combines several of the transduction types presented here. This is to reflect the different human modalities of tactile sensing, where both static and dynamic force can be sensed. The most common is to use a combination of piezoresistive sensor for static force estimation and piezoelectric one for dynamic forces detection [*Göger et al.*, 2009; *Kawamura et al.*, 2013]. In the former, the static sensor is used to determine the force distribution over the sensor while the dynamic one is used to detect vibration caused by slippage. The main drawback of these kinds of sensor is their relatively bigger size. Also our application requires the global load estimation on the leg rather than the distribution of pressure on the foot and slipage information.

4.1.3 Sensor choice

From table 4.2, there is no sensor transduction type that is perfectly suited for the load estimation. Piezoresistive sensors and capacitive sensors may show hysteresis and nonlinearities that would make it hard to estimate load differences between legs near their saturation force.

Optical sensors are good candidates, but they are an high end technology, so only commercial solutions could be considered, which could restrict a lot the type of foot they could equip. Finally dynamical forces and vibration sensing are not the priorities for legged locomotion control.

Therefore we would like to explore a multimodal approach where we would combine a static force sensor and a dynamic one, and rely on data fusion techniques to produce an accurate estimate of the load on the leg. For the dynamic force sensing parts a piezoelectric sensor will be used, such as the one described in *Sonar and Paik* [2016]. On this sensor the piezoelectric cell is constituted by a small PZT crystal that may be repeated in an array to cover a large surface (see figure 4.6a), thus it could easily be adapted to any foot shape. For the same reason, we decide to build our static sensing sensor using piezoresistive transduction as they are the simpler to operate.

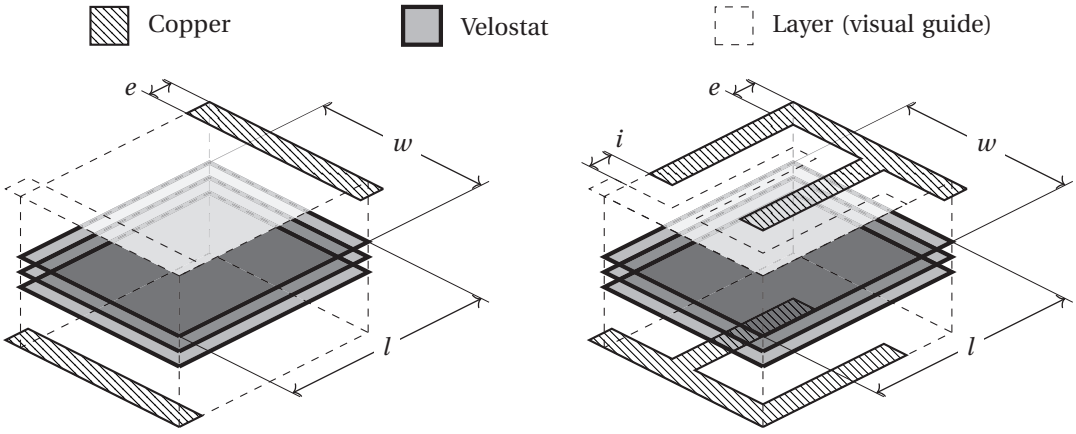
The next section will describe an empirical validation of the static force sensor design, and the last one will describe how such data fusion can be implemented.

4.2 Validation of the Piezoresistive Sensor

The material retained as a piezoresistive element is the Velostat, a plastic that is normally used for making Electrostatic Discharge (ESD) protection bag for electronic devices transport and stockage. For this particular usage, its main characteristic is its high surface contact impedance. Its piezoresistivity is a side effect and his not documented in any way by its manufacturer. One of its advantages is that it is really easy to acquire and manufacture as it comes in form of sheets. One drawback is that it is not easy to model the piezoresistivity of this material, therefore only an empirical validation could be done. We propose here to use Design of Experiment (DoE) techniques to screen for the important factors that influences the sensor characteristics, and build a linear model that would predict if the usage of such sensor is appropriate or not for any given goal.

4.2.1 Proposed design

The structure of the sensor is quite simple, as it is made from one or several layers of Velostat, with two copper electrodes on each side. For the electrodes, sheets of adhesive copper are used. The glue used on these sheets is electrically conductive. The resulting sensor is therefore flexible and can adapt to any shape. To simplify this study, only rectangular shaped sensors are considered. As only piezoelectric transduction should be considered, the electrodes are designed not to overlap over each other. Indeed preliminary experiments using simpler electrode design covering the whole Velostat area showed that there was a non-negligible capacitance between those. To avoid that effect, two different kinds of electrodes are proposed. The first one (type A, figure 4.2a), is simply to put on two opposing sides of the cell a thin band of copper. The second one, (type B, figure 4.2b) is to make a complex shape of zebra



(a) type A electrodes (b) type B electrodes

Figure 4.2 – The two types of considered piezoresistor design. Here three Velostat are represented. In this represenation, the cell are measuring vertical forces. w (cell width) l (cell length), e (electrode width) and i (type B electrode distance) are parameters of our model. In this study e is fixed to 2 mm and i to 1 mm has the sensor are hand manufactured.

that covers most of the area of the cell. The B shape avoids overlapping but covers the whole area, and hopefully leads to a higher sensitivity, however the A shape is far more simpler to build. These difficulties in the building process are not negligible at all and it will be a great advantage if sensors built using this shape could meet the desired specifications.

These two electrode shapes also share the same design parameters (figure 4.2): l is the size of the cell in the direction that separates the two main conductors, w is the length of those main electrodes and e is the electrode thickness. One would note that the width of the type B cell can only take discrete values depending on the electrode width e and the number of bridges.

The acquisition electronics for this sensor is constituted of an non-inverting voltage amplifier of a fixed voltage reference (figure 4.3), as proposed in the FlexiForce datasheet [*Tekscan*, 2016]. The variable piezoresistor is placed as the non-feedback resistor and changes the gain of the amplifier :

$$V_O = \left(1 + \frac{R_F}{R_{PZR}}\right) V_{REF} \tag{4.2}$$

So the value of V_O will increase with the applied load as the resistance R_{PZR} is decreasing. The tuning of this circuitry requires to choose the value of the reference voltage V_{REF} and the feedback resistor R_F. The linear model resulting of the screening is expected to give insight on how to adequately choose these two values. However for the screening itself, and to ensure more accurate measurement, a standard multimeter has been used (see figure 4.5).

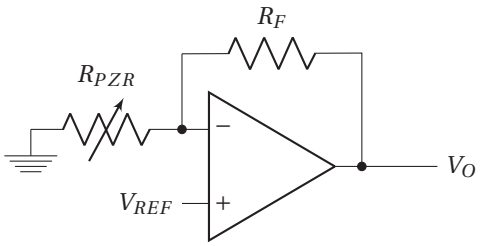


Figure 4.3 – Amplification circuitry for the piezoresistive sensor. It is made of a voltage amplifier of a fixed voltage reference (V_{ref}) and whose gain is changed by the varying piezoresistor R_{PZR}.

4.2.2 Screening of relevant factors in the sensor design parameters

4.2.2.1 Design of Experiment

The goal is to build a model that from the design parameters of the sensor, will predict what its conductance versus load characteristic is:

$$C = g(F, \vec{\alpha}) \tag{4.3}$$

Where C is the conductance of the sensor, F a perticular load applied to the sensor and g is a parametrized function with $\vec{\alpha}$ parameters. The choice of g and $\vec{\alpha}$ will be detailed in section 4.2.2.1. Our goal here is to determinehow those parameters $\vec{\alpha}$ depends on the structure and environment of the sensor. Building such a model that could depends on a large number of factors could be a tedious task and if not conducted carefully, lead to unreasonably large number of experiments to draw a conclusion that may possibly in fine not be statistically relevant. Therefore this screening of the importance factor is carefully conducted by choosing an appropriate DoE.

Choice of factors As factors, five structural parameters are retained: The cell type (A or B, see section 4.2.1), the electrode length l, its width w, the number of Velostat layers between the two electrodes, and the use of conductive glue. As the cells are manually manufactured, a relatively large electrode width e of 2 mm and a minimal distance between electrode of 1 mm are used. For the same reason the lower bound for w was chosen to be 5 mm. In order to have to have a different topology between type A and type B, the l factor is required to be larger than 8 mm. If sensors with a single layer are not tested, which require to have electrode to be glued on both sides of the same piezoresistive layer, it is easy to modify any cell by adding and removing Velostat pieces between two layer with fixed electrode on one face. By avoiding to build a specific sensor for each situation the manufacturing time could be divided by two as it will be furtherly shown. The use of conductive glue between the copper electrodes and the Velostat could also be an interesting effect to test, and therefore the presence of glue is retained as a factor. Without glue, the electrode contact will be maintained using electrically

Objectif	Determine how design parameters of the piezoresistive cell influences the resitance versus load response	
Factors		
Continuous	Length	8 mm — 29 mm
	Width	5 mm — 29 mm
	Temperature	5 °C — 50 °C
Discrete	Number of Velostat Layers	2 — 6
	Presence of glue	0 — 1
	Electrode Shape	A — B
Responses		
	Cell sensitivity	$\Omega^{-1}\,N^{-1}$
	Base Conductance	Ω^{-1}
	Saturation Force	N
Model	Additive with first degree interactions between factor	
Strategy	Fractionnal Factorial 2^{6-2}_{IV}	

Table 4.3 – Summary of the DoE for the characterization of the piezoresistive cell design.

isolating tape. The last factor chosen is an environmental one: the temperature at which the cell is used. Here values that could be easily encountered in outdoor conditions are retained. Thus the experiments is made of three discrete (Shape, Glue, Number of Layer) and three continuous (Length, Width, Number of Layer and Temperature) factors.

Choice of responses From prior knowledge and the Tekscan datasheet, the change of the resistance of the cell under load is expected to be nonlinear. However as mentioned by the Tekscan datasheet, the conductance of the cell changes linearly under the load. Preliminary measurement identified that there is also a saturation of the conductance, where above certain loads, we do not measure any decrease of the resistance.

Such a nonlinearity makes it difficult to fit a model directly on the conductance with points only on the boundary of the tested range, which is the case using orthogonal design matrices. Furthermore nonlinear regression makes it difficult to compute F-value or p-value, as it is unclear what should be the null hypothesis of a nonlinear model. Therefore we suggest, rather than using load on the cell as a factor and the resistance as the response, to experimentally measure for each cell its conductance versus load characteristics, and then use nonlinear regression to fit a linear function with saturation, defined by :

$$G(F) = \begin{cases} 0 & \text{if } a_1 F + a_2 < 0 \\ a_1 F + a_2 & \text{if } 0 \leq a_1 F + a_2 \leq a_3 \\ a_3 & \text{if } a_3 < a_1 F + a_2 \end{cases} \tag{4.4}$$

a_1 is the sensitivity of the cell. a_2 is the base conductance, i.e. the conductance under no

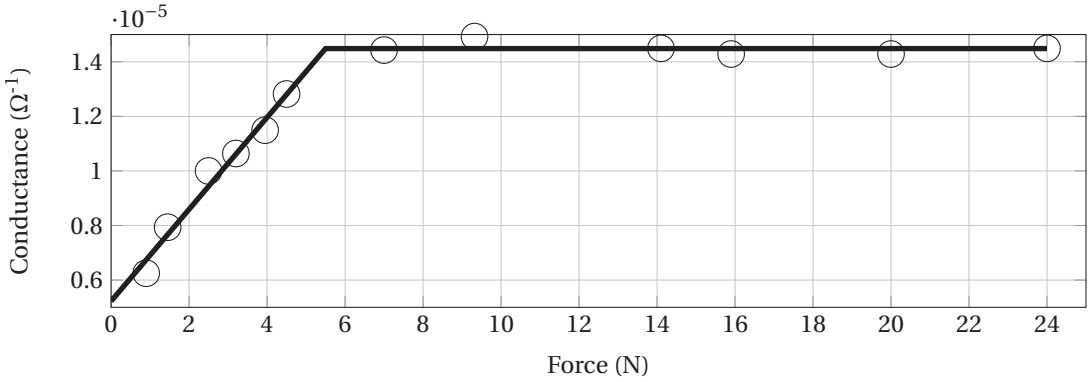


Figure 4.4 – Characteristic of the cell A WG L2 8×29, and its fitting ($R^2 = 0.9994$).

load. Please note that we allow it to be negative, i.e. that a certain load is required in order to have a positive conductance of the cell. finally a_3 is the saturation conductance, which is related to the saturation force :

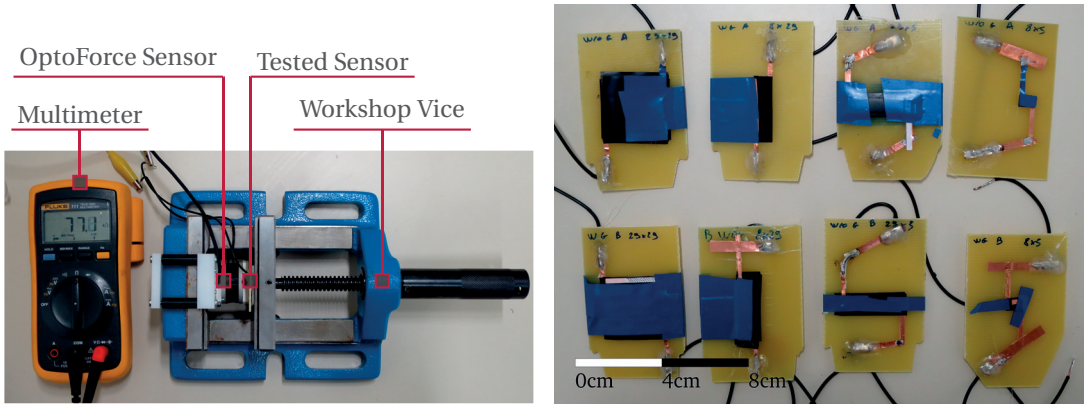
$$F_{sat} = \frac{a_3 - a_2}{a_1} \tag{4.5}$$

As an illustration, figure 4.4 shows the results of the nonlinear fitting for one of the cells. The parameters of the fitted curves are used as the responses of our model, since it is more likely that these parameters would be linearly dependent on the chosen factors.

Choice of model Preliminary experiments showed that the characteristic of the cell would change with its area. Therefore the model retained is an additive model with first order interactions between factor, to validate this hypothesis.

Strategy Since the manufacturing and characterization were thought to be costly experiments in terms of time, and moreover since there are quite a number of factors, a Full Factorial design should be avoided, as it would result in the manufacturing and characterization of 64 different cells. This could be reduced using a fractional design. Since interaction have to be considered in the model, we needed at least a design of resolution IV or ideally V to avoid any confusion of the interactions with the main factor [*Box et al.*, 2005, Chapter 5]. Using a design of 2^{6-2}_{IV} the number of characterizations is reduced to 16, which is more time affordable.

Ordering of factors By using a resolution IV design, some of the interactions between two factor will be aliased or confused, i.e. there will be no statistical way to estimate how independently they affect the model, but only how any unknown linear combination of the two



(a) Experimental setup (b) Manufactured cells

Figure 4.5 – Experimental setup for cell characterization. (a): a 3 axis force sensor (OptoForce OMD-30-FF-1600N) is mounted on a workshop vice what allows us to precisely measure the load applied by the vice to a cell. The resistance of the cell is measured by a Fluke 111 multimeter. (b): manufactured cells glued on FR-4 plates to minimize undesired mechanical efforts applied by the connection cables.

affect our model. [*Box et al.*, 2005, Chapter 5] Therefore, by examining the confusion matrix (i.e. how those interactions are aliased), we will have to try to alias the area of the cell with some interactions hypothesized to be less likely to occur.

When using fractionnal factorial design, not every combination of factor are tested. Only a subset of factors are chosen to have every combination tested. For the remaining factors, the choice of the tested values are generated from the other factors using generators. The factor generators for the 2^{6-2}_{IV} are not symmetric. If the factors are named $\{A, B, C, D, E, F\}$, and the generator chosen to be $E = ABC$ and $F = BCD$, A and D are used only once, and B and C in both. Two of the factors (Number of Layers and Temperature) are easy to change without manufacturing a whole new cell. By choosing these two factors as B and C, the number of cell to be manufactured can be reduced to 8 instead of 16, as shown in table 4.4.

Considering these two facts, the order of factors is chosen to be: A: Glue, B: Number of Layers, C: Temperature, D: Shape, E: Length, F: Width. This choice aliases the area of the cell with the interaction between the presence of glue and the shape of the cell, which intuitively seems less likely to play an important role in the final model.

4.2.2.2 Experimental setup

In order to measure the conductance versus load characteristic of a cell, a small test bench was developed (figure 4.5). It uses a small workshop vice to apply the load on a cell, and a 3 axis force sensor (OptoForce OMD-30-FF-1600N) to measure the forces applied. To measure the resistance of the cell, we used a Fluke 111 multimeter. The cells are fixed on a 1.6 mm thick FR4 plates to ease their manipulation. Connection cables are directly soldered on the plate, to

minimize the undesired mechanical effort they transmit to the cell.

The advantage of this setup is to be relatively small, so that it could be easily placed in a fridge or in a small workshop oven to change its environmental temperature. In practice, in order to precisely characterize a cell, this setup showed some disadvantages. First, the load applied by the vice can be several order of magnitude higher than the higher considered load, by simply turning the handle of half a turn. Therefore it makes the backslash in the lead screw hard to experimentally deal with when a particular smaller load is desired. Furthermore, as the vice is too loosely tighten, the lead screw tends to slowly unscrew itself with time (in the order of a Newton per minute) which makes accurate measurements harder to achieve. Finally, the mechanical stress on the plate transmitted by the cables is still readable by the cell and moreover the force sensor.

Accordingly, these difficulties increased a lot the experimental noise and the time needed to measure the characteristic of a cell, from an expected few minutes to more than an hour per cell. Furthermore it was not possible to handle the setup in close space as initially expected, and all manipulations had to be performed at room temperature.

From a statistical point of view, removing the temperature factor does not make the design worse, because the dispersion matrix is still orthogonal. With only five factors a 2_{V}^{5-1} design could have been used, which would have improved the resulting model. However in that situation, the optimization described in section 4.2.2.1 is not valid anymore, and 16 cells should have been manufactured. Therefore the design was kept.

4.2.2.3 Results

The parameters of the 16 nonlinear regression fitting on the idealized characteristics are shown in table 4.4. All detailed fittings are presented in appendix B To discriminate between statistically relevant and irrelevant factors, usually normal plot are used. However, due to aliasing between interactions, our design has a relatively small number of factors combination: 12 including identifiable interactions. Therefore normal plots are not well-suited to identify statistically dominant factors, as each normal plot had a bad line fitting, and it was only obvious to isolate one or two factors, which was either the constant and the type of cell, which in turn does not produce any useful model.

A more empirical approach was used instead. An ANOVA analysis was performed considering all possibles interactions, and iteratively the factor with the highest p-value was removed, until all factors or interactions fell down to a p-value approximately under 10 %. The final models and their ANOVA analysis can be found in table 4.5.

Cell sensitivity The shape of the cell is the most important parameter for the cell sensitivity. The electrode of type B seems to increase the sensitivity of the sensor. Moreover, the glue has a negative effect on the sensitivity so by its presence, the conductivity has a less steeper slope.

Factors										Responses		
Glue		Layer		Shape		Length		Width		Sensitivity	Base Conductance	Saturation
∅	∅	∅	∅	∅	∅	∅	mm	∅	mm	$\Omega^{-1} \cdot N^{-1}$	Ω^{-1}	N
+1	+	+1	6	+1	B	+1	29	+1	29	1.3982e-5	2.9882e-5	26.919
+1	+	+1	6	-1	A	+1	29	-1	5	1.1016e-6	7.8661e-6	14.358
+1	+	+1	6	+1	B	-1	8	-1	5	1.2232e-6	9.0181e-6	16.719
+1	+	+1	6	-1	A	-1	8	+1	29	5.5699e-7	1.0786e-5	54.284
+1	+	-1	2	+1	B	-1	8	-1	5	1.3968e-7	2.8418e-6	39.206
+1	+	-1	2	-1	A	-1	8	+1	29	1.6838e-6	5.2261e-6	5.4990
+1	+	-1	2	+1	B	+1	29	+1	29	1.3655e-5	3.9793e-5	23.556
+1	+	-1	2	-1	A	+1	29	-1	5	4.1681e-7	2.0652e-6	20.194
-1	−	+1	6	+1	B	-1	8	+1	29	1.8126e-5	9.8095e-5	69.953
-1	−	+1	6	-1	A	-1	8	-1	5	3.4456e-6	-3.8753e-5	25.433
-1	−	+1	6	+1	B	+1	29	-1	5	5.5612e-6	6.1428e-5	70.625
-1	−	+1	6	-1	A	+1	29	+1	29	3.3149e-6	1.4522e-5	8.0650
-1	−	-1	2	+1	B	+1	29	-1	5	9.7197e-6	1.5654e-5	31.604
-1	−	-1	2	-1	A	+1	29	+1	29	5.4386e-6	4.7005e-6	3.5070
-1	−	-1	2	+1	B	-1	8	+1	29	1.3452e-5	2.4862e-5	34.412
-1	−	-1	2	-1	A	-1	8	-1	5	7.3956e-7	-5.0260e-6	19.597

Table 4.4 – Experimental results of the DoE. Due to practical issues the temperature factor is removed. If the layer column is ignored, each combination of factors appears twice, thus the number of cell to be manufactured is 8.

It may be explained by the fact that at low force, the contact between the electrodes and the piezoresistive material is less enforced and thus the conductivity lower. Width (or electrode length), does have a positive impact on the sensitivity, but length seems to have no effect alone. Furthermore, there is apparently no identifiable effect of the area on the sensitivity.

Unloaded conductance The shape is again the most influential factor for the unloaded conductance. It could be easily explained as, for the first shape, the shortest distance (if we consider the thickness of Velostat negligible) is $w - 2e$, whereas in the type B cell, this distance is i. Since the current path are shorter, they experience less the resistance of the Velostat, which explain that the effect of this factor is almost twice larger than the others.

The other two relevant factors are related to the dimension of the cell: the width and the area. The first has a positive effect and the second a negative one. It could be explained under the assumption that the Velostat constitutes an infinite network of resistance between the two electrodes. By increasing the width, more resistors in parallel are added to the network, and decreases the total resistance, but increasing the length (with the same width) adds more resistors in series to the network, and increases the total resistance.

Factor	coeff.	SumSq	DoF	MeanSq	F	pValue
a_1: Cell Sensitivity (p-value vs Constant model: 7.85e-06)						
Constant	5.785×10^{-6}	·	·	·	·	·
Glue	-1.690×10^{-6}	4.569×10^{-11}	1	4.569×10^{-11}	10.93	7.008×10^{-3}
Shape	3.698×10^{-6}	2.188×10^{-10}	1	2.188×10^{-10}	52.32	1.68×10^{-5}
Width	2.992×10^{-6}	1.432×10^{-10}	1	1.432×10^{-10}	34.23	1.106×10^{-4}
Glue * Length	2.330×10^{-6}	8.689×10^{-11}	1	8.689×10^{-11}	20.78	8.192×10^{-4}
Error	·	4.600×10^{-11}	11	4.182×10^{-12}	·	·
a_2: Unloaded Conductance (p-value vs Constant model: 0.00719)						
Constant	1.769×10^{-5}	·	·	·	·	·
Shape	1.751×10^{-5}	4.907×10^{-9}	1	4.907×10^{-9}	11.25	5.736×10^{-3}
Width	1.080×10^{-5}	1.866×10^{-9}	1	1.866×10^{-9}	4.278	6.087×10^{-2}
Length * Width	-1.056×10^{-5}	1.785×10^{-9}	1	1.785×10^{-9}	4.093	6.591×10^{-2}
Error	·	5.233×10^{-9}	12	4.361×10^{-10}	·	·
$\frac{a_3-a_2}{a_1}$: Saturation Force (p-value vs Constant model: 0.0174)						
Constant	29.00	·	·	·	·	·
Layer	6.799	739.6	1	739.6	3.128	0.1023
Shape	10.13	1641	1	1641	6.941	0.02179
Length * Width	−8.621	1189	1	1189	5.028	0.04461
Error	·	2838	12	236.5	·	·

Table 4.5 – Additive model for the cell characteristic parameters

Saturation force The saturation force is mainly explained by three factors but with one (the number of layers) which has a $p-value$ of around 10%. This choice is intentional, as further seen in section 4.2.3. Moreover, the shape has still the most important effect, but its proportion with respect to the size of the area shows that it is less crucial than for the unloaded conductance. Finally, the width has no more an impact by its own but only with interaction with the length of the cell.

4.2.3 Feasibility of the sensor

From the three responses of the model presented above, only two are relevant for the design of the sensor. Indeed the base resistance of any sensor within the boundaries of this study are higher than 10 kΩ. Measuring a resistance in this range and above with the circuitry presented in figure 4.3 will not present any technical difficulties. This response of our model is still relevant to choose the right parameter of the amplification circuit. On the opposite, the two other responses directly impact the suitability of this type of sensor for leg load estimation.

For any application, one would like to maximize the sensitivity of the sensor, and it saturation force. In this regards, the type B shape should always be considered, as it have on both responses the higher positive factor. Furthermore, if a type A cell with a small area is used, the saturation force may even drop below the specifications. Finally for any particular application, the length and width of the cell will be imposed by the foot design. However

these two factors are not equally impactful as seen in table 4.5, with the width having an higher positive effect. Therefore the orientation of the zebra of the electrode should be chosen accordingly and the width of the sensor should always be larger than its length (with the naming convention of figure 4.2). As for the saturation, the model depends only on the shape and area, and the number of layers. Since the two first factors are already chosen, only the number of layers could be increased to meet an higher saturation specification. This is the reason why this factor is kept in the model, despite its high p-value (0.1023).

Finally, if we take the foot of the original Cheetah-Cub the sensor should have a size of 2 cm by 3 cm. Using an electrode of type B, two layers of Velostat and the use of glue, we would get a sensitivity of $1.01 \times 10^{-5}\,\Omega^{-1}\,\mathrm{N}^{-1}$, a base conductance of $4.14 \times 10^{-5}\,\Omega^{-1}$, and a saturation force of 31.5 N, which is over the 1.4 body weight ($\approx 15\,\mathrm{N}$) Ground Reaction Force (GRF) shown in *Spröwitz et al.* [2013]. This sensor would have a unloaded resistance of $\approx 25\,\mathrm{k}\Omega$ and a resistance at max load of $\approx 5\,\mathrm{k}\Omega$. By choosing $R_F = 75\,\mathrm{k}\Omega$ and $V_{REF} = 250\,\mathrm{mV}$ the voltage V_O of the amplification circuit, would change from $\approx 750\,\mathrm{mV}$ to $\approx 4.7\,\mathrm{V}$, which can then be easily acquired using a standard Analog to Digital Converter (ADC) IC.

Experimentally, the same values for the static case were retrieved when building a sensor with the above parameters. However, when applying dynamic loads, the sensor showed a lot of hysteresis which made it non reliable to estimate accurately the leg load. An extension using the combination of this sensor and a PZT dynamic sensor is presented in the next section.

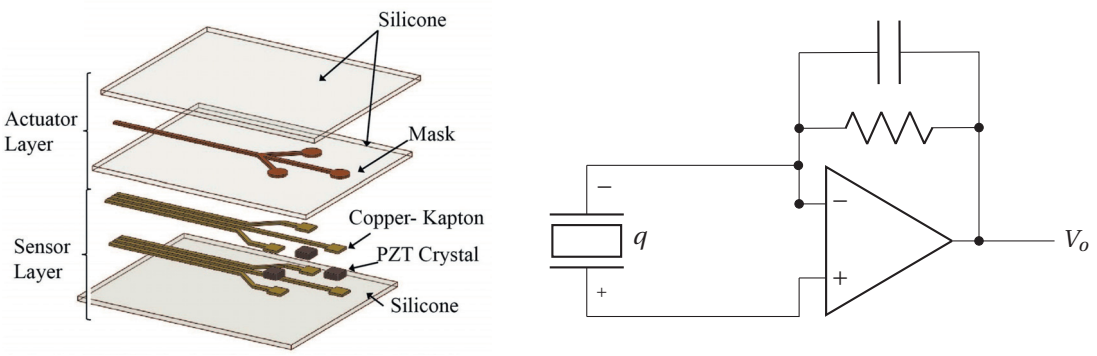
4.3 Multimodal Approach Using Kalman Filters

We suggest here to develop a multimodal sensor, by combining a piezoresistive sensor and a piezoelectric one, as performed by *Göger et al.* [2009]. However instead of estimating separately static and dynamic forces, we suggest here to fuse the two information to improve the accuracy of the load estimation.

The dynamic sensor that will be used is the one presented by *Sonar and Paik* [2016]. Although this sensor is a dynamic tactile sensor, it can estimate static forces using SPA. However this case, adding the SPA actuation in the Cheetah-Cub robot is not desirable, at it will add too much inertia to the robot. Therefore the same sensor will only be used only for the dynamic sensing. part. The amplification circuitry for the piezoelectric sensor is the same than the one used in *Acer et al.* [2015]; *Sonar and Paik* [2016], i.e. a charge amplifier (figure 4.6b)

4.3.1 Data Fusion using Extended Kalman Filter

The Extended Kalman Filter (EKF) is widely used for data fusion in Inertia Measurement Unit (IMU) [*Ahmad et al.*, 2013]. In that case, one is interested, from a gyroscope and an accelerometer to build a stable estimation of the IMU absolute orientation. The gyroscope is a biased orientation rate sensor, and the accelerometer, a biased estimator of earth gravity, subject to a



(a) Structure of the sensor

(b) Amplification circuit (charge amplifier)

Figure 4.6 – Structure of the piezoelectric sensor, taken from *Sonar and Paik* [2016]. (a) Mechanical construction: It is constituted of a PZT crystal encapsulated between two silicon layers. (b) Amplification Circuit performed by a charge amplifier.

lot of external perturbation. The earth acceleration is in turn an incomplete observation of a reference orientation of the IMU. The idea is to integrate the gyroscope information, that due to error and biases will drift, and to compensate that drift using the inaccurate partial information of the accelerometer. Such a filter will need to make an assumption on which part of the measured acceleration is due to earth gravity and which is due to the IMU self-acceleration. The data fusion between a static and dynamic force sensor will work in a similar way. We need to integrate a biased rate information, as the dynamic sensor acts closely like a rate sensor, and correct it with a biased static information, the static piezoresistive sensor described in section 4.2.

One of the most diffused method to deal with such incertitudes are Bayesian Filters. Kalman Filter and its Extended version are Bayesian Filters using Gaussian probability distribution function to model this incertitude. A more in-depth explanation and derivation of these filters can be found in *Thrun et al.* [2005].

The prediction step in IMU data fusion is a nonlinear operation. It requires the integration of a three dimensionnal orientation from the angular velocities. Therefore an EKF or an Unscented Kalman Filter (UKF)is required In the case presented here, the prediction step is a linear operation, since its a discrete integration of an unidimensional signal. However here the sensors presents nonlinearities such as hysteresis and saturation, that requires the use of an EKF, for the measurement function derivation.

The next subsections will describe two different EKF implementations that differ mainly on what is included in the state and how the measurement functions are defined.

4.3.1.1 Mathematical formulation

Rate only based The first filter F_1 use only a three dimensional state:

$$X^T = \begin{bmatrix} F & \dot{F} & D \end{bmatrix} \tag{4.6}$$

Where F is the actual force measured by our sensor, $\dot{F}$ the rate of this force and D a bias on the rate. Indeed any static error in the integration would lead to a drift and incoherent measurement. We therefore add a degree of freedom to the Kalman filter to avoid to integrate this bias and avoid any asymptotic drift in the absolute estimation.

The prediction step associated to this state is:

$$X_{t+1} = \underbrace{\begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{=A} X_t + \epsilon_t \tag{4.7}$$

It just derives from the fact that the rate is the derivative of the force, where Δt is the data acquisition timestep. ϵ_t is the added prediction noise, which is assumed normally distributed, centered and with a covariance R_t. The measurement step is:

$$h^T(X_t) = \begin{bmatrix} h_{PZR}(X_t) & h_{PZT}(X_t) + D \end{bmatrix} + \zeta_t^T \tag{4.8}$$

Where $h_{PZR}(X)$ and $h_{PZT}(X)$ are the response functions identified in section 4.3.2.2. We note $H_t(X_t)$ the jacobian of this function estimated at X_t. ξ_t is the measurement n oise, which is again assumed normally distributed, centered and with a covariance Q_t. As a reminder, the EKF iteratively estimates the state X_t at each step by a Gaussian represented by its mean μ_t and covariance Σ_t, with the measurement z_t through the following computation (adapted from *Thrun et al.* [2005]):

$$\begin{aligned}
\bar{\mu}_t &= A\mu_{t-1} && (4.9)\\
\bar{\Sigma}_t &= A\Sigma_{t-1}A^T + R_t && (4.10)\\
K_t &= \bar{\Sigma}_t H_t(\bar{\mu}_t)\left(H_t(\bar{\mu}_t)\Sigma_t H_t(\bar{\mu}_t)^T + Q_t\right)^{-1} && (4.11)\\
\mu_t &= \bar{\mu}_t + K_t\left(z_t - h(\bar{\mu}_t)\right) && (4.12)\\
\Sigma_t &= \left(I - K_t H_t(\bar{\mu}_t)\right)\bar{\Sigma}_t && (4.13)
\end{aligned}$$

where R_t and Q_t are respectively the process and measurement noise covariance.

Acceleration based As we will discuss in section 4.3.2.2, the piezoelectric sensor response seems not dependent on the rate alone, and could be better explained by a linear combination of the rate and the acceleration. Furthermore this relation is dependent on the frequency of the estimated signal. Therefore another EKF F_2 is proposed that is estimating the acceleration:

$$X_t^T = \begin{bmatrix} F & \dot{F} & \ddot{F} & \omega & D \end{bmatrix} \tag{4.14}$$

Where ω is the frequency of the signal. The corresponding prediction function is:

$$X_{t+1} = \underbrace{\begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{=A} X_t + \epsilon_t \tag{4.15}$$

and the measurement function is:

$$h_t^T(X_t) = \begin{bmatrix} h_{PZR}(X_t) & h_{PZT}(X_t) + D & \omega \end{bmatrix} + \zeta_t^T \tag{4.16}$$

Here a direct observation of the frequency of the signal is assumed. It comes that during locomotion, this frequency will be known as it is directly the mean value of the derivative of the driving phase of the leg.

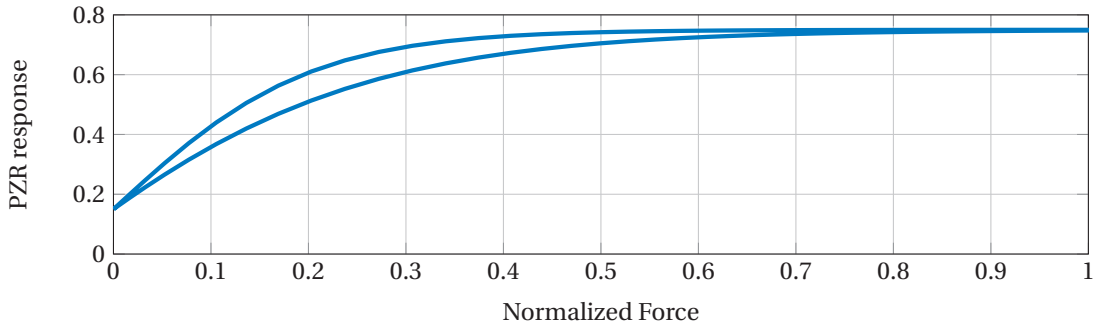
4.3.1.2 Validation using simulated Data

Finally, to validate and illustrate this approach, a simulation is conducted. For that purpose a bump shaped periodic signal of 2 Hz is generated (figure 4.7b). The bump function is defined in the interval $]-1;1[$ by:

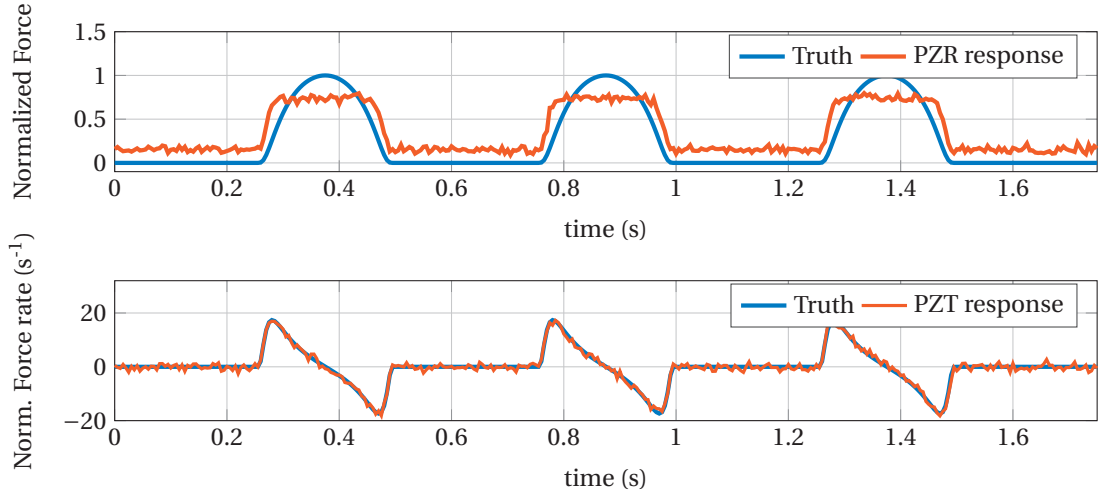
$$\forall x \in]-1;1[: \ f(x) = e^{\frac{1}{1-x^2}} \tag{4.17}$$

The bump function has the nice property to be closely similar to the vertical GRF signal observed in Cheetah-Cub [*Spröwitz et al.*, 2013], and to be infinitely differentiable, which avoids large rate values.

For simulating the piezoresistive sensor response, a function $h_{pzr}(X)$ similar as the one shown in section 4.3.2.2 is used. This function is a combination of sigmoid functions. It has the advantage to be continuously differentiable, on the opposite of the one described in section 4.2.2.1. The simulated response without noise is shown in figure 4.7a. The piezoelectric sensor response is simulated by the rate of the force, with a 0.5% added bias. Gaussian noise was added to both sensor responses (figure 4.7b).



(a) Piezoresistive response model



(b) Data generated for simulation

Figure 4.7 – Data used for simulating the proposed EKF. (a) The piezoresistive sensor is simulated with a response that shows both nonlinearities (saturation) and hysteresis. (b) Sampled data used for simulation, with Gaussian noise applied to both simulated sensor responses.

The EKF presented in section 4.3.1.1 was then implemented. The gain are manually hand tuned: the measurement covariance is chosen to be a diagonal matrix with variances equal to twice the engineered noise for both signals. The process covariance is chosen as:

$$Q = \begin{bmatrix} 1 \times 10^{-4} & & \\ & 1 \times 10^{-1} & \\ & & 1 \times 10^{-4} \end{bmatrix} \tag{4.18}$$

The idea behind is that without any control variable, the rate should be much more free to change step to step than the signal or the drift variable.

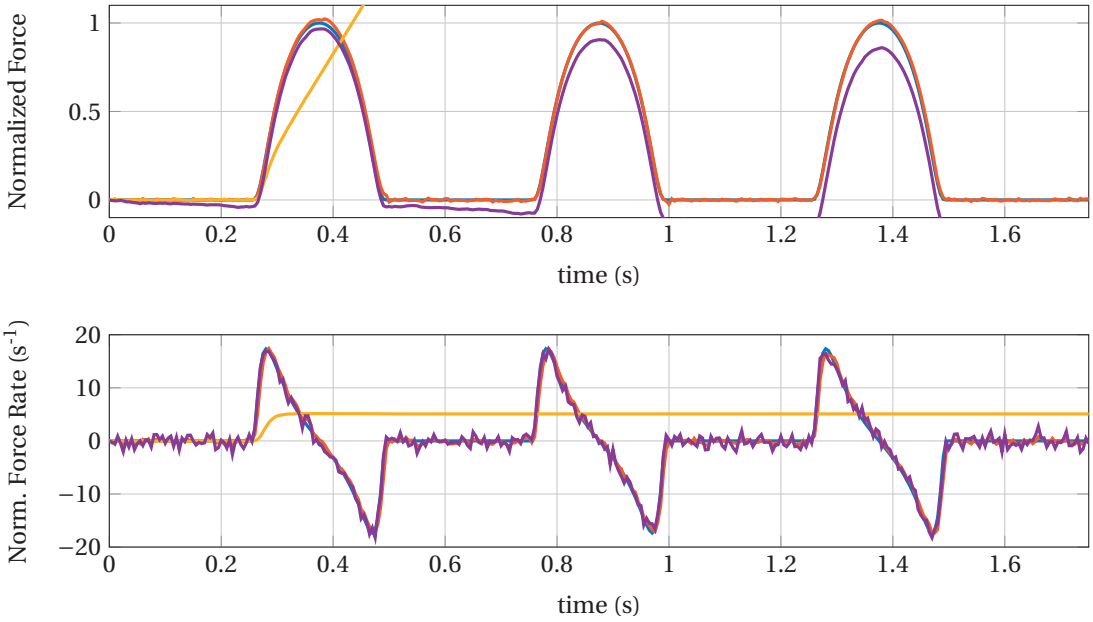


Figure 4.8 – EKF simulation results. Blue: truth, orange: EKF estimation using both piezoresistive and piezoelectric data, purple: estimation using only piezoelectric data, gold: estimation using only piezoresistive data. The use of only a single sensor leads to drifting estimation, whereas combination provides accurate tracking.

Figure 4.8 shows the estimation of this EKF with two other versions. Each one considers only the piezoresistive or piezoelectric data, by removing the corresponding line in the observation matrix (4.8). We see that the EKF is able, when it uses the two sensor data to reconstruct pretty well the truth signal. The estimation shows at the maximum of each bump a small hiccup. It could be explained by the fact that the hysteresis introduces instabilities when the rate is changing sign. On the other hand, estimation based solely on the piezoresistive data quickly diverges due to the saturation and noise. The EKF solely based on piezoresistive data shows the same shape than the truth data but slowly diverge because of its static bias. Such a bias is almost inevitable in practice, and the sole integration still needs to be corrected with the poor static estimation.

4.3.1.3 Optimal covariance estimation

The process (R_t) and measurement (Q_t) covariance matrices in (4.9) play a crucial role. They model the incertitude in each of the integration steps. Even if the sensor response identification gives insight on which values to use, they have to be chosen precisely to have a good estimation made by the Kalman Filter. Moreover there is no easy way to estimate the prediction noise R_t that should be used in (4.10). One solution for such simple problem would be to estimate this meta parameters by trial and error until the results are satisfying.

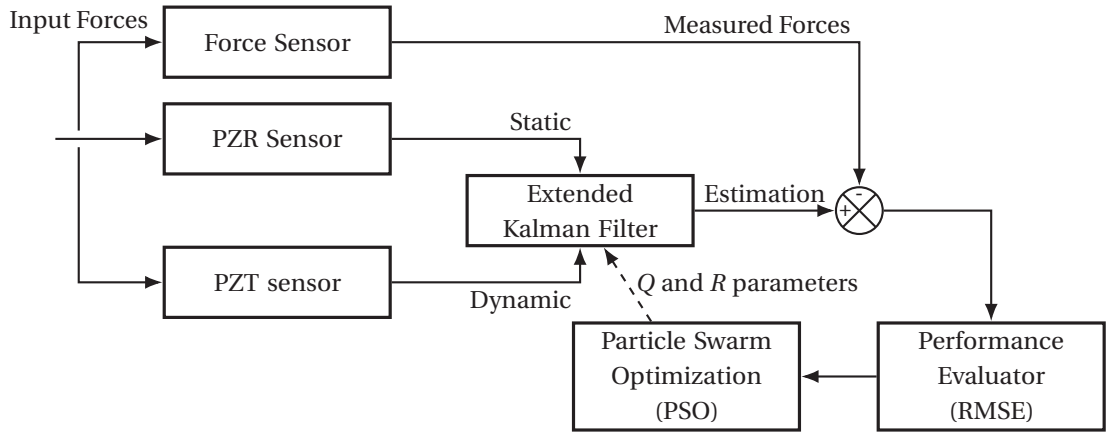


Figure 4.9 – Optimal covariance estimation of the EKF using Particle Swarm Optimization (PSO). Performance of the EKF is measured using RMSE with the ground truth signal and is used in the fitness function of the PSO to find optimal coeeficients for Q and R matrices.

Another approach would be to rely on optimization to determine these matrices coefficients. In simulation we would have a ground truth to compare our estimation to, as well in the test bench presented in section 4.3.2.1 which feature a force sensor that measure forces applied to our tactile sensor. We propose to use here PSO [*Kenndy et al.*, 1995] to estimate optimally these meta parameter. Such evolutionary algorithm has already been used for Kalman Filter tuning [*Jatoth and Kumar*, 2009; *Laamari et al.*, 2015]. The idea is, for some test data, to use as an objective function the root mean square error between the value estimated by the EKF and the ground truth. The choice of PSO make sense as there is almost no particular relation on the input parameter, and PSO is well-suited for multimodal objective function.

For conciseness, the fundamentals of PSO will not be stated here. The following objective function is chosen:

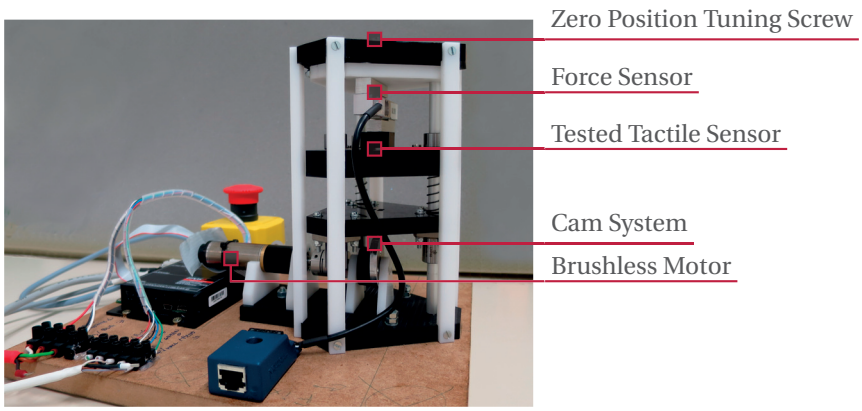
$$O = e^{-\alpha RMSE(F)^2 - \beta RMSE(\dot{F})^2 - \gamma RMSE(D-\bar{D})} \tag{4.19}$$

Where $RMSE(x)$ stands for the Root-Mean-Square Error of the x estimation, and $D-\bar{D}$ the difference between the value of D and its mean value over the dataset. α, β and γ are importance factor and were experimentally chosen to reflect relative importance between each criterion. Experimentally $\alpha = 5$, $\beta = 1 \times 10^{-2}$ and $\gamma = 1 \times 10^{-5}$ gave the best result in terms of convergence.

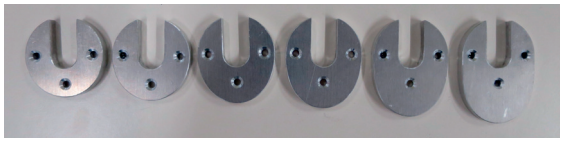
4.3.2 Experimental validation

4.3.2.1 Test bench design

In order to validate the sensor design and processing, it is needed to use it in conditions close to its intended use. Directly testing it on the Cheetah-Cub robot is not desirable, as it would be



(a) Test bench



(b) Replaceable Cams

Figure 4.10 – Test bench setup. A motor is applying forces on a tested tactile sensor and a strain gauges based force sensor. The cams are made replaceable to change the maximal forces that could be applied on the sensors. By varying the rotation speed of the motor, one can easily change the frequency of the applied forces.

difficult to repeat the experiments, or just to get an accurate absolute estimation of the vertical GRF. The condition on which they have been acquired using external force plate in *Spröwitz et al.* [2013] are extremely difficult to reproduce. Therefore a dedicated test bench is required to measure both the signal of the sensor and the actual forces applied to it.

This test bench needs to simulate the same normal efforts that occur during locomotion, i.e to apply a force with a bump profile ranging from zero up to 2 body weight ($\approx 30\,\mathrm{N}$), and with a frequency ranging from 0 to 5 Hz, to be well into the conditions, measured in *Spröwitz et al.* [2013].

For that purpose a cam system was used to transform the rotational movement of a brushless DC motor to move a carriage guided with linear bushes. This carriage would in turn press on three low stiffness springs ($k = 0.78\,\mathrm{N\,mm^{-1}}$), that presses on a second carriage on which our sensor is fixed. The whole is pressing on a force sensor rated for 50 N. This force sensor is an high end one dimensional strain gauge sensor. Apart from the cam and the linear rails in aluminum, all the test bench is made out of CNC-milled Polyoxymethilene (POM).

The advantages of using a cam system over a leading screw system are the simplified motor control and more efficient system. The main drawback is that the shape of the force profile applied on the sensor is limited by the constraints on the cam design. Not any periodic signal can be achieved, and steps-like signal are not feasible to achieve with such setup. The

one chosen here is a bump composed of two harmonic curves, in order to minimize the acceleration and torque on the motor. The cams are made replaceable, in order to be able to change the maximum of the bump force. Six cams where build in aluminum, with an eccentricity ranging from 1 to 12 mm, so maxima of the applied force profiles ranges from 2.34 to 28 N.

For the motor, a 90 W, 24 V Maxon Brushless motor, with a 14:1 planetary gear is used. This motor, before reduction, has a stall torque of 51.3 $\mathrm{mN\,m^{-1}}$, and a maximal speed of 163 000 RPM. This motor is oversized for this usage, but was used as it is a spare part of an Oncilla robot. The force sensor measuring the actual forces applied on the tested sensor is mounted on a plate that can be easily fixed at different lengths, in order to tune the setup for different sensor thickness.

The output V_O of the amplification circuits (figure 4.3 and 4.6b) are acquired with a National Instrument Compact RIO system equipped with a c9220 16 bit ADC, running at 1 kHz. The force sensor data was acquired at the same rate by a c9237 24 bit bridge module.

The cams used here were in-house CNC-milled, which is not an easy task. During the milling process, when bridges are made to hold the pieces in place, small scratches where carved by the cutting tool. Such manufacturing artifacts are unfortunately not avoidable with a desktop CNC machine. Those scratches, when the roller of the cam system passes over introduces a lot of vibrations that are caught by both the truth force sensor, and the dynamic one. In order to get rid of this noise, all data is low-pass filtered numerically by a fourth order 10 Hz Butterworth filter, before any processing by the Kalman Filter.

To validate the sensor, it was tested with a 12 mm cam (28 N peak force) at various frequencies ranging from 0.5 to 4 Hz, in increment of 0.5 Hz, each time for at least 10 periods. The measurements were performed in a single run.

Since the EKF is a filter based on model of our process and its sensor, its reliability is directly linked to the accuracy of those models. One of these runs are used to build a model of each of our sensor, and a second run was used to tune the filter. A third run is used for validating the filter approach

4.3.2.2 Sensor response identification

Piezoresistive response As shown in figure 4.11, the piezoresistive sensor shows both a quick saturation around 10 N and hysteresis, even if this saturation would expected to occurs at more than 30 N in the purely static case. The EKF model needs the measurement function to be continuously derivable by the state variable. Considering a parametrized sigmoid function :

$$S_{a,b,c,d}(x) = \frac{a}{1 + e^{b(x+c)}} + d \tag{4.20}$$

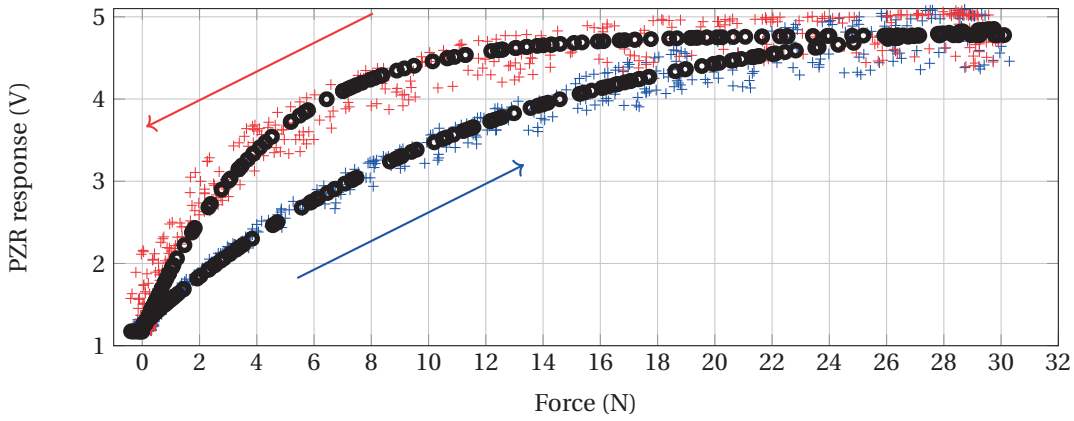


Figure 4.11 – Piezoresistive sensor identification using a mixing of sigmoid function. Blue points are increasing forces, and red decreasing decreasing. Black are point from the fitted model. $R^2 = 0.995$

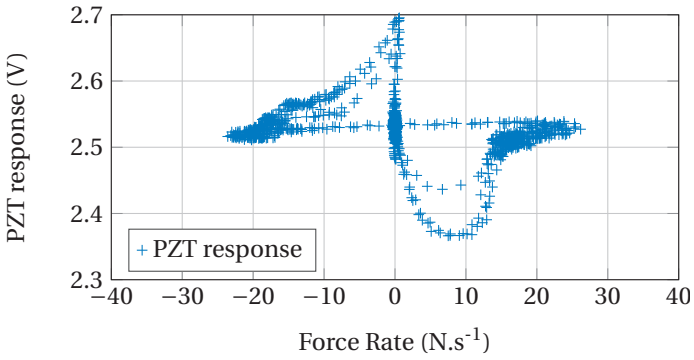


Figure 4.12 – Piezoelectric sensor response without preload.

Using a combination of those, a parametrized function that displays some hysteresis can be constructed :

$$\begin{aligned} h_{PZR}(X) &= S_{1,b,0,0}(\dot{F}) * S_{a_1,b_1,c_1,d_1}(F) \\ &\quad +(1 - S_{1,b,0,0}(\dot{F})) * S_{a_2,b_2,c_2,d_2}(F) \end{aligned} \tag{4.21}$$

It is constituted of two different sigmoid parts on the force estimation, one for increasing forces and one for decreasing forces. Another sigmoid is used on the rate estimation, to switch between this two different curves. Nonlinear least square fit is then used to identify the parameters in (4.18). Results are shown in figure 4.11.

Piezoelectric response First, there was some technical issues to operate the piezoresistive sensor close to a rate transducer as shown in *Acer et al.* [2015]. In the first trials, the bench was tuned to leave a small gap between the force sensor and the tactile sensor when the cam was at its lowest. This was intended, as in locomotion, during a part of the gait cycle, the foot does not touch the ground. However, under that condition, there was no easy relation to be found between the piezoelectric sensor signal and the actual force rate as shown in figure 4.12. Those effects could come from nonlinearity in the silicon stiffness and the fact that some air bubbles, normally used for SPA are present above the piezoelectric cell.

By adding some preload and using plain silicon rubber, the response shown in figure 4.13 was obtained. However there is still no simple relation seen between the rate and the PZT response. But using a linear combination of the rate and the acceleration of the force, a better relation could be identified. Furthermore those linear coefficients decreased exponentially with the frequency of the signal. It explains why the rate based EKF (section 4.3.1.1) needs to be augmented with the two new state variables ($\ddot{F}$ and ω in F_2), to rely on a better model for the PZT sensor observation.

4.3.2.3 EKF parameter optimization

First, the measurement noise was estimated by observing the distribution of the residue of the identification. Those residues are not distributed along a normal function (see figure 4.14) for both sensors, making harder to use a Kalman Filter to filter this data. Nonetheless an attempt to tune the Covariance coefficient using PSO was conducted.

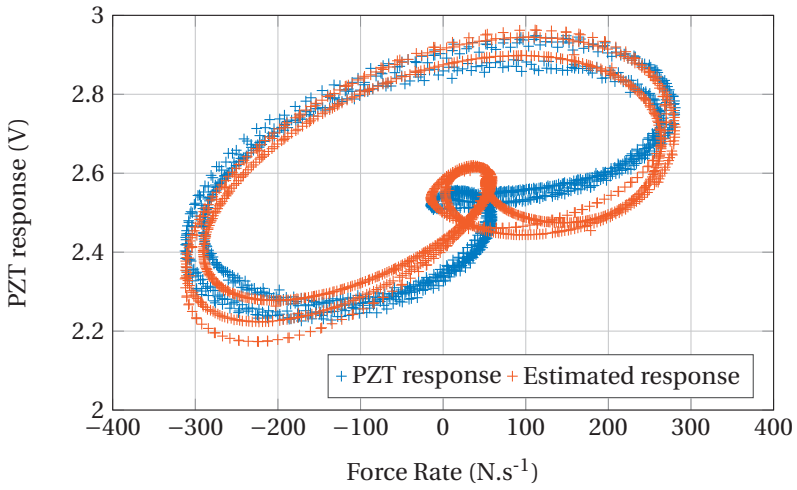
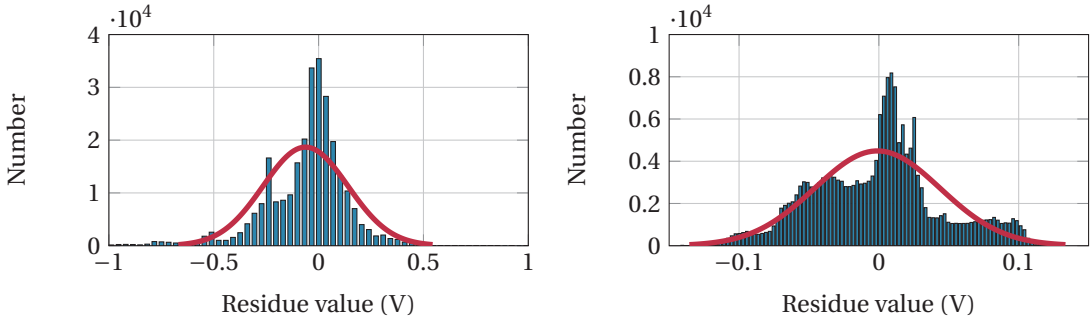


Figure 4.13 – Piezoelectric sensor identification with preload. There is no direct relation between the rate of the force and the response of the piezoelectric sensor. By trying to fit a linear combination of both the rate and the acceleration of the force, the following least square fitting could be proposed. $R^2 = 0.9997$



(a) Piezoresistive residues

(b) Piezoelectric residues

Figure 4.14 – Distribution of the errors of the identification

The resulting optimized EKF estimations are depicted in figure 4.15, for part of the data (at a frequency of 1 and 2.5 Hz). The resulting EKF produces stable pattern for the whole experiment, but it is worth to note that a large combination of gain were given divergent results. The estimated curves are also not really close to the truth data. They tend to have lower maxima for lower frequency (fig. 4.15a) and larger maxima at higher frequencies (fig. 4.15b).

Furthermore, the estimation confidence is going higher near the maxima of the bumps. This behavior could easily be explained by the saturation of the piezoresistive response. When it saturates, there is no information gain by the data given by the piezoresistive sensor, and the estimation has to be solely based on the piezoelectric data. As there is a numerical integration, variance on the rate is cumulatively added to the force estimation, without any correction. When the piezoresistive sensor goes out of its saturation region, the filter could use the correlation between the two signals to increase back its confidence.

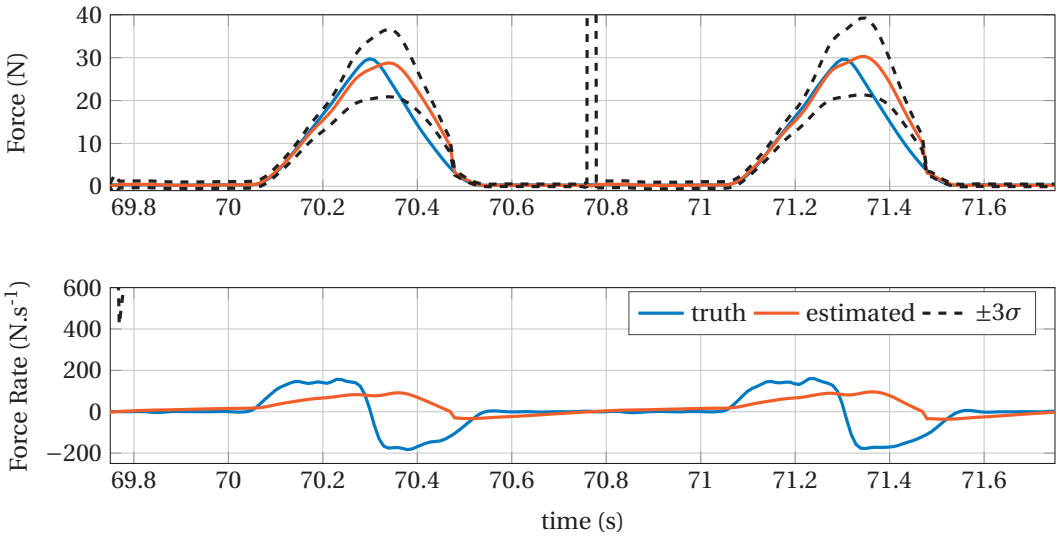
Moreover, both rate estimates are quite poor and shows some phase lag with the truth data, nor is quite close to the rate of the estimate.

Looking at the covariance values chosen by the optimization process, there is 3 order of magnitude higher variance of the piezoelectric measurement noise coefficient and the piezoresistive one. Such disproportion in the signal confidence could be explained that these two signals contains contradictory data for our model, for example a non-null rate with a constant, non-saturated, static signal. Therefore, the Kalman filter, in order to be stable, requires a poor confidence in one of the two signals. Indeed some particles, but with a lower fitness, shows the opposite property of a variance higher in the piezoresistive measurement, but after a few iterations of the PSO, no particles shows variance in the piezoresistive and piezoelectric measurement of the same order.

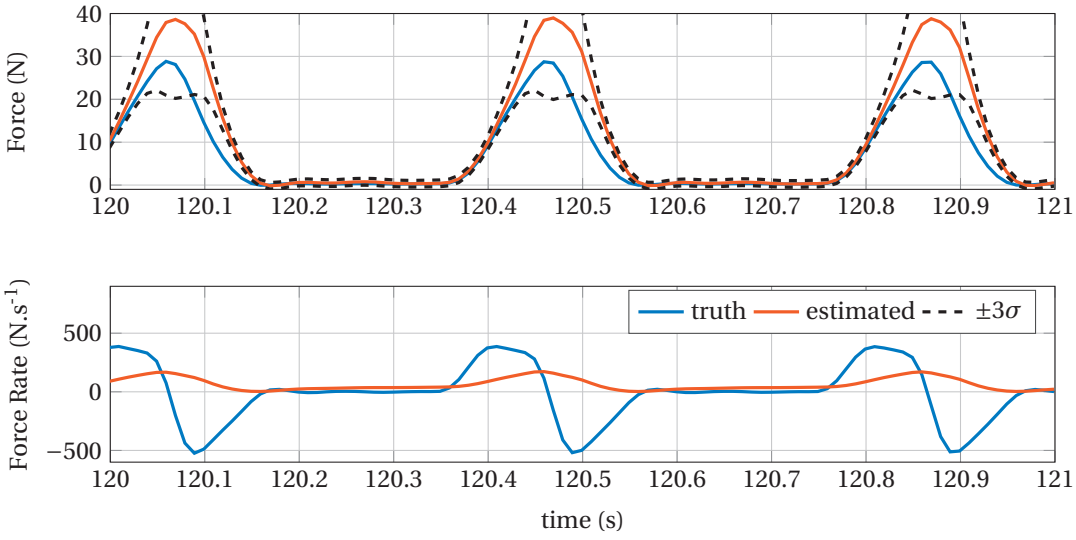
4.3.3 Discussion

The EKF approach shows practical results which were quite poor compared to the one expected in theory (section 4.3.1.2). One assumption to explain this result is the fact that the signal from both sensors, due to delay and poor identification shows some contradictory data. Actually the piezoelectric sensor does not produces a signal closely related to the rate of the force (section 4.3.2.2). *Cutkosky and Ulmen* [2014] shows that a stress rate sensor can be build with a PVDF material and a circuit measuring the current produced by the sensor. The only differences with the charge amplifier used here (figure 4.6b) is the abscence of the feedback capacitor, which here acts as a voltage integrator. The sensor used here is not close to a stress rate sensor. We can see in figure 14 and 15 of *Acer et al.* [2015], there is a larger fall time than the rise time of the curve, were both should have been exactly the same if the rate of the force would have been estimated. We hypothethize that this small difference between the dynamic force sensor used here and a true stress rate sensor is a reason why we have contradictory data. However the hyperelasticity of the silicon used to seal the sensor may also have an effect on the stress rate response and it may not only be sufficient to simply use the amplification circuit proposed in *Cutkosky and Ulmen* [2014].

The EKF we propose still show some stable patterns using a much higher confidence on the piezoresistive signal alone. However the final estimation stills depends on the frequency of the signal. There was other issues with the piezoresistive sensor, such as repeatability and long term operation, as after a few minutes, the amplitude of the signal increased even if the same forces would be applied. The Velostat resistance is subject at some kind of double response: one depending on a “fast” deformation of the material, and one that slightly builds up with the average long term load on the sensor. This repeatability issues may come from the fact that the Velostat is not designed to handle elastic deformations. The approach here may be easier using a commercial sensor or switching to another type of transduction such as capacitive sensors.



(a) 1 Hz



(b) 2.5 Hz

Figure 4.15 – EKF estimation results with PSO optimized noise covariance matrices. Truth, estimation and the $\pm 3\sigma$ boundaries of the estimation are shown. The estimation does not diverge but has frequency dependent maxima. On the other hand the estimated rate is poorly estimated.

5 Integration of Robot Software and Hardware

This section details the work I did over the years at BioRob on the integration of software and drivers on a variety of quadruped robots (figure 5.1). This chapter, in addition to a common presentation of the achievements that can be found in a thesis dissertation, also aims at listing tips and advice on how to develop software and integrate complex systems. Indeed, software development and architecture are fields that I did not study during my undergraduate studies, and integration issues are often only learned by practice. Furthermore robotics is a field at the intersection of many disciplines (mechanics, electronics, software development and mathematics), and other students may easily end up in a situation where they have to develop highly polished software for a whole consortium or integrate a robotic platform without being perfectly prepared. I hope some of the advice would help them to build their own perspective and avoid some pitfalls.

This chapter will first present some of these personal guidelines, and illustrate them with two examples of development I did during this thesis: the real-time robot framework `robo-xeno` and the integration of firmware and drivers on the Oncilla robot.

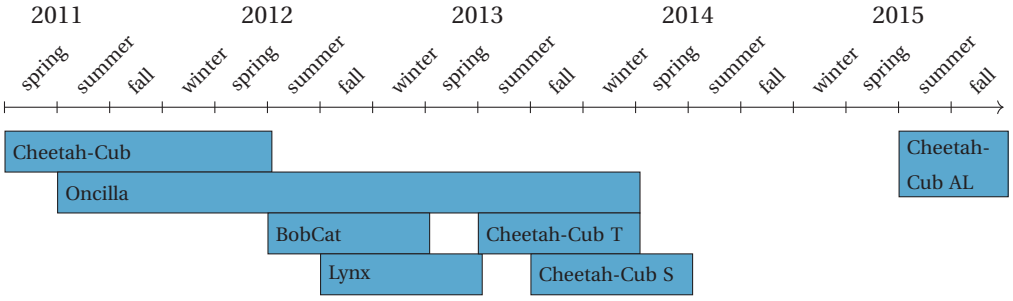


Figure 5.1 – Timeline of robots developed during this thesis. Published research: a) Cheetah-Cub [*Spröwitz et al.*, 2013] b) Oncilla [*Spröwitz et al.*, 2011] c) BobCat [*Khoramshahi et al.*, 2013] d) Lynx [*Eckert et al.*, 2015] e) Cheetah-Cub S [*Weinmeister et al.*, 2015]

Project	Contribution	Development team size	Size of audience
sbcp-uc	Lead developer	1	3 other Adaptive Modular Architecture for Rich Motor Skills (AMARSi) Firmware developers
sbcpd / libsbcp	Lead Developer	1	3 other AMARSi software developer
libwebots	Contributor	3	15 Other lab members
codyn	Package maintainer	3	15 other lab members
liboptimization	Package maintainer	3	15 other lab members
robo-xeno	Lead developer	1	≈ 5 other lab members
libamarsi-quad	Lead developer	1	≈ 8 other AMARSi PhD student
amarsi-benchmark	Lead developer	1	≈ 8 other AMARSi PhD student
liboncilla	Developer	3	≈ 5 Oncilla user
liboncilla-hw	Lead Developer	3	≈ 5 Oncilla user
liboncilla-sim	Lead Developer	2	≈ 5 Oncilla user
oncilla-simulator	Lead Developer	2	≈ 5 Oncilla user
cdn-webots	Lead Developer	2	≈ 5 Oncilla user

Table 5.1 – List of some software project related to this thesis.

5.1 Developing Software for Robotics

Over the year, with the experiences acquired on working on more than a dozen of software project (see table, and through collaboration with more experienced developers, I built my own personal definition of the task of developing software:

Definition. *Developing a piece of software is communicating to other people a solution to a current problem.*

This definition relies on three key points:

Solution to a problem From a pure logical point of view, a computer program is just a complex mathematical operation performed by a machine on some given data and output its results. As much as this definition is true, it does not explain the intent or the reason of the existence of this software. These intentions arise when we see software as a solution of a problem it tries to solve. Furthermore the quality of a software could be estimated on how good the solution it presents solves the problem. A corollary of this approach to software development is, to develop good software, one must carefully define and state the problem it tries to solves. Furthermore defining well the scope of a program library could help balance development cost by, for example defining a threshold at which the piece of software could be

considered to solve its problem well enough without further development.

Current problem The previous key point could be just summarized as basic engineer project management and as first steps in system development approach, that could be applied in almost any field, such as mechanical or electronic design. However, there is a fundamental difference in software development as the replication costs are null, and modification costs are orders of magnitude lower. Therefore it could often be less costly to just modify an existing piece of software than build a new one. Going even further, current practice in software architecture encourages the aspect of design that could easily be adapted to a new situation. This is why I personally see the aforementioned problem as punctual, and doomed to change in the future.

Communication with other people In most situations, any project will not only be done by you for your single purpose. Even for strictly personal projects, if they span over several months, yourself could be considered another person, because you may have forgotten some aspects of your code, acquired new skills that give you a new mindset over the particular approach. This reason is often given to explain why documenting code is important, but it should go even further. Each piece of code has intent a purpose, and its by the clarity of its structure, definition, paradigm that you can also communicate those intentions to other developers or users.

5.1.1 Some useful design principles

Modularity Modular designs are widely pushed forward especially in the open source community or in the UNIX philosophy. Those designs enforce the sectioning of a program in smaller, lightly coupled entities, that solve only a subproblem the complete program tries to solve. A modular approach will bring several benefits. Development effort and optimization could be done in parallel, as once the purpose and interface of those modules are defined they could work and be tested in isolation of the complete system. Unit testing and automatic testing could be simplified and often be quicker. Finally the complete system could be modified more easily in certain situations, as modifications could be performed only on subparts of the system and not its whole.

These statements are however only true if the modularity or the architecture of the program are defined cleverly enough, i.e. if those modules at any level are lightly coupled. If those are heavily coupled, or highly dependent on each others, then they are high chances than any modifications of one would require to modify another module, and so on. A system with highly coupled modules is only modular in appearance and closer to a monolithic system. There exist many rules of thumb in order to design lightly-coupled modules, such as the Demeter law, or the fact to use the most simple representation between modules. For example if the data (or relation) that is passed between modules is complex, it is more prone to change, and

as consequence, the two modules will be more prone to change. Another rule would be to avoid spreading a functionality or responsibility in the software over two modules.

Once a modular design is adopted, in order to ensure it to be more compliant to change or evolution, there is a dual rule from this latter rule: The Single Responsibility Rule.

Single Responsibility Principle This principle states that any entity (function, class, or module/package, program) should endorse a single and unique responsibility. A responsibility is often mistaken with the role or functionality. A module has functionalities or a role that respond to a particular purpose or concern. What we want to achieve is not a module to have a single functionality but only one concern, a single purpose, only person to respond to.

Level of abstraction Often it could be difficult to follow strictly the aforementioned principles. When facing the situation where we would like a module to cover several responsabilities, it may highlight that we did not choose the right level of detail for that module. By moving one level of detail up, we may still make the module to respond to one, less precise, more broad responsability, and use a less complex interface. The details could still be addressed one level deeper, with submodules which could also have more complex boundaries that cover those details.

Convention over configuration Another design principle that could be useful for robotic system, is "convention over configuration". It arose recently to solve complex, cumbersome configuration, as for example the build system of Go [*The Go Team*, 2015]. Specifying a build system for a large software is a complex task: for the C/C++ language, there exist exists a lot of tool to address this issue — autotools, CMake, Ninja, Waffle…— and each of these tools provides their own approach to setup build. On the other hand, Go, use convention over configuration for its build system. The compiler knows how to build the software, just by knowing where the files of the project are placed. Adding a new software module is as simple as adding a new file to the project. Just by using strong conventions, their is no need to take any manual actions to configure the build. In the Go example it is also possible in some extend to specify some specific build rules, but for most of the cases this is not needed. This concept is here relevant because robotic systems are complex. They often have a large number of inputs and outputs, and the task of configuring a robot polynomially grows with this number of I/O. Using convention over configuration wherever it is possible, we could help to leverage this cumbersome task, as this will be furtherly illustrated on how configuration of I/O in `robo-xeno` is done.

5.2 robo-xeno: a Framework for Real-Time Drivers and Low-Level Controllers

5.2.1 Purpose and context

robo-xeno aims to be an hardware agnostic real-time framework for developing robot drivers and run low-level algorithms. It tries to provide the following features: given a robot prototype help the user to build the software structure that will be able to run some simple experiments on the robot, i.e. schedule I/O and process tasks, and provide adequate API for the user to specify those tasks. It should meet the following specifications:

- Perform I/O data cycles in the range of 100 Hz to 10 kHz
- The core framework should be hardware agnostic.
- It should be real-time, i.e. the scheduling should have jitters of the order of 10 µs
- Be suitable for embedded devices.
- low-level control could be run on the embedded device, so the process should be allowed to be computationally intensive in regards of pure I/O operations.

The need to solve this particular problem was not well-defined at the start of this project, as early as the implementation of the first cheetah-cub robots. Slowly with the arrival of new robots, with different hardware and different structure, in order to avoid to use different versions of substantially the same controller, its role was generalized in a more hardware independent fashion, and necessitated major rewrites. When such a generalization was needed, the use of a more general robotic platform such as the one described in section 5.2.5 was not considered. Indeed at that times most of those platforms lacked real-time capabilities, but very low jitter and high data cycle I/O are required.

5.2.2 Architecture and overview

5.2.2.1 Identification of main roles

As explained previously, one way to design a clean architecture is to identify the role or purpose of the actor interacting with our software. For robo-xeno, at the highest level of abstraction, we can define two distinct roles or responsibilities for the user that would need to use it:

Robot integrator This responsibility deals solely with the real-time delivery of data. Any robotic application requires some data to be acquired from a physical device and some other data to be delivered to other devices. The integrator or the person developing the driver for the robot has to focus only on this particular task. It does not need to know what the actual values of these data are, but only that their delivery meets the required deadline and real-time constraints, and how to communicate with this external device. i.e. which physical interface to use, which protocol implementation to use, what is the most efficient way to schedule these operations.

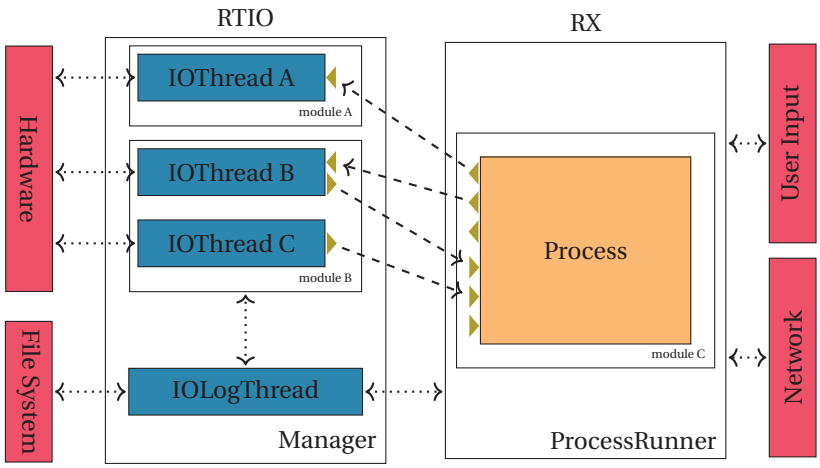


Figure 5.2 – `robo-xeno` software architecture overview

Robot user This responsibility deals solely with the actual value of the data it receives and it has to send. Knowing at which point in time it will receive information and at which point in time it will need to provide that information, its responsibility is to compute the latter from the former, i.e. provide the control algorithm of the robot. From the robot user point of view, the actual mechanism on how the data is delivered could be completely opaque and abstracted away.

In `robo-xeno`, these two responsibilities are separated using two different C++ namespaces. `rtio` would be the namespace containing classes and objects that deal only with integration role, and on the other hand `rx` is a namespace that deals solely with user parts.

5.2.2.2 Main components

These two top-most abstractions are further decomposed into smaller abstractions, as depicted in figure 5.2. From left to right, on top of the hardware lies the `rtio` namespace, then the `rx` namespace which interfaces with the user or the network.

`rtio` **decomposition** Each operation with the hardware are handled by `IOThread`. They are real-time tasks whose purpose is to actually communicate with the hardware, i.e. talking with the kernel drivers.

Its the responsibility of the integrator of a system to specify how many of those tasks are required given a robot architecture, and what is the purpose of each of those tasks. For example, to optimize the communication with a robot which have to deal with many serial ports at once, it may be more optimal to have a single task with a `poll` loop rather than a collection of concurrent tasks that may put too much overhead on the system scheduler. On the other hand having a generic module for each of these devices may not be optimal in

performance but permits a fast integration for development purpose. Since it is required from the integrator to provide the full implementation of those tasks, it gives the integrator a leverage on the development costs versus performance trade-off.

There are another kind of real-time task threads that are defined in rtio: IOLogThread. Their purpose is to log the data on the file system. They lie on the rtio side, as its the responsibility of the integrator to optimize how this data is logged. Accessing this log could be helpful from the user side, but it is only required from the user to enable this service, not to handle him itself.

Finally all the scheduling of these objects are handled by a Manager. This module only abstracts the real-time tools used for the scheduling. The scheduling algorithm itself will be later explained in section 5.2.2.3.

rx decomposition On this side, the user is mostly asked to provide a Process object, that will run the desired control algorithm that would need to be performed. This object is then used by the ProcessRunner that will handle and abstract all communications with the User Interface and the network from one side, and the rtio::Manager on the other side. The first communication channel uses ProcessVariable object a user can define to tune its Process. Those can be accessed either by a command line interface executed on the embedded robot, or over IP network.

Coupling In order to ensure the lightest coupling between the rtio and rx namespaces, communication between the two is performed using the simplest object. Each rtio::IOThread registers a number of rtio::Input and rtio::Output that represents the data it is entitled to transmit. These objects are just defined by a name, and a size, and they define a directional stream of vectors of double of that given size.

On the other side, a Process is also entitled to define its rtio::Input and rtio::Output. Then the ProcessRunner can link those objects and enable the flow of data between the rtio and rx domains. These matches can be configured manually, or using the simple convention than any I/O on separated side that share the same name, the same direction and the same size should match.

5.2.2.3 Scheduling algorithm

The scheduling algorithm of robo-xeno uses an eager and simple approach as shown in figure 5.3. The time is discretized in fixed time step Δt. The integrator defines this time step, as it is directly related to the fastest data I/O cycle the robot can achieve. The user on the other hand chooses which multiple of this time step he wants to use for its control algorithm time step. At each of this control time step, the ProcessRunner will trigger one iteration of the user Process. At that point (t in figure 5.3), all Input data of all idle IOThread will be up to date

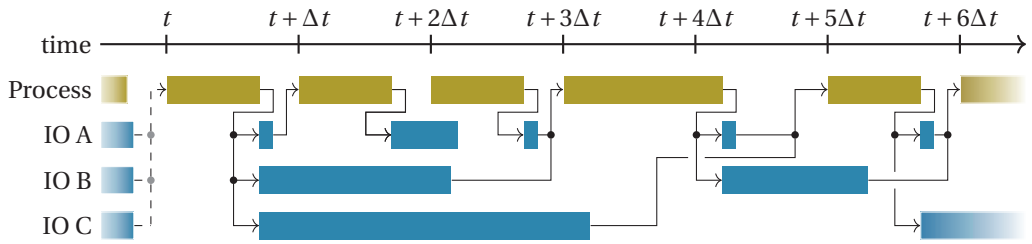


Figure 5.3 – `robo-xeno` example time-line execution. Detailed explanation are found in section 5.2.2.3.

for the process. Once this iteration is done, all `Output` data is sent to the idle `IOThread` and those are asked to iterate once.

Due to the different time scales of I/O tasks between different hardware, they may not be available for each time step, or due to an hardware jitter (IO B &C in figure 5.3, or IO A at $t+2\Delta t$). The corresponding `Input` will only be updated for the `Process` in the following time step. Similarly, as a process may take longer to iterate, it may miss a time step (see $t+3\Delta t$ in figure 5.3). The start of `IOThread` would be delayed. However in that case that exceptional situation will be reported to the next iteration of the `Process`, as at each iteration it receives the physical time elapsed since its last iteration. It forces the user to provide `Process` robust to variable time step invocation.

This scheduling heuristic is one of the main contributions of `robo-xeno`, and may seems quite simple, and not able to profit of systems with more than one core. However the contribution lies into the implementation details, as all synchronization mechanisms use clever lock-less signaling and triple buffer for concurrency management, and avoid any data copying. These technical implementations are required to ensure the lightest overhead of the real-time scheduling and to sometimes just to meet the required real-time deadline. One contribution of `robo-xeno` is to some extent abstract these technical details to the integrator and the user.

5.2.3 Implementations required by the user

To summarize, in order to use `robo-xeno`, one must provide:

- The implementation of `IOThread` : However generic implementation for all drivers required by the robot in this work are already provided. It consists of a) RB-110 driver for servomotor outputs and ADC acquisition (all robots) b) SBCP protocol (see section 5.3) required by Oncilla c) Phidget boards d) OptoForce sensors
- The `Process` object the user want to run. If the controller has the form of coupled dynamical system, bindings to the cȯdγn language [*Van den Kieboom*, 2014] are provided. Otherwise the control algorithm can be directly implemented in C++ through `robo-xeno` API.
- The configuration of the scheduling of `IOThread`

- The mapping between provided I/O by the rtio side and the needed I/O from the rx side. This process is highly simplified if good naming conventions are used between the integrator and the user.

5.2.4 Main operating features

To help the conception and tuning of low-level control algorithms, robo-xeno offers the following features:

- Online tuning of the Process through a user command-line interface. While it is running, the user can modify in real-time any ControlVariable that the Process exposes. For the CPG model we presented in chapter 3, this would be for example the step height or the step length parameters. robo-xeno provides the way to save and recall the values of those numerous parameters, in order to help the hand-tuning of controllers.
- Logging of all the real-time I/O data to the file system, to help the debugging of the Process and the IOThread.
- Modification of ControlVariable over the network. Currently this is used to connect a gamepad to teleoperate the robot.

5.2.5 Discussion and comparison with other alternatives

Framework	Real-time	TCP/UDP Communication	Hardware Support	Modularity	Data-flow Configuration
robo-xeno	Soft	No	Minimal	Yes, **Role Oriented**	Untyped, **Convention over Configuration**
Orocos	Hard with RTAI	No	+	Yes	Typed, Manual Configuration
OpenRTM	Soft	Yes (not Real-time)	+	Yes	Typed, Manual Configuration
YARP	Soft for some modules	Yes (not Real-time)	++	Yes	Typed, Manual Configuration
ROS	None	Yes (not Real-time)	+++	Yes	Untyped, Manual Configuration

Table 5.2 – Comparison of robo-xeno with other robotic frameworks: Orocos [*Bruyninckx*, 2001], OpenRTM [*Ando et al.*, 2011], YARP [*Metta et al.*, 2006] and ROS [*Quigley et al.*, 2009]

There exists a number of alternatives to robo-xeno that aims to be generic robotic framework (see table 5.2): Orocos [*Bruyninckx*, 2001], OpenRTM [*Ando et al.*, 2011], YARP [*Metta et al.*, 2006] and ROS [*Quigley et al.*, 2009]. We present here a qualitative comparison of these frameworks. Indeed, re-implementing any robotic application and porting drivers to a different platform is a tedious tasks, and the resulting performance results could be biased if those implementations are not well-enough optimized. All the mentioned frameworks are modular and provide tools to provide components: an interface to define what a module do, what are

its inputs and outputs, and a transport mechanism that executes the data-flow between the module.

Regarding real-time constraints, these frameworks are not equal. For example `ROS` does not provide any tool to perform real-time operations. On the other hand, project like `Orocos` take this matter seriously, giving the user the possibility to define module with hard real-time constraints when it is run on Linux/RTAI systems. Furthermore several of these frameworks provide a middle-ware to make the data-flow to be transmitted over Transmission Communication Protocol (TCP) and User Datagram Protocol (UDP). As real-time networking is a complex task with these protocols — in the sense to meet deadline with certainty — none of this frameworks provide hard real-time communications. On the level of configuration, many of those framework provides typed data-flow. Meaning that each I/O of a module has a type information: position information, force, torque... When two modules are connected together, the type of data-flow is checked by the framework to detect any errors.

`robo-xeno` differentiates itself on two points. Firstly, all the configuration of the data-flow of these frameworks is manual, and when building an application, one must specify how all of these modules must be plugged together. Convention could easily be done in `robo-xeno` as it does not aim to build large-scale applications. It may also not be easy to define coherent conventions in those applications. Secondly `robo-xeno` is the only framework to enforce a clear separation of concern between modules. It asks for modules to either explicitly process data and not to deal with hardware, or to be responsible to ensure real-time capabilities of I/O data cycles. This architecture is really scalable, and a small team of developer could support controllers for more than eight robots (figure 5.1). Recently new `IOThread` where swiftly developed for `robo-xeno`. A module able to communicate with OptoForce sensors was developed in two work days, and another to read the ADC module of an embedded computer in a single day. This included not only to make the link between the drivers and the framework, but making these modules fully configurable, and producing the necessary examples and documentation. This scalability and ease of development is an advantage of this small, really focused framework.

On the other hand, the other frameworks aims to support large-scale projects, and most of them provides their user with a collection of re-usable components that implements a variety of algorithm widely used in robotics (Inverse Kinematics, Bayesian Filtering, SLAM...). It allows their user are able to rapidly build complex applications by pluging together those components. Furthermore, it may be questionable to prefer `robo-xeno`, a single person project, rather than any more widely accepted and developed framework for a new robotic projects. But what should persist from robo-xeno is not necessarily its implementation, but rather its analysis of purpose when it comes to build the architecture of this new robotics application.

5.3 Internal communication protocol of the Oncilla Robot

This section describes the communication protocol used between the different electronic components of the Oncilla robot. I decided to include this work in this thesis to follow two purposes. Firstly to present a technical achievement that was required to operate the Oncilla robot, and in second place to present to the reader some interesting issues that could arise at the integration stage of the development of any robot.

5.3.1 The Simple Binary Communication Protocol

5.3.1.1 Oncilla electronic architecture

The development team of the Oncilla robot chose to develop a modular architecture for its electronics, as depicted in figure 5.4. The main components are four "motordriver" boards that are responsible for controlling two brushless motors (for the Hip and Knee joint of each legs), reading three absolute rotational encoders (one hip and two "knee" joints) and one three-axis force sensor (leg load estimation at the scapula level). The purpose of these boards is a) to read all connected sensor data, b) to run low level Proportional–Integral-Derivative (PID) control of the motors, c) to perform trajectory tracking and startup position calibration procedures, d) to implement low-level security layers. Their main electronic processor is a dsPIC33FJ128MC804 microcontroller running low-level PID, trajectory tracking and sensor management, and two A3930 IC with their respective three phase inverter constituted of IRFR48Z MOSFETs that acts as current and voltage regulator of the brushless motors. These boards are in turn driven by a RB-110 embedded computer. It is a single board computer build around a DM&P 1.0 GHz i586 Vortex86DX System on Chip (SoC). This computer acts as the master of the robot and of the bus used for inter-device communication. It is responsible to run the low-level robot drivers, and to either run directly either low-level control algorithms designed by the users, or to act as a relay over IP network, to let one or several computers drive the robot remotely. These two modes correspond respectively to API level 0/1 and level 2 as described in *Nordmann et al.* [2013].

At the exception of the embedded computer itself, all electronic components were entirely developed by the AMARSi consortium itself, and no commercial modules or boards was used. Brushless motors offers top performance in terms of achievable torque and speeds, but off-the-shelf driver components are usually large and heavy. The decision to build a custom solution was therefore made in order to achieve a better performance over weight ratio. Therefore a complete freedom was left in regards of inter-device communication. We chose to rely on an $3.3\,\mathrm{Mbit\,s^{-1}}$ RS-485, half-duplex, multi-point, master — slave bus to transmit data between components, primarily motivated by the fact that some electronic devices developed for other robotic platforms (Roombots [*Spröwitz, Moeckel, Vespignani, Bonardi and Ijspeert*, 2014]) could be used to bootstrap the development. The communications over this bus are following the Simple Binary Communication Protocol (SBCP) specifications, which were designed specifically for this project. The SBCP bus controller is a dedicated board whose role is to

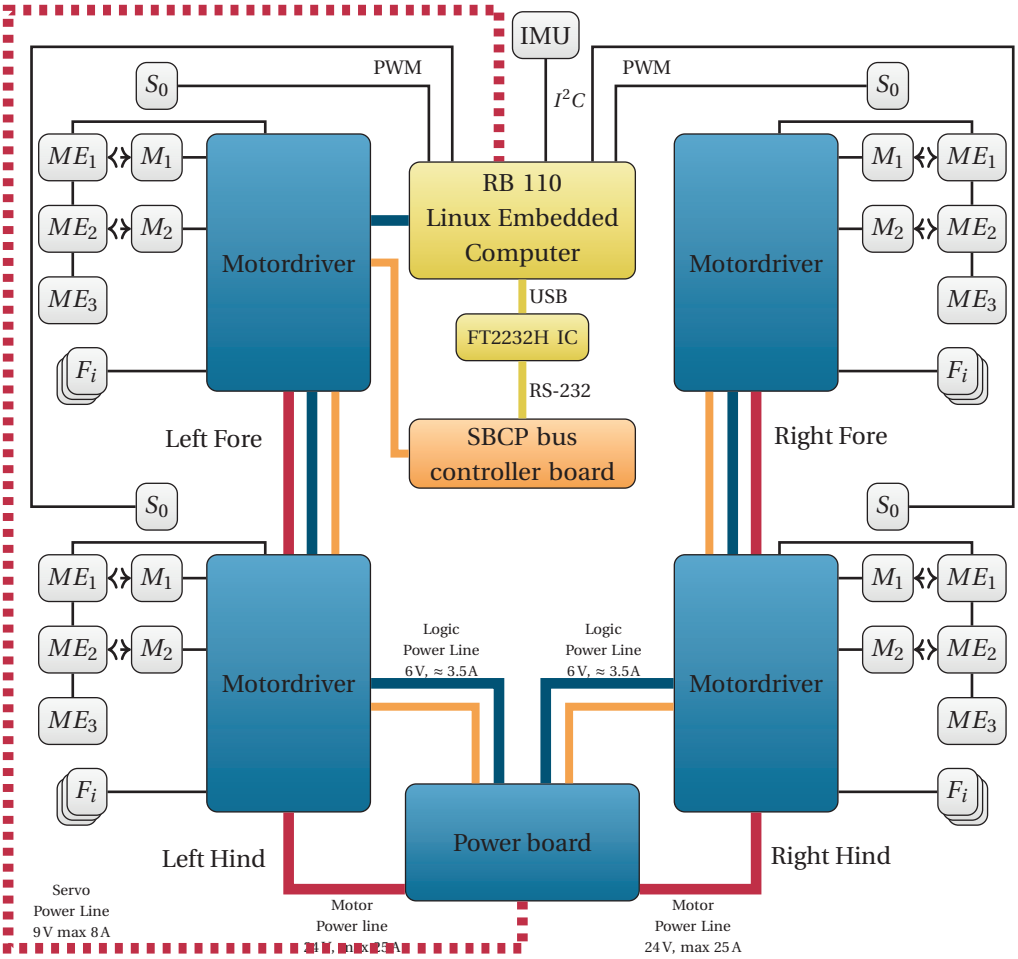


Figure 5.4 – Oncilla Electronic Architecture. Most of the sensors and actuators are driven by four electronic boards, one per leg. Each drives two brushless motors, three absolute encoders and one three-axis force sensor. They are driven over a RS-485 bus by a Linux embedded computer. This bus is running the SBCP protocol, specifically developed for the Oncilla robot, and is managed by a dedicated board. The motordrivers, power and SBCP bus controller boards were entirely developed by the AMARSi consortium.

efficiently manage communication over this bus (see section 5.3.2).

My role in this team was to develop the Linux real-time driver for this protocol, and to improve the board firmware and the protocol early implementation to fully meet our requirements. It resulted in four main contributions: the development and design of the SBCP bus controller, the improvement of the specification with low latency instruction (section 5.3.2), the standardization of the protocol implementation on both master and slave side, and finally fixing the integration of the motordriver firmware with the communication protocol (section 5.3.3).

5.3.1.2 Protocol requirements

The following specifications were chosen for this protocol:

Device agnostic protocol As seen in figure 5.4, on the Oncilla robot three different devices will communicate over the bus: the motordriver board, the powerboard that manages the distribution of electrical power on the robot, and the SBCP bus controller board. Since we use a multi-point topology all those components should use the same protocol.

Real-time high bandwidth throughput One of the goals of this protocol was initially to achieve a full trajectory tracking of the robot at 1 kHz. To achieve that, all motordriver boards would require to receive and send all data listed in table 5.3, i.e. 20 bytes per board, without considering any data overhead of the protocol itself. This sums up to an *effective* data rate of 800 kbit s^{-1} between five different components. Those numbers seems relatively small in comparison with the *net* data rate of common physical transport such as Universal Serial Bus (USB) (480 Mbit s^{-1} for USB 2.0), Ethernet (1 Gbit s^{-1}), WiFi (600 Mbit s^{-1} for 802.11n)... However a standard UART baud rate would often be 115 200 bit s^{-1} and very few Embedded Computers or SoC feature dedicated UART I/O over a few Mbit s^{-1}. Moreover, USB and WiFi protocols are not suited for real-time data transfer, due to their nature. Finally the microcontrollers used to control our board (from the dsPIC33 family) does not feature any Ethernet PHY (i.e the physical dedicated module decoding Ethernet frames), which explained why the UART was retained.

5.3.1.3 Description of the protocol

One of the specification protocols closest to ours that existed for a RS-485 physical layer at the start of the AMARSi project (2010) was the Dynamixel BioLoid protocol [*Robotis*, 2016]. Our SBCP protocol was designed as an extension of this protocol. Indeed the BioLoid one aims to drive a daisy chain of homogeneous devices, which can grow relatively large, but here we aim

Master to Slave		**Slave to master**	
Parameter	**Size**	**Parameter**	**Bytes**
Command (position, velocity or torque) for two motors	2 × 12 bits over 4 bytes	Actual position and of two motors	2 × 12 bits over 2 Bytes
		Value and status of three absolute encoders	3 × (14 + 2) bits over 6 bytes
		Three axis force sensor value	3 × 12 bits over 6 bytes

Table 5.3 – Listing of data to be transmitted at 1 kHz for each motordriver board.

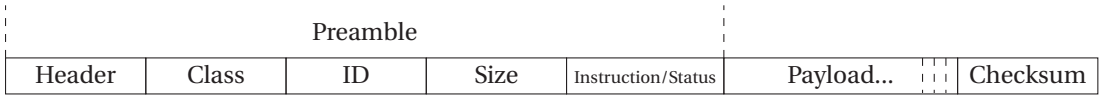


Figure 5.5 – Description of an SBCP packet. In packet sent by the master, class and ID designate the destination of the packet, and in response packets, they designate the source of the packet. Similarly, in the first case the fifth byte is an instruction byte and in the second case, an error return byte.

to drive a smaller, heterogeneous daisy chain of devices. Furthermore the BioLoid hardware already uses baud rate up to 1 Mbit s^{-1}, and it was originally thought that using hardware that could achieve baud rate above 2 Mbit s^{-1} would have been sufficient to meet our data transfer requirements (see section 5.3.2).

Both of those protocols could be seen has a simplified and downscaled version of the Ethernet protocol (figure 5.5). Each communication happens in frames or packets that are of variable sizes. Each packet has a preamble, that defines its size, type of instruction and destination. Since its a master — slave bus, only masters can initiate a communication, and the preamble of the packet contains its destination. Each communication initiated by the master expects a response from the slave, and therefore the response packets contains its source and not its destination, and an error status telling if the instruction was successful or not. After the preamble and the variable payload, a checksum byte is transmitted to check for communication errors. The checksum is the 8-bit sum of all bytes of the packet at the exception of the header.

Instead of a 16 bytes MAC addresses in Ethernet protocol to designate devices, SBCP uses two bytes. The first indicates the type of device (here motordriver, powerboard or SBCP bus controller) and the second is an ID uniquely identifying the targeted device. Masters of the SBCP bus has no address as they are unique.

On the bus arbitration size, only the master has the right to initiate a transfer, and the addressed slave is always expected to respond. Absence of responses are detected using timeouts, and any collision, when two slaves respond at the same time are expected to be detected by checksum mismatch.

As the opposite of the Ethernet protocol, the preamble of each SBCP packet features an instruction byte, which is specific to the application. In case of a response packet, this byte is instead a return byte, which indicates if the previous instruction was successful or a value describing a particular error. To help to standardize communication with each kind of devices, a small standard set of instructions are defined. Each devices has a table of registers that could be modified, and this set contains instructions such as getting or setting a parameter of this table. The first parameters of this table are also standardized, and contains the device ID, the version of the protocol used... Registers in this table could also be marked as non-volatile, so they could keep their value after a power reset, to retain information such as the device ID,

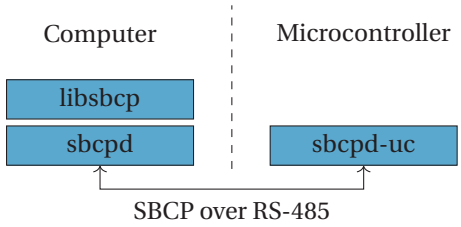


Figure 5.6 – Software implementation of the protocol

calibration information or security threshold values... Further information could be found in the documentation of each device and `libsbcp` [*Tuleu, Nordmann, Degrave and Ajallooeian*, 2012].

Finally the software implementation was split in three libraries. `sbcp-uc` [*Tuleu et al.*, 2013] is a C library that implements the slave protocol and is highly optimized for the Microchip dsPIC33 microcontroller family that is used in all of the Oncilla boards. On the master size, the implementation was split in two libraries. `sbcpd` [*Tuleu, Nordmann, Rückert, Ajallooeian and Wrede*, 2012] is a small library that is used to start transaction and responses over the SBCP bus. This library is only responsible for driving the UART in an efficient way and is completely application agnostic. On top of that `libsbcp` [*Tuleu, Nordmann, Degrave and Ajallooeian*, 2012] has been developed, which is a higher level C++ library that provides specific interface for each device supporting the SBCP protocol.

5.3.2 Dedicated bus management

The first issues had to deal with too small data throughput of the first protocol implementation and large latencies imposed by the use of USB to UART IC in the RB-110 to provide fast UART interfaces.

Considering the protocol overhead of 6 bytes per packet and the requirement on the data that need to be transmitted each millisecond, one can compute the theoretical minimum baud rate B_{th} for our application:

$$\begin{aligned} B_{th} &= N_{\text{boards}} \times (N_{\text{payload}} + 2 \times N_{\text{overhead}}) \times F_{\text{IO}} \times N_{\text{bits}} \\ &= 4 \times (20 + 2 \times 6) \times 1000 \times 10 \\ &= 1.28\,\text{Mbit}\,\text{s}^{-1} \end{aligned} \tag{5.1}$$

Where N_{boards} is the number of boards to address N_{payload} is the desired byte payload per board N_{overhead} is the number of bytes overhead per packet, F_{IO} the desired data cycle frequency and N_{bits} the number of bits to transmit one byte over an UART (here with one stop bit and no parity). Considering that all components on the bus feature UART that could communicate at $3.33\,\text{Mbit}\,\text{s}^{-1}$, a straightforward implementation, even considering large

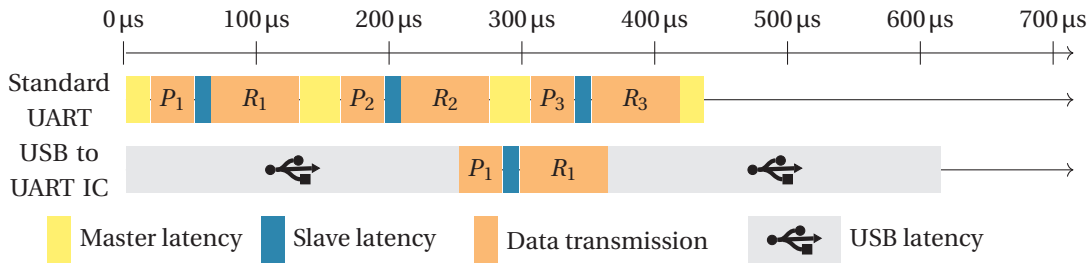


Figure 5.7 – Comparison between dedicated and USB to UART IC timing. The USB framing adds large ($\approx 250\,\mu s$) latencies.

delays to treat the information, was initially expected to meet the bandwidth specification. In practice, such an implementation could not perform full data tracking over approximatively 150 Hz, and could not be considered real-time as there was jitter in the data reception of the packet of the order of 500 µs.

The latter effect could be explained by the fact that the main loop of the motordriver board was fixed at 1 ms, and the packet processing was internally synchronized with this loop. This has been fixed by a more careful implementation of the protocol, the use of the DMA module of the dsPIC33 to avoid consumption of CPU cycle for the UART treatment, and the implementation of a low latency packet instruction. For this low-latency instruction, specific to each device, the response packet is precomputed at the time new data from sensor is received, so the response can instantaneously be sent back, without the need to reach the next start of the device main loop. Details on this implementation can be found in the documentation of `sbcp-uc`. Such a final implementation could reduce the response time of motordriver board down to a maximal time of 10 µs, which is well enough to ensure real-time data transfer.

However the maximal tracking frequency could not go above 400 Hz. This could not retraced back to any mistake in the firmware or Linux software implementation, but did arise because the RB110 computer relies on a FT2232H USB to Serial IC to provide its 12 $\text{Mbit}\,\text{s}^{-1}$ serial port. The use of such IC on single board computer has become more and common. As mentioned previously the USB is not tailored for fast real-time communication [*Korver*, 2003]: any communication over the bus is meant to happen during certain time frames. The USB controller is responsible to schedule which device is transmitting data during these frames. Those frames are scheduled each 125 µs for USB 2.0 [*USB 2.0 Promoters*, 2000], which is the protocol used by the FT2232H. With each UART transaction adding a 250 µs latency to poll the FT2232H, a slave could only be polled every 625 µs (see figure 5.7).

The solution was the development of the SBCP bus controller board. Indeed this board acts as a buffer and real-time manager of the SBCP bus. It can receive a number of packets from the master computer, and transmit them one by one to slaves. This reduces the latencies imposed by the USB to only 500 µs (see figure 5.8). By using a dedicated microcontroller running our

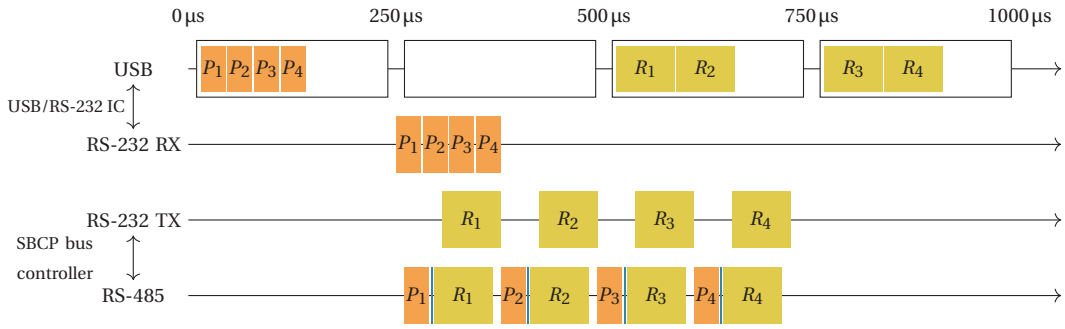


Figure 5.8 – SBCP communication with dedicated management and USB latencies. By using a SBCP bus controller that can bufferize and manage communication over the bus, the latencies introduced by USB framing can be divided by four.

algorithm in hard real-time, timeout for communication loss could be reduced down to 20 µs, to increase the data throughput when error arise on the bus, as the bus controller does not need to hang for at least 250 µs to detect timeout situations.

These optimizations led to an implementation that was able to transmit the required amount of data at 1 kHz, even if this data is transmitted over an USB bus. For that purpose, special care has been taken in the `sbcpd` API so the user could specify transfers that should happen in a single USB frame.

5.3.3 Interaction with the motor trajectory tracking

Transmitting the needed data with the sufficient speed was not sufficient to achieve our specification. In the final integration steps of the first prototype of the Oncilla robot, when testing the position tracking of the motors, those suffered hiccup and stutter every few seconds. This has been ultimately traced back to a bad interaction between the jitter in communication between the master and the slave, and the trajectory tracking processed on board of the motor.

The Oncilla brushless motor were chosen with a minimal rotor inertia, and the leg designed to be as lightweight as possible to ensure the fastest leg swinging time. However this low inertia caused problem for the trajectory tracking using a standard PID controller, as depicted in figure 5.9. If the trajectory is sent to the slave at a frequency below the PID update frequency, which could be the case when the RB-110 controller is not able to generate this trajectory fast enough, the motor would quickly jump to the next tracked point and stop, resulting in undesired motor stuttering. There was a large diversity of control approaches investigated by the AMARSi consortium [*Ajallooeian et al.*, 2010], and a large portion of them where not lightweight enough to produce trajectories sampled at 1 kHz in real-time. The solution to this problem was to fix the period at which the trajectory would be generated as a multiple of 1 ms and communicate it to the slave. Then the slave would limit the maximal achievable speed of the motor to ensure that it would reach the next tracked point at the same time step where he would need to track the next one (see “Smooth interpolation” in figure 5.9). This mode was developed independently on

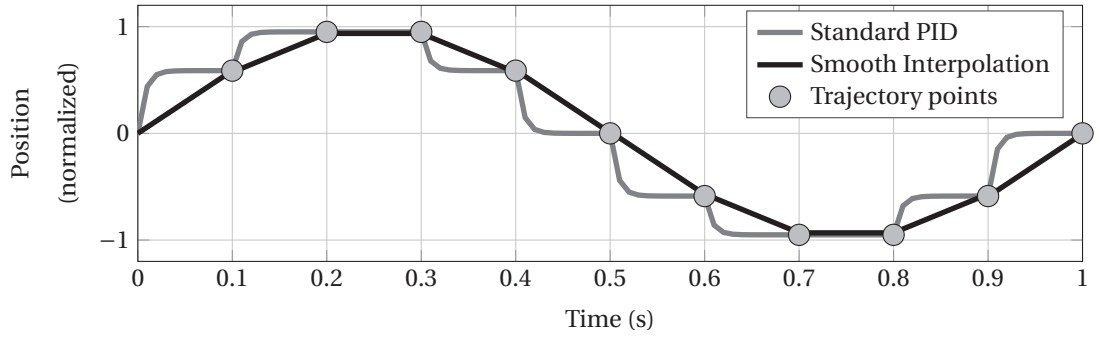
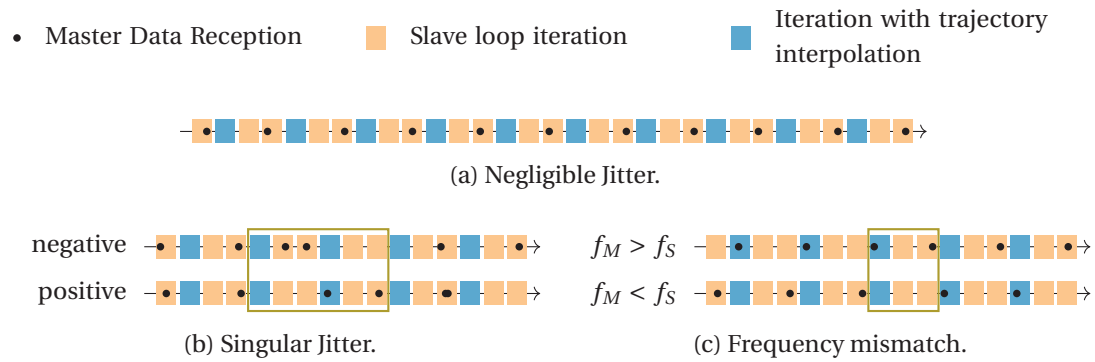


Figure 5.9 – Oncilla motordriver trajectory tracking, at 100 Hz. Due to the low inertia of the brushless motor, a PID controller can introduce a lot of motor stuttering. By asking the user at which frequency the data point will be sent, a smooth interpolation can be performed by limiting the maximal speed between two trajectory points.

the motordriver board, using trajectories computed by the motordriver board, and it removed completely motor stuttering for trajectory tracking as low as 50 Hz.

On the other end, communication on the SBCP bus would always suffer jitter from the point of view of the slave (see figure 5.10). Any communication error would delay or even advance the time arrival of a new trajectory point (figure 5.10b). Furthermore small inaccuracies in the devices clock frequency make them to slowly desynchronize themselves, and in a few seconds, one time step would have no point to treat or have two to treat at the same time. Both of these issues could not be practically avoided.

The solution to deal with this two situations was to add some internal buffer to the motordriver board to treat data overflow situations (when the rate at which the master is sending data is slightly higher than the slaves clock frequency) and to use a simple heuristic for underflow situations. In the latter case, a new point would be computed keeping the previous speed divided by two, and when the next tracking point would be received, the internal clock of the interpolation would be reset to treat the point immediately. To avoid filling completely the internal buffer, once it starts to fill up, the motordriver would reduce the interpolation period until the buffer becomes empty again. These two mechanisms could easily deal with exceptional communication errors and equally to frequency mismatch by smoothly re-synchronizing the slave's internal clock to the master one. However since the smooth interpolation period should be able to be reduced, it should always be at least two PID update time long, limiting the trajectory tracking to be achieved without artifact at a maximal rate of 500 Hz. These heuristics did remove any artifact in the trajectory control. Communication and trajectory errors for a typical Oncilla robot are reported in table 5.4.



(a) Negligible Jitter.

(b) Singular Jitter.

(c) Frequency mismatch.

Figure 5.10 – Illustration of different jitter issues in trajectory tracking. Steps that suffers tracking interpolation issues are marked with a box. a: If jitter is relatively small, it has no impact as data always arrive during the step prior to trajectory interpolation. b: high jitter can arise at singular point as data arrive at slave well before (above) or well after the deadline (below). In both case an interpolation iteration treats two point at once and no point at all. c: Because of frequencies mismatch between master (f_M) and slave (f_S), at some point in time two trajectory points will be treated at once (when $f_M > f_S$) or one iteration will miss a point (when $f_M < f_S$).

5.3.4 Discussion

The USB latencies and trajectory tracking issues are illustrative that integration of different subprojects to constitute a robot is a complicated step, and this complexity is often underestimated by the designers. In the first case, a protocol was designed and successfully tested at a slower rate, assuming that a simple change of baud rate would be sufficient to meet the requirements. When it was used with the embedded hardware, that was using an USB to UART converter IC its performances dropped well below the desired specifications, and it could not met those without complex and costly optimization of the firmware. In the second, while trying to solve a local problem, i.e. reducing motor stutter, an interpendency with an-

Test time	2000 s	
Packets transmitted	4 000 108	
Average cycle frequency	500.0135 Hz	
Communication Success Rate	100 %	
Board ID	**Missed points**	**Filled Buffer steps**
1	0	1528
2	0	1498
3	0	1534
4	0	1524

Table 5.4 – Test of communication and trajectory tracking during 2000 s on an Oncilla robot. The slave and master where reporting every exceptional situation. Without the heuristic described in section 5.3.3, the motor will show hiccup every few seconds due to data overflow situation.

Criterion	SBCP	MIT-Cheetah	EtherCat
Real-time capacity	≈ 250 µs delays (USB) 2 µs jitter	< 10 µs jitter	< 1 µs jitter
Effective Data Rate	800 kbit s^{-1}	> 3.4 Mbit s^{-1}	> 10 Mbit s^{-1}
PCB Space on Slaves	none	none	large (dedicated IC or specialized microcontroller, Ethernet PHY)
Controller Size	Small PCB (3.5 cm × 1.5 cm)	Large (FPGA + compactRIO 9082)	none — using host Ethernet PHY
Master CPU requirements	Light, only UART driver needed	Complex RTOS integration	Complex RTOS integration with real-time networking
Slave CPU Requirements	Complex module integration: UART, DMA…	none	Complex but different implementation available
Commercial solutions	Non available	Non available	Available

Table 5.5 – Qualitative comparison of SBCP with other communication protocols.

other subsystem, the inter-device communication, was introduced because some non-trivial, unchecked assumption was made, i.e. the fact that the embedded computer clock would be exactly synchronized with the slave ones.

Nonetheless, these issues have been fixed and a reliable solution was developed and deployed over five different robots. It was at the cost of not meeting the initial specification of 1 kHz but only 500 Hz trajectory tracking. In comparison with the RC servo used in the cheetah-cub robot, it is still almost an order of magnitude faster I/O cycle.

The improvement of the initial protocol and its integration was quite an investment and it would be interesting to compare it to other solutions, as shown in table 5.5. An interesting communication architecture is the one adopted by the MiT Cheetah team [*Seok, Hyun, Park, Otten and Kim*, 2014]. Instead of relying on a single bus topology to control all their components, like SBCP, they use a single Field-Programmable Gate Array (FPGA) chip with a high number of programmable I/O and that have a single link to each sensor and actuator using a standard communication protocol such as RS-232, SPI Inter-Integrated Circuit (I^2C), CAN…All the synchronization is left to the FPGA, efficiently communicating with the Embedded computer CPU and providing I/O cycle with negligible jitter up to 4 kHz over 69 sensors and 16 actuators.

Finally one other solution would be to use nowadays standard in automotive communications. For example the EtherCat protocol is a modification of the Ethernet standard to meet soft and hard real-time applications [*EtherCat Technology Group*, 2008]. It reuses the same hardware than the Ethernet protocol, at the exception that slaves of the bus have two Ethernet PHY and are linked in a daisy chain. This protocol has specifications that are well over our

requirements, ensuring data cycle frequencies over 10 kHz and jitter below 1 µs. Since a few years, more and more commercial solutions are available to simplify the implementation of EtherCat on embedded devices. On the master or host side, Open Source EtherCat solutions are available, such as Etherlab or SOEM, that could be run over a real-time kernel using Xenomai and RTnet [*Moon et al.*, 2009]. With todays availability of these tools, it seems that such technology should be the first option to consider for a project with specification similar to the Oncilla robot.

6 Conclusion

In this thesis we aimed at push further the extents of what low-cost lightweight quadruped robots could achieve in terms of anticipated locomotion. Looking at how other successful robotic platforms achieve outstanding results in term of agility, robustness, and ability to move in complex environments, we have seen that most of them relies on body-wide, complex and precise model-based control architectures. This approach requires these projects to use or develop new solutions in term of actuation or sensing in order to be performant. On the other hand *Spröwitz et al.* [2013] showed that by using off-the-shelve components and a strongly bioinspired mechanical design, we could achieve state of the art dynamic locomotion. This seminal work set up our main motivation to address complex locomotion scenarios through the prism of a strongly bioinspired control architecture, i.e. by modulating a CPG model and avoiding to rely on a global precise model-based control approach. This in order to show, that in some situations, we could build low-cost robots as efficient as their high-end counterparts. In order to achieve this ambitious goal, we addressed three questions related to three main domains: modeling, control, and software development. As the research project progressed, a fourth question emerged: how could we estimate mechanical load on the leg using tactile sensing? This chapter summarizes all of our research questions and it also recommends topics for future research.

How can we accurately model the ASLP leg?

Summary of contributions To answer this question, we compared three different RBD models. The first of these used a formulation of the equation of motions with maximized coordinates. This approach was motivated by the fact that it was trivial using this formalism to model the leg's closed kinematic loop, and many robotic simulation software packages also employ it. The second model used instead a generalized coordinates formalism, as its is common practice to build dynamical models of robot for control. As it was relatively easy to analytically solve the ASLP kinematic loops, we proposed employing a serial model, requiring state-dependent joint end-limit constraints. We also proposed an extension of the method for solving unilateral constant constraints, namely a LCP formulation, for the state-dependent

case. Finally, to avoid having to deal with numerical instabilities introduced by high-frequency mechanical oscillations, we proposed a third method. It still relied on generalized coordinates, but which it ignored all the leg segment's mass and concentrated all of the leg inertia in the foot. We demonstrated that the dynamic of the leg could be solved off-line as the solution of a quadratic problem with non-linear equality constraints.

Discussion Three main points summarize the answers to this research question:

- The *quantitative* accuracy of the maximized approach is questionable. *Spröwitz et al.'s* [2013] results hinted that this would be the case, as they indicates more than an order of magnitude of difference in Cost of Transport (CoT) between the simulated model and the real hardware for similar gaits. When testing the method's numerical stability with different leg stiffness levels we find that there is no variation in the simulation's numerical stability in regards to the spring stiffness, as there normally should be. This finding relates to the fact that maximized coordinates approaches are heavily reliant on numerically solving the system constraints, and the solvers used by most RBD software packages prioritize numerical stability over physical accuracy. However, if a quantitative result (e.g. determining the optimal size of a motor, or the spring constant of the leg) is not required, these models are qualitatively equivalent to the real robot. *Spröwitz et al.* [2013] illustrates that similar relations between the robot's speed, gait frequency, hip swing amplitude and desired duty ratio are observable in the simulated Cheetah-Cub model expressed with maximized coordinates and the real robot. A similar Oncilla model was also used to prototype and test robust locomotion controllers [*Gay*, 2014; *Ajallooeian et al.*, 2014], prior to their implementation in real robots [*Ajallooeian*, 2015].

- The generalized coordinate approach can potentially be very accurate with a proper parameter identification, but it is quite computationally inefficient. To achieve numerically stable simulations with the parameters extracted from Cheetah-Cub CAD data, an integration time-step of less than 40 µs should be used, which makes the simulation almost 25 times slower than real-time when performed on a standard desktop computer. We hypothesize that since the model requires relatively stiff springs to act on the small masses of the leg segments, we simulate high-frequency oscillators, which requires a small time-step to be integrated without numerical issues using an explicit Euler integration. This hypothesis is confirmed when we compare the model's numerical stability at various leg spring stiffness values.

- Our mass-less segment assumption yielded promising results, especially in terms of computational efficiency. By reducing the problem size as we remove intermediary bodies, and by solving an off-line computation of the non-linear quadratic problem, we could maintain a low computation time of 357 µs for a time-step integration, while increasing the numerically stable time-step limit to 1.69 ms. However these are simply initial results, as this model still lacks many desired features, such as

an asymmetrical knee actuation and internal friction in the leg. Introducing these new traits would render the quadratic problem much more complex. As a result, we might be forced to solve the problem online, which would require much more computational power.

Future directions Our first recommendation regarding building an accurate model of the ASLP leg, is to further develop the promising mass-less generalized coordinate model. A first step should be introducing the asymmetrical knee actuation, which would require a new set of constraints on the quadratic problem, as well as two more input variables for the resulting off-line look-up table: the position and the torque of the knee actuator. The second insight, is regarding the use of maximized coordinate models of the ASLP-equipped robots. The quantitative accuracy of such model is questionable, and their level of detail requires a lot of computational power: LCP solvers are polynomial in time with the number of bodies and constraints. Therefore it might be wiser to reduce the complexity and use simpler legged robot models — a two-segmented leg with a serial compliance in the knee joint — to prototype new control approaches, as that would drastically cut down the amount of required computation.

How can we develop software capable of addressing the specificity of a variety of robots?

Summary of contributions We described a personal methodology based on a hierarchical analysis of the roles and purposes of the objects or people interacting with the software. This methodology aims to build modular, scalable and easily maintainable software architectures, by forcing each module to endorse a *single responsibility*. This helped to decouple modules, preventing any changes to one of them from affecting the others. This methodology, when applied to the problem of developing a software framework to support the implementation of real-time, low-level, robotic controller, took the form of `robo-xeno`. At the most abstract level, `robo-xeno` primarily separates the people interacting with the robot into two orthogonal roles: the *robot user* and the *robot maintainer*. The latter is concerned with ensuring that the internal components of the robot exhibit the correct behavior in terms of communication, drivers and timing constraints. The former is only concerned with the robot's general behavior, i.e. the robot actions at any time given its inputs. This framework was ported on up to eight different robots and the smallest data I/O loop time it could reach was 2 ms. Different low-level controllers were implemented on these robots, an the hand-tuning of their numerous parameters was facilitated with `robo-xeno` dedicated user interface.

Discussion In term of the effectiveness of our approach, we qualitatively compare it to the many robotic frameworks that were developed prior to, or alongside, `robo-xeno`. The scope of those frameworks exceeds that of `robo-xeno`, as they aim to build robotic applications running over several program or computers. The larger ones, like `ROS` or `Orocos` seek to provide their users with a large library of reusable components that can be plugged together to

rapidly build large-scale applications, which is not the purpose of robo-xeno. Furthermore, very few of these platforms provide hard or soft real-time capabilities for their modules, while robo-xeno specifically aims to build soft real-time applications. All of these platforms, including robo-xeno provide tools and concepts for creating a modular architecture. However, the enforcement to the developer of the separations of responsibilities, is not present in any other frameworks than robo-xeno, neither is the relatively new concept of convention over configuration.

Future insights The work begun with robo-xeno could persist beyond this project in different ways. As it focuses heavily on the exclusive development of low-level controller, we could make it interface with any other larger-scale framework, including YARP, Orocos or ROS. In that manner, we could gain access to the large collections of modules and algorithms that these projects maintains. Achieved at the rx level, this link would not require real-time constraints, as those might be difficult to integrate within the larger-scale framework. An approach that is well-aligned with this thesis' bioinspired locomotor architecture: robo-xeno is taking care of the fast (i.e. requiring hard real-time constraints) low-level, simpler controller, which stay close to the hardware, and the other framework take care to implement the higher-level complex algorithms, that would run at a slower rate. Finally, for any new robotic project at the point of selecting a robotic framework, it would be questionable to employ and trust a single-person project over a more broadly developed and maintained framework. We agree that the key takeaway from robo-xeno is not necessarily its implementation, but rather its architecture and its proposed separation of concerns. In our opinion, adhering to the previously discussed design rules, would make any new robotic application inherently more scalable and maintainable.

How can we estimate the leg-loads using tactile sensors on the ASLP leg?

Summary of contributions This question emerged as we want to implement the Tegotae feedback rule in the Cheetah-Cub robot, as the other methods we had tried presented major inconveniences. Due to the specific shape requirements and to minimize the inertia to the foot, we decided to investigate whether a piezoresistive sensor, made of Velostat sheets would be a feasible option. We screened the relevant factors among the sensor design parameters (size,shape,glue), and concluded that for estimating static forces, we would be able to find a design suited our needs. However, due to hysteresis in the sensor responses, we were unable to build an accurate estimation of the leg load. We then proposed a novel approach, that used a second piezoelectric dynamic sensor together with the first one. The data from the two sensors was then fused using an EKF. We tested this filtering approach both in a simulation and in a custom-designed test bench, using the response model we empirically characterized for both sensors.

Discussion We demonstrates that this novel approach is able to more accurately estimate the normal load with the combination of a stress rate and a static tactile sensor. The output of the filter, applied on simulated data, highlights the necessity of fusing both type of information to build a stable and accurate estimation. However, practical experiments do not demonstrate the same level of strength. The EKF can be tuned, after extensive optimization of its gains, to yield a stable estimation of the force. An analysis of the measurement matrices, indicates that post-optimization, the filter was strongly biased towards one or the other sensor. We connect this issue to two technical elements. Firstly, the piezoresistive response lacks repetitivity, as its response changes over the course of a few minutes. Secondly the piezoelectric sensor was not strictly designed to be a rate sensor, and was instead intended to be a dynamic force sensor. Finally, our approach had implications regarding the choice of a piezoresistive transduction for our static sensor. It may be possible that a capacitive transduction might mean less hysteresis and fewer dynamic response issues, that may not even need to be fused with a stress rate sensor.

Future directions Several technical issues need to be solved in order to test our approach's effectiveness for real applications. First the amplification electronic of the piezoelectric sensor could be changed to transform it into the required stress-rate sensor. Secondly, the proposed approach would still be valid if we would change the static sensor to another transduction type. Since, according to *Kappassov et al.* [2015], capacitive sensors stills suffer from hysteresis and bandwidth issue, and we should investigate how effective the combination with a stress rate sensor would be at mitigating these two limitations .

At which extend, can the combination of CPG, sensorimotor coordination and bioinspired mechanics achieve complex locomotion scenarios?

Summary of contributions We narrowed the investigation of this broad question to the simpler problem of dynamic footstep placement, and we addressed it in two steps. First, we examined whether the open-loop CPG without sensorimotor coordination proposed by *Spröwitz et al.* [2013] would be sufficient for tackling this problem. For that purpose, we proposed a new CPG computational model. Instead of encoding and directly modulating the joint space trajectories, this CPG encodes bioinspired foot locus trajectories in the hip parasagital plane. To investigate the effect of sensorimotor coordination, we were one of the first to implement the Tegotae feedback rule on a quadruped with two degrees of freedom. We achieved this by isolating each oscillator from our original CPG model and individually synchronizing their phases with the mechanical load of the leg they drove. These approaches were compared in terms of their performances on the task of dynamic footstep placement. Specifically we assessed the displacement of the footstep after an online change in the desired step length parameters of both CPG models.

Discussion Our results indicated that a purely open-loop approach would probably not be well-suited for managing dynamic footstep placement. With this approach, the displacement of the footsteps is quite erratic when the step length changed . This result invalidates our assumptions that the self-stabilization of the ASLP leg would be sufficient for handling the transient state induced by the desired pattern change and that the robot would quickly reach a new limit cycle. However, the parameterization of a bioinspired foot locus proved to be quite helpful for hand-tuning the gait parameters on the real robot, as the parameters felt "less correlated". In other words, modifying one parameter did not necessarily mean that the others need to be adjusted to produce stable locomotion. Furthermore, the Tegotae rule approach seems to be an appropriate technique for dynamically adjusting the footstep placements. Indeed, modification of the desired leg length produces very regular and predictable changes in the footstep position, with a linear relationships between this displacement and both the desired length modification and the numbers of step taken after that change. This means that the strong interaction between CPG, sensorimotor coordination and the mechanical behavior were able to maintain stable locomotion patterns. In other words, an higher-level, body-wide, balance control algorithm might not be required to address footstep placement. Finally it is worth noting than further research is still needed to determine whether this approach would still be valid for obstacle avoidance. Indeed, in our experiment, the desired step lengths were modified by the same amount for all of the leg. To avoid a real obstacle, it may be necessary to displace the expected footstep position at a different distance for each leg. This may induce a larger perturbation of the gait limit cycle and so verifying whether our approach would still be able to produce stable and predictable locomotion is necessary.

Future directions Our results suggest that open-loop CPG alone cannot effectively deal with transients when it comes to adapting the produced kinematic pattern. However, there is another situation in which CPGs are very useful: gait transitions. They can be used to produce smooth phase transitions between two desired locomotion patterns. In nature, the criterion that defines why an animal uses a particular gait is not yet well-known. It could be that the animals optimize the long-term CoT [*Alexander*, 1989], metabolic cost, or minimize peak musculoskeletal forces [*Wickler et al.*, 2003]. Since the Tegotae rule is a bottom-up approach and gait emerges from the morphology or the dynamical state of the robot, it would be interesting to examine how these transitions are connected to the robot's CoT or mechanical forces, and to then search for similarities in nature.

Secondly, one interesting use of the Tegotae rule would be to apply it to morphological changes. Quadrupeds,especially dogs, are able to rapidly adapt after a leg amputation, and they display different strategies depending on the injury [*Jarvis et al.*, 2013; *Hogy et al.*, 2013]. It would be interesting to see if the Tegotae rule, when applied to an amputee robot, would still produce efficient locomotion patterns. For such scenarios, future researchers could also investigate whether the controller would need to be tuned, and whether the resultant gaits would be related to those reported in animals. If successful, the results could shed light on the animal locomotor system, and they would also point towards a simple approach to hardware failure recovery in legged robots.

Finally, this study did not consider how a vision system could be integrated to autonomously handle complex locomotion scenarios. However, if off-the-shelf components (e.g. commercial RGBD sensors) were to be employed for that purpose, one problem would be that those systems are tuned to detect large, human-scale objects. They would be of little use in detecting obstacles for a 15 cm tall robot. We suggest that those obstacles could be identified via optical flow, as that approach is more suited for detecting closer objects than more distant ones.

Low-Cost Bioinspired Robots as High-End Alternative

Initially *Spröwitz et al.* [2013] showed that bioinspired lightweight robots could display dynamic trot gait locomotion and *Ajallooeian* [2015] showed that they could perform blind robust locomotion. In this work, we showed that they are close to be able to perform dynamic footstep placements with only the combination of a CPG model, strong sensorimotor coordination and bioinspired mechanics. We also provided reliable software and firmware that is well-suited to implement our proposed bioinspired control architecture. We presented an original method to estimate GRFs through tactile sensing in order to preserve the leg low inertia footprint. Finally we proposed RBD models that help to the prototyping of new controllers for the ASLP legged robot, and showed how we could make those model more accurate in order to ease the creation of new designs.

All of these results are small steps towards showing that low-cost robots could, in specific situations be an alternative to high-end quadruped platform. Of course, the latter would always have a upper end in terms of versatility or ability to handle complex tasks. But the design of low-cost, goal oriented, robots would be a key step to promote the widespread use of mobile robots in search-and-rescue missions and hazardous environment exploration. In turn, this would help to reduce the risks that rescuers and other workers face in these situations. In a personnal, philosophical and ethical opinion, performing tasks where human life would otherwise be at risk, is one of the best use humanity could make of robotics.

A Forward and Inverse Kinematics of the Advanced Spring Loaded Pantograph (ASLP)

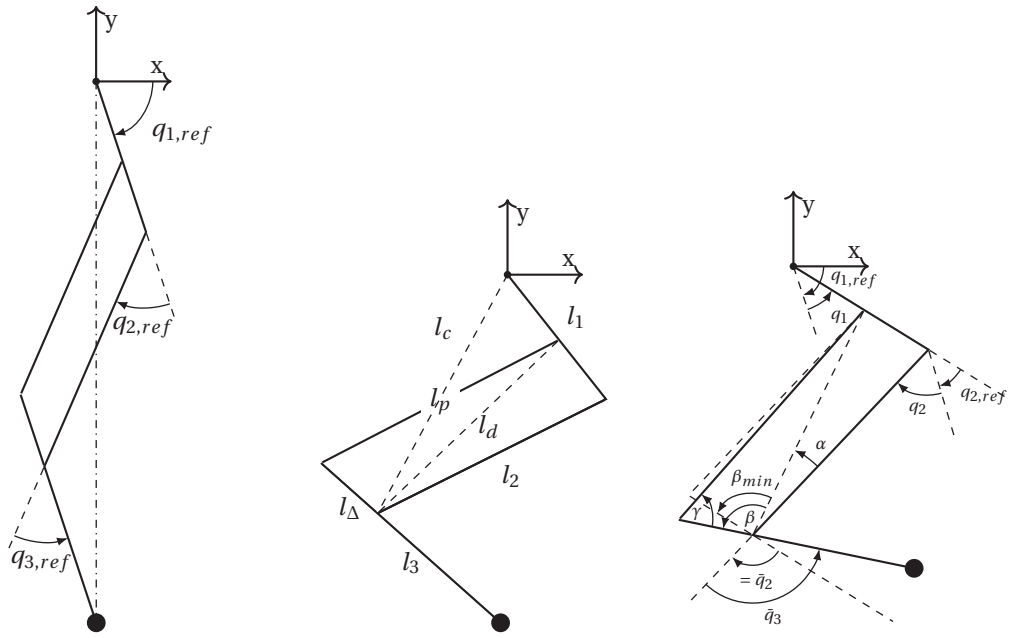
This appendix describes some mathematical derivation of the ASLP leg. For that purpose the parameter describing the system are intoduce in figure A.1 and tables A.1 A.2 and A.3. The pantgraph mechanism has several closed kinematic loop, and therefore equation sometimes can have several solutions. All formulas describe in this will always choose the configuration depicted in figure A.1a. Furthermore, if we use zero values for each joint, it would lead to a non achievable configuration mechanically. To simplify the operation, calibration of the robots, and specification of various model in simulation, the reference angles in figure A.1a are introduced. Those correspond to the segment angles when the leg is extended as it mechanical maximum, and the hip joint axis lies vertically above the foot joint. We note:

$$\forall i \in \{1,2,3\} : \bar{q}_i = q_i + q_{i,ref} \tag{A.1}$$

$\bar{q}_i$ is the angle that would be used in any robotic coordinate convention.

Fixed Parameters		Variable Parameters	
Name	**Description**	**Name**	**Description**
l_1	Length of the proximal segement	l_d	Length of the diagonal of the open pantograph
l_2	Length of the middle segment	l_c	Length of the cable
l_3	Length of the distal segment	l_p	Length of the opened segment of the pantograph.
l_Δ	Spacing between "parallel" segment of the pantograph		
r	Radius of the pulley (only for Oncilla robot)		

Table A.1 – Description of the different leg length



(a) reference angles (b) Lengths (c) Angles

Figure A.1 – Notation for Advanced Spring Loaded Pantograph (ASLP) leg parameters. Origin is taken in the center of the hip joint. (a) Reference angles are defined as those when the leg is fully extended, the foot above the hip joint, and the inner leg segments parallel ($q_{2,ref} = -q_{3,ref}$). (b) Lengths. l_c is the length of the cable, l_d and l_p are related respectively to the diagonal and parallel spring. (c) Angles α, β and γ are named for the resolution.

A.1 Inverse Kinematic

The inverse kinematics consist of finding the value of the the geometric control variable of the leg q_1 and l_c from the position of the foot (x_F, y_F)in the hip referential. The system is under actuated, as there is no possibility to control the angle of q_3 which is passively actuated by the parallel spring. Therefore to simplify the problem, we make the assumption to derive the following formulas that the pantograph is closed, i.e. that $l_p = l_2$. Considering the total L_L leg length:

$$L_L = \sqrt{x_F^2 + y_F^2} \tag{A.2}$$

Since the pantograph is closed l_1 and l_3 are parallel. To relate the L_l to q_2 since they are parallel, we can use the equivalent two segmented leg depicted in figure A.2. Then we have:

$$L_L^2 = l_2^2 + (l_1 + l_3 - l_\Delta)^2 - 2l_2(l_1 + l_3 - l_\Delta)\cos(\pi - \bar{q}_2) \tag{A.3}$$

Therefore q_2 depends solely on the desired leg length. Furthermore q_2 can be expressed

Fixed Parameters		Variable Parameters	
Name	**Description**	**Name**	**Description**
$q_{1,ref}$	Reference angle for q_1	q_1	Angle at trunk and l_1 junction
$q_{2,ref}$	Reference angle for q_2	q_2	Angle at l_1 and l_2 junction
$q_{3,ref}$	Reference angle for q_3	q_3	Angle at l_2 and l_3 junction
$\beta_{\min}$	β value when the pantograph is closed (i.e. l_p and l_2 parallel)	γ	Angle between l_p and l_3
		α	Angle of the force of the diagonal spring acting on l_2
		β	Angle between l_p and l_3. $\beta - \beta_{\min}$ is the opening of the pantograph
		γ	Angle of the force of the parallel spring acting on l_3

Table A.2 – Description of the different angles. γ and β does not ahve a perticular meaning, they are introduce to develop mathematical formulation

Fixed Parameters		Variable Parameters	
Name	**Description**	**Name**	**Description**
k_d	Stiffness of the diagonal spring	τ_d	Torque applied by the diagonal spring at l_1 — l_2 junction
k_p	Stiffness of the parallel spring	τ_p	Torque applied by the diagonal sprint at l_2 — l_3 junction
p_d	Diagonal spring pre-compression		
p_p	Parallel spring precompression		

Table A.3 – Description of the dynamic parameters

solely on the cable length l_c:

$$l_c^2 = l_1^2 + l_2^2 - 2 l_1 l_2 \cos(\pi - \bar{q}_2) \tag{A.4}$$

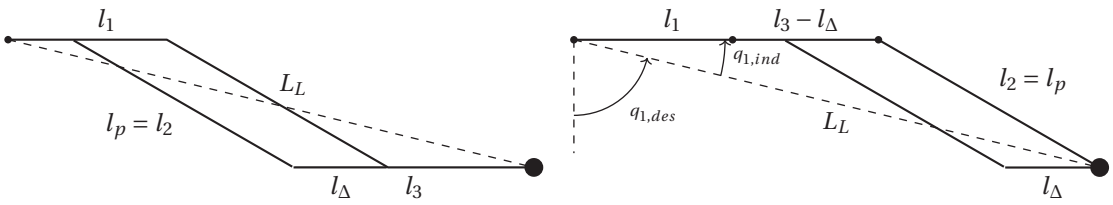
This gives the expression of q_2 from the cable length l_c:

$$q_2(l_c) = -\arccos\left(\frac{l_c^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) - q_{2,ref} \tag{A.5}$$

This equation is not needed for the inverse kinematic, but will be useful for other calculus. By combining (A.2) (A.3) and (A.4):

$$l_c(x_F, y_F) = \sqrt{l_1^2 + l_2^2 - \frac{l_1\left(x_F^2 + y_F^2 - l_2^2 - (l_1 + l_3 - l_\Delta)^2\right)}{l_1 + l_3 - l_\Delta}} \tag{A.6}$$

Finally the hip angle can also be expressed in two part (see figure A.2b): the desired leg



(a) ASLP closed pantograph (b) two segmented ASLP equivalent

Figure A.2 – Advanced Spring Loaded Pantograph (ASLP) leg equivalent simplification. If the pantograph mechanism is closed, for kinematic derivation, the ASLP leg (a) is equivalent to the two segmented version shown in (b): the l_3 segment is reduced to l_Δ and $l_3 - l_\Delta$ is added to l_1

angle $q_{1,des}$ and the one induced by the mechanism $q_{1,ind}$

$$\bar{q}_1 = -\frac{\pi}{2} + q_{1,des} + q_{1,ind} \tag{A.7}$$

with $q_{1,des} = \arctan(x_F/y_F)$, and $q_{1,ind}$:

$$q_{1,ind} = \arccos\left(\frac{l_2^2 - (l_1 + l_3 - l_\Delta)^2 - L_L^2}{2L_L(l_1 + l_3 - l_\Delta)}\right) \tag{A.8}$$

Finally the hip angle is defined:

$$q_1(x_F, y_F) = \arctan\left(\frac{x_F}{y_F}\right) + \arccos\left(\frac{l_2^2 - (l_1 + l_3 - l_\Delta)^2 - L_L^2}{2L_L(l_1 + l_3 - l_\Delta)}\right) - \frac{\pi}{2} - q_{1,ref} \tag{A.9}$$

A.2 Reference Angles

The reference angles are defined when the leg is fully extended, and the hip joint is lying vertically above the foot. Those angle are uselful to have the same configuration defined for the actual robot and the simulation for example. It is quite inconvenient to measure angle on the actual robot, as it lacks angle reference position. However, measuring the length of the cable is pretty easy, when no pulley is used. Therefore it is needed to express the reference angle from the length of the cable, and determine experimentally and or using CAD model the range of cable length to have the same limitation between simulation and real robot.

For q_2, (A.5) can be used to determine the reference angle from the maximal cable length

$l_{c,\max}$:

$$q_{2,ref} = -\arccos\left(\frac{l_{c,\max}^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) \tag{A.10}$$

For $q_{1,ref}$, it could be deduced from‘(A.8) and (A.3):

$$L_{L,\max} = l_2^2 + (l_1 + l_3 - l_\Delta)^2 + l_2(l_1 + l_3 - l_\Delta)\cos(q_{2,ref}) \tag{A.11}$$

$$q_{1,ref} = \arccos\left(\frac{l_2^2 - (l_1 + l_3 - l_\Delta)^2 - L_{L,\max}^2}{2L_{L,\max}(l_1 + l_3 - l_\Delta)}\right) - \frac{\pi}{2} \tag{A.12}$$

Finally, since at the reference position the pantograph is closed, we have:

$$q_{3,ref} = -q_{2,ref} \tag{A.13}$$

A.3 q_3 End-Limit Angles

One requirement of the second approach described in section 2.3 is the value of the q_3 joint limit. Here the assumption of a closed pantograph is not held anymore, but $l_p \geq l_2$ is assumed. On the actual leg, the joint end limit are enforced by the end limits of the prismatic joint l_p:

$$l_2 \leq l_p \leq l_2 + l_{p,\max} \tag{A.14}$$

To transpose this joint limit to q_3, we need to solve the extremal value for β from the values of l_p. Using the Al-Kashi theorem (law of cosine) on the triangle $\{l_\Delta, l_p, l_d\}$ we have:

$$\beta(q_2, l_p) = \arccos\left(\frac{l_d^2(l_c) + l_\Delta^2 - l_2^2}{2 l_\Delta l_d(l_c)}\right) \tag{A.15}$$

This depends on l_d which can be expressed in terms of q_2 or l_c:

$$l_d(l_c) = \sqrt{l_2^2 + l_\Delta^2 + 2 l_2 l_\Delta \cos\left(q_2(l_c) + q_{2,ref}\right)} \tag{A.16}$$

Using the fact that when the pantograph is close l_2 and l_p are parallel, we have the lower

bound of q_3, $q_{3,\min}$ that depends on q_2:

$$q_{3,\min}(q_2) = -q_2 \tag{A.17}$$

We need to deduce the maximal bound from the relation between q_2 q_3 and β (see figure A.1c):

$$\beta = \bar{q}_2 + \bar{q}_3 + \beta_{\min} = q_2 + q_3 + \beta_{\min} \tag{A.18}$$

Remarking than $\beta_{\min} = \beta(q_2, l_2)$ we have:

$$q_{3,\max}(q_2) = \beta(q_2, l_2 + l_{pm\max}) - \beta(q_2, l_2) - q_2 \tag{A.19}$$

A.4 Static Forces Resolution

To compute the forces applied by the diagonal and parallel spring, l_d and l_p should be known. The first is already be given in function of q_2 or l_c by (A.16). The second requires to express $\beta_{\min}$ using (A.15):

$$\beta_{\min}(l_c) = \arccos\left(\frac{l_\Delta^2 + l_d^2(l_c) - l_2^2}{2 l_\Delta l_d(l_c)}\right) \tag{A.20}$$

Therefore l_p can be expressed from l_c (or q_2) and q_3:

$$l_p(l_c, q_3) = \sqrt{l_d^2(l_c) + l_\Delta^2 - 2 l_\Delta l_d(l_c)\cos\left(q_2(l_c) + q_3 + \beta_{\min}(l_c)\right)} \tag{A.21}$$

Finally to express the torque of the springs, the sinus law is needed to express α and γ:

$$\frac{l_\Delta}{\sin(\alpha)} = \frac{l_d(l_c)}{\sin(\pi - \bar{q}_2)} = \frac{l_d(l_c)}{\sin\left(q_2(l_c) + q_{2,ref}\right)} \tag{A.22}$$

$$\frac{l_d(l_c)}{\sin(\gamma)} = \frac{l_p(l_c, q_3)}{\sin\left(\beta(l_c, q_3)\right)} \tag{A.23}$$

It follows the springs torque:

$$\tau_d(l_c) = -k_d l_\Delta \sin\left(\bar{q}_2(l_c)\right)\left(\frac{l_{d,\max} + p_d}{l_d(l_c)} - 1\right) \tag{A.24}$$

$$\tau_p(l_c, q_3) = -k_p l_d(l_c) \sin\left(\beta(l_c, q_3)\right)\left(1 + \frac{p_p - l_2}{l_p(l_c, q_3)}\right) \tag{A.25}$$

Using the sine law, one can also express the torque exerced by the cable tension F_c on l_1 — l_2 junction:

$$\tau_c(F_c, l_c) = F_c \frac{l_1}{l_c} \sin\left(q_2(l_c)\right) \tag{A.26}$$

A.5 Forward kinematic expressed from (l_d, l_p, q_f)

We want to express L_L in function of (l_d, l_p, q_f). From figure A.1c, we have:

$$\theta = \alpha + q_2 + q_3 + q_f = \alpha + \beta(l_d, l_p) - \beta(l_d, l_2) + q_f \tag{A.27}$$

furthermore by inverting (A.16), we have:

$$q_2(l_d) = \arccos\left(\frac{l_d^2 - l_2^2 - l_\Delta^2}{2 l_2 l_\Delta}\right) \tag{A.28}$$

Now we can compute the coordinate of the foot position, (in the reference frame rotated by q_1):

$$x_f(l_d, l_p) = l_1 + l_2 \cos(q_2(l_d)) + (l_3 - l_\Delta)\cos(q_2 + q_3) \tag{A.29}$$

$$y_f(l_d, l_p) = -l_2 \sin(q_2(l_d)) + (l_3 - l_\Delta)\sin(\underbrace{q_2 + q_3}_{=\beta(l_d,l_p)-\beta(l_d,l_2)}) \tag{A.30}$$

$$\tag{A.31}$$

Finally we have:

$$\alpha(l_d, l_p) = \arctan\left(\left|\frac{y_f(l_d,l_p)}{x_f(l_d,l_p)}\right|\right) \tag{A.32}$$

$$L_{leg}(l_d, l_p) = \sqrt{x_f(l_d, l_p)^2 + y_f(l_d, l_p)^2} \tag{A.33}$$

B Characterisation of Piezoresistive Cells

This appendix regroup the characterization of all the cell presented in section 4.2.2.1 and whose parametrization is summarized in table 4.4.

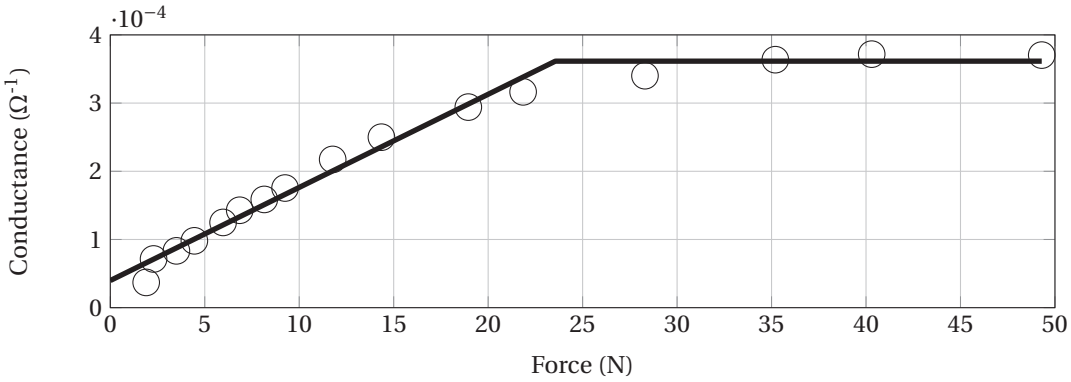


Figure B.1 – Characterization of cell type B, two Velostat layers, with glue, $l = 29\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.997088$

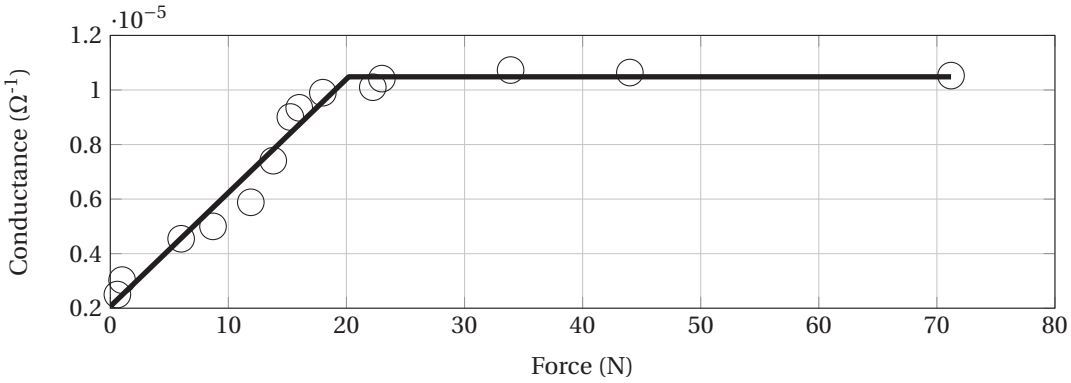


Figure B.2 – Characterization of cell type A, two Velostat layers, with glue, $l = 29\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.996530$

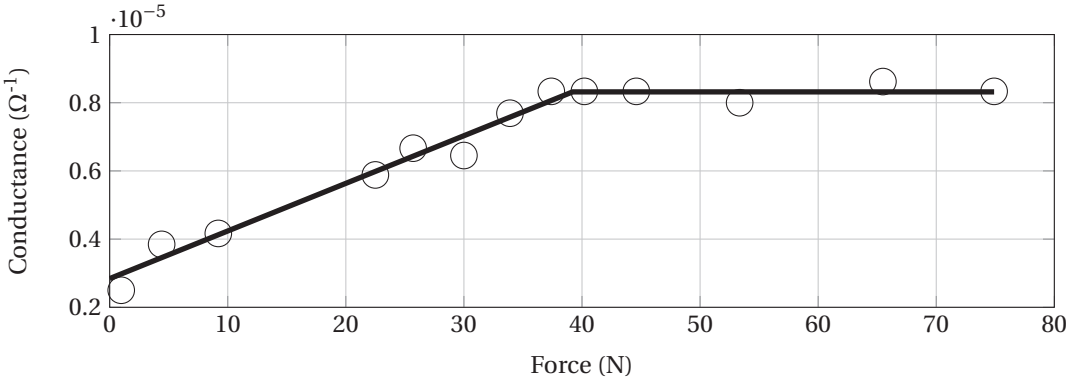


Figure B.3 – Characterization of cell type B, two Velostat layers, with glue, l = 8 mm and w = 5 mm. R^2 = 0.998323

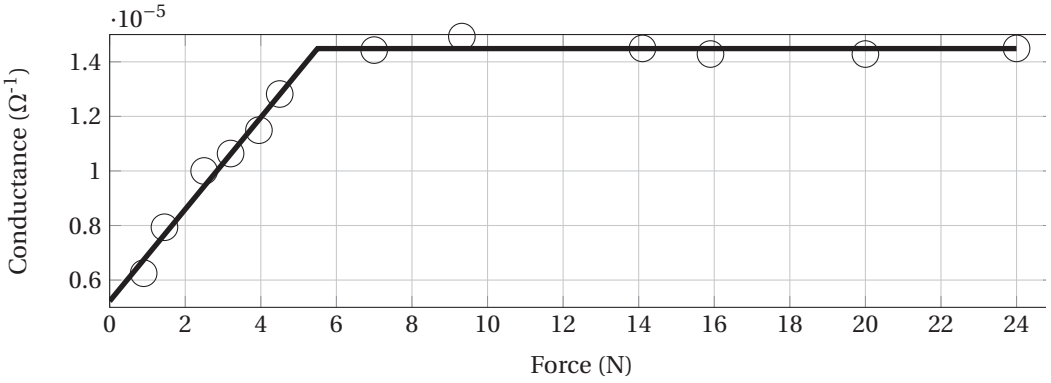


Figure B.4 – Characterization of cell type A, two Velostat layers, with glue, l = 8 mm and w = 29 mm. R^2 = 0.999435

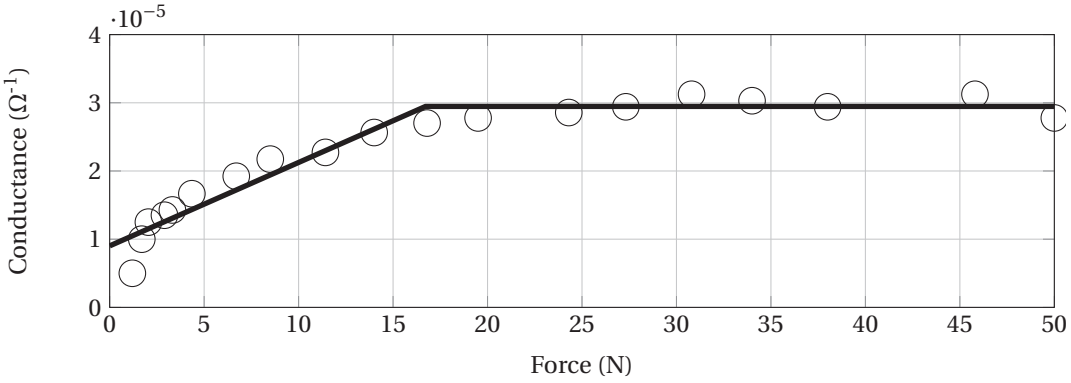


Figure B.5 – Characterization of cell type B, six Velostat layers, with glue, l = 8 mm and w = 5 mm. R^2 = 0.993491

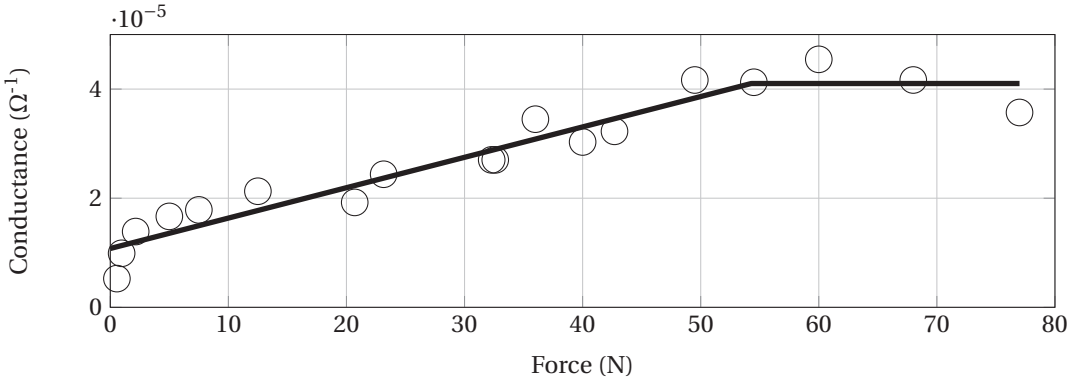


Figure B.6 – Characterization of cell type A, six Velostat layers, with glue, $l = 8\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.988849$

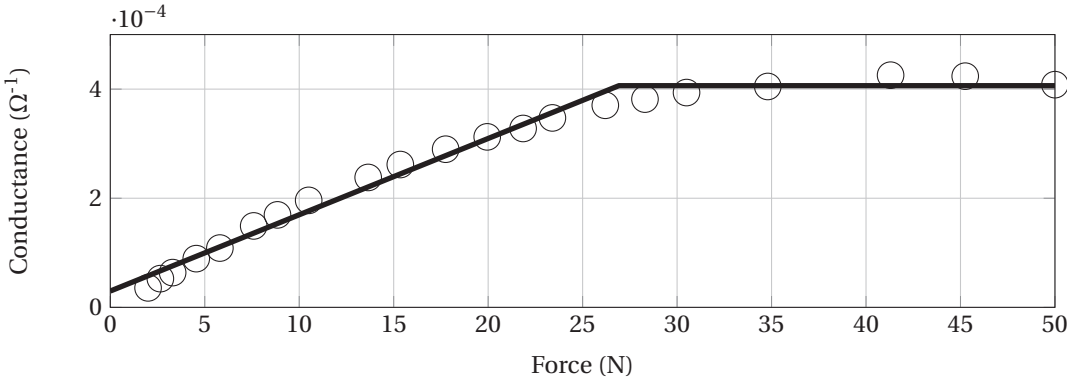


Figure B.7 – Characterization of cell type B, six Velostat layers, with glue, $l = 29\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.997367$

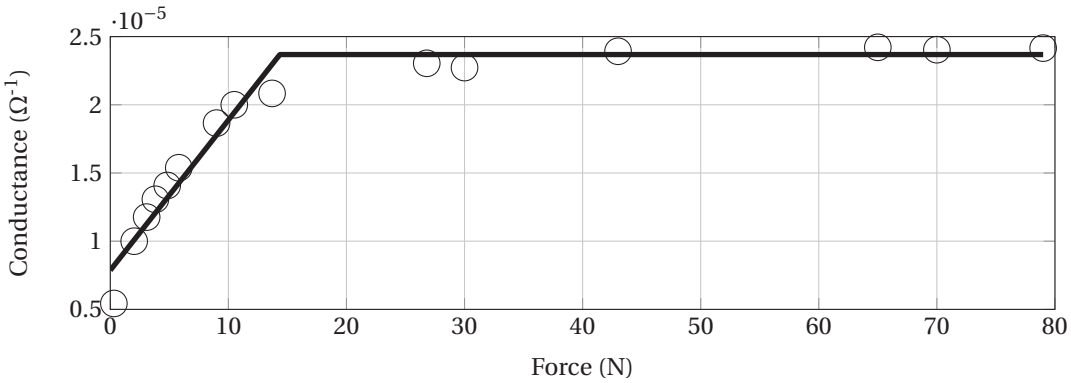


Figure B.8 – Characterization of cell type A, six Velostat layers, with glue, $l = 29\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.996566$

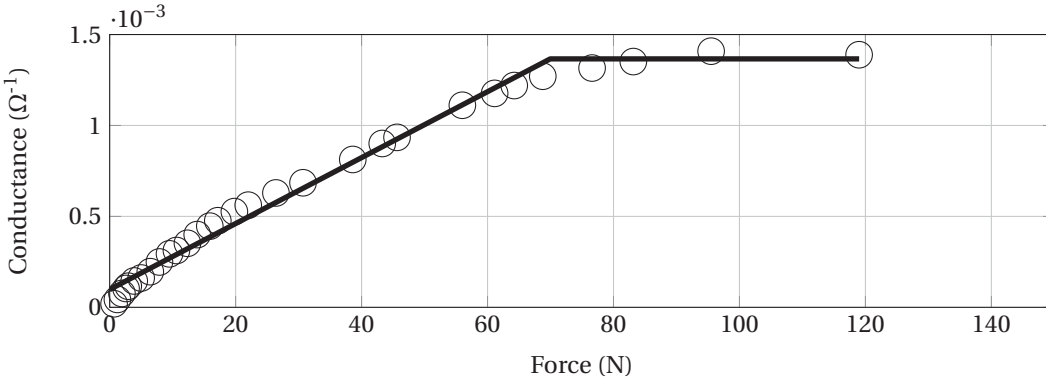


Figure B.9 – Characterization of cell type B, six Velostat layers, without glue, $l = 8\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.996604$

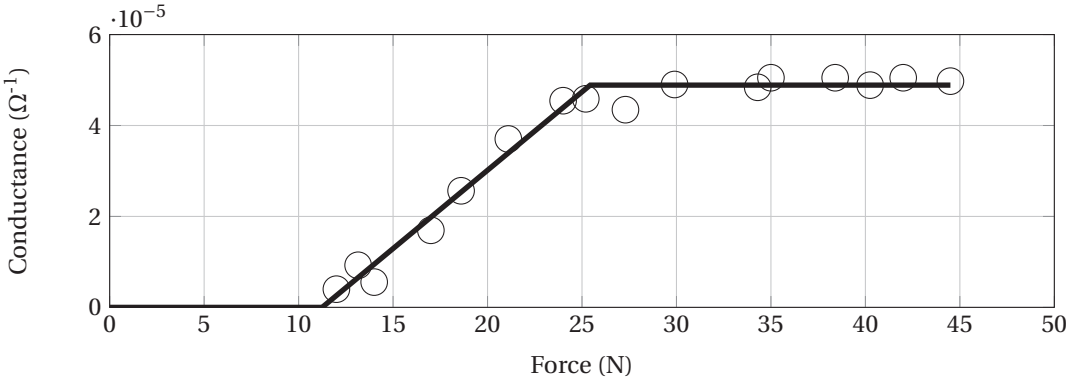


Figure B.10 – Characterization of cell type A, six Velostat layers, without glue, $l = 8\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.996591$

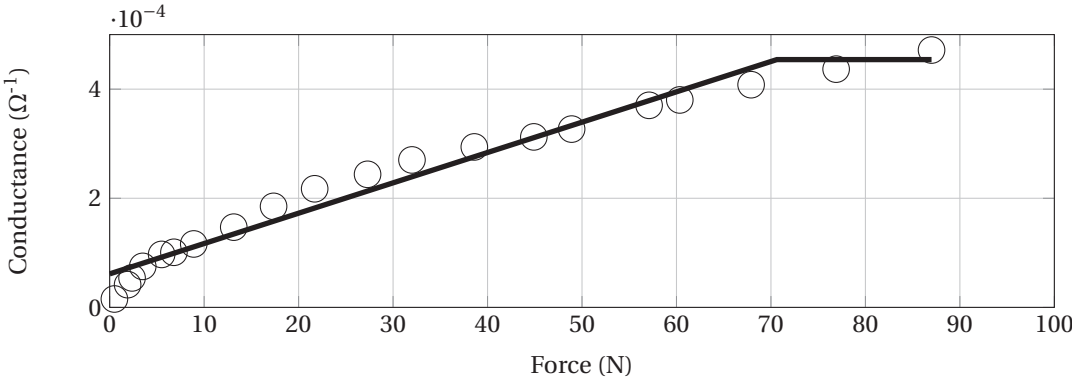


Figure B.11 – Characterization of cell type B, six Velostat layers, without glue, $l = 29\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.992912$

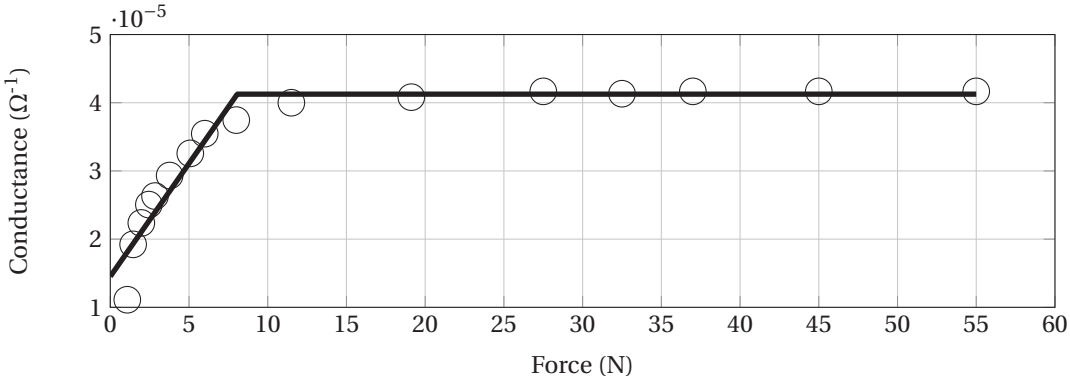


Figure B.12 – Characterization of cell type A, six Velostat layers, without glue, $l = 29\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.995463$

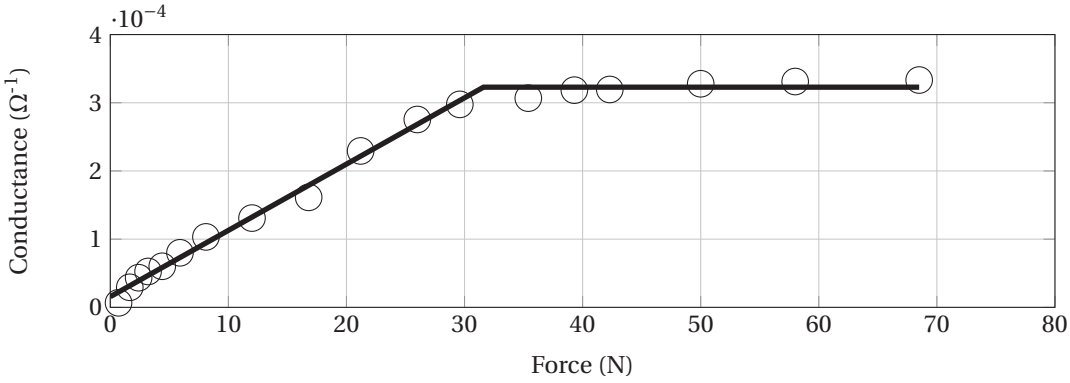


Figure B.13 – Characterization of cell type B, two Velostat layers, without glue, $l = 29\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.998476$

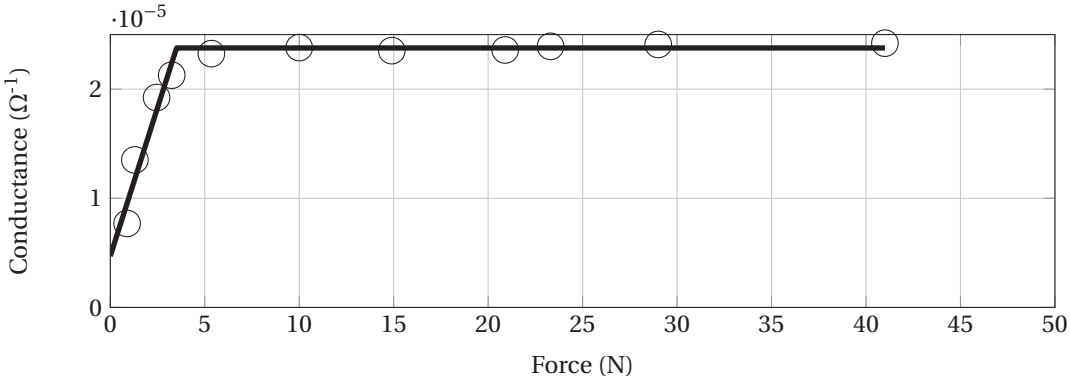


Figure B.14 – Characterization of cell type A, two Velostat layers, without glue, $l = 29\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.998049$

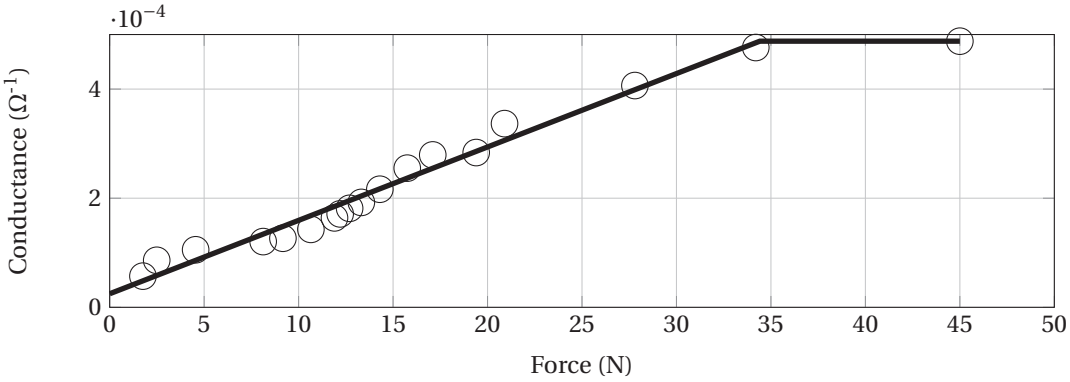


Figure B.15 – Characterization of cell type B, two Velostat layers, without glue, $l = 8\,\text{mm}$ and $w = 29\,\text{mm}$. $R^2 = 0.995355$

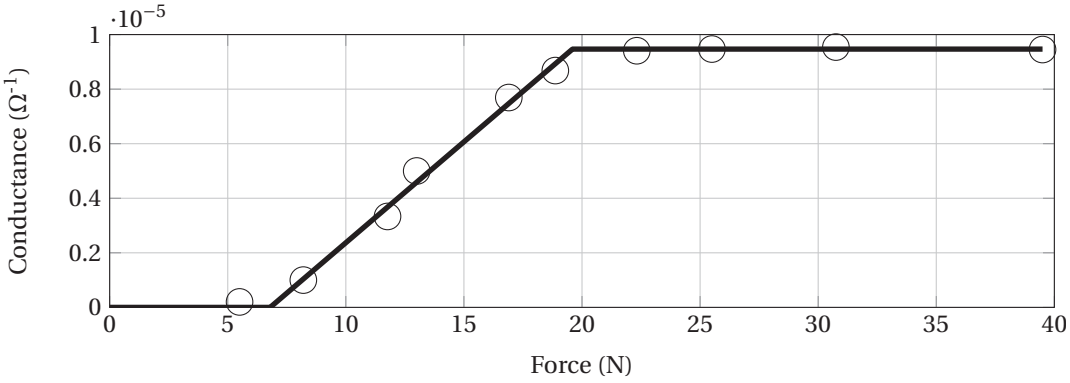


Figure B.16 – Characterization of cell type A, two Velostat layers, without glue, $l = 8\,\text{mm}$ and $w = 5\,\text{mm}$. $R^2 = 0.999156$

Bibliography

Acer, M., Salerno, M., Agbeviade, K. and Paik, J. [2015], 'Development and characterization of silicone embedded distributed piezoelectric sensors for contact detection', *Smart Materials and Structures* **24**(7), 075030. (pages 55, 66, 76, 78).

Ackerman, E. [2016], 'Boston Dynamics' SpotMini Is All Electric, Agile, and Has a Capable Face-Arm - IEEE Spectrum'.
URL: *http://spectrum.ieee.org/automaton/robotics/home-robots/boston-dynamics-spotmini* (page 3).

Ahmad, N., Ariffin, R., Ghazilla, R., Khairi, N. M. and Kasi, V. [2013], 'Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications', *International Journal of Signal Proccesing Systems* **1**(2), 256–262. (page 66).

Ajallooeian, M. [2015], Pattern Generation for Rough Terrain Locomotion with Quadrupedal Robots, PhD thesis, Ecole polytechnique fédérale de Lausanne. (pages 1, 9, 13, 19, 104, 109).

Ajallooeian, M., Gay, S., Ijspeert, A. J., Khansari-Zadeh, S. M., Kim, S., Billard, A., Rückert, E., Neumann, G., Waegeman, T., Wyffels, F., Schrauwen, B., Lemme, A., Reinhart, R. F., Rolf, M., Steil, J. J., Carbajal, J. P., Sumioka, H., Zhao, Q. and Suresh Kuppuswamy, N. [2010], Adaptive Modular Architectures for Rich Motor Skills Deliverable 4.1: Comparative evaluation of approaches in T.4.1-4.3 and working definition of adaptive module, Technical report. (page 97).

Ajallooeian, M., Tuleu, A., Sprowitz, A., Eckert, P., Vespignani, M. and Ijspeert, A. [2014], Rich Locomotion Skills with the Oncilla Robot, *in* 'Dynamic Walking 2014', number EPFL-POSTER-202161. (pages 30, 104).

Ajallooeian, M., Van Den Kieboom, J., Mukovskiy, A., Giese, M. A. and Ijspeert, A. J. [2013], 'A general family of morphed nonlinear phase oscillators with arbitrary limit cycle shape', *Physica D: Nonlinear Phenomena* **263**, 41–56. (page 9).

Alexander, R. M. [1984], 'The gaits of bipedal and quadrupedal animals', *The International Journal of Robotics Research* **3**(2), 49–59. (page 45).

Alexander, R. M. [1989], 'Optimization and gaits in the locomotion of vertebrates.', *Physiological reviews* **69**(4), 1199–1227. (pages 20, 41, 52, 108).

Ando, N., Kurihara, S., Biggs, G., Sakamoto, T., Nakamoto, H. and Kotoku, T. [2011], 'Software deployment infrastructure for component based RT-systems', *Journal of Robotics and Mechatronics* **23**(3), 350–359. (page 89).

Barasuol, V., Buchli, J., Semini, C., Frigerio, M., De Pieri, E. R. and Caldwell, D. G. [2013], A reactive controller framework for quadrupedal locomotion on challenging terrain, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', IEEE, pp. 2554–2561. (pages 3, 4, 7, 19, 38).

Bazeille, S., Barasuol, V., Focchi, M., Havoutis, I., Frigerio, M., Buchli, J., Semini, C. and Caldwell, D. G. [2013], Vision enhanced reactive locomotion control for trotting on rough terrain, *in* 'IEEE Conference on Technologies for Practical Robot Applications, TePRA', IEEE, pp. 1–6. (pages 3, 4).

Bigland-Ritchie, B. and Woods, J. J. [1984], 'Changes in muscle contractile properties and neural control during human muscular fatigue', *Muscle & Nerve* **7**(9), 691–699. (page 7).

Boaventura, T., Semini, C., Buchli, J., Frigerio, M., Focchi, M. and Caldwell, D. G. [2012], Dynamic torque control of a hydraulic quadruped robot, *in* '2012 IEEE International Conference on Robotics and Automation', pp. 1889–1894. (page 3).

Bosworth, W., Kim, S. and Hogan, N. [2016], The MIT super mini cheetah: A small, low-cost quadrupedal robot for dynamic locomotion, *in* 'SSRR 2015 - 2015 IEEE International Symposium on Safety, Security, and Rescue Robotics', IEEE, pp. 1–8. (pages 12, 14).

Box, G. E. P., Hunter, J. S. and Hunter, W. G. [2005], *Statistics for experimenters : design, innovation, and discovery*, Wiley-Interscience. (pages 61, 62).

Bruyninckx, H. [2001], Open robot control software: the OROCOS project, *in* '2001 IEEE International Conference on Robotics and Automation (ICRA)', Vol. 3, IEEE, pp. 2523–2528. (page 89).

Buchli, J., Pratt, J. and Roy, N. [2011], 'Editorial: Special Issue on Legged Locomotion', *The International Journal of Robotics Research* **30**(2), 139–140. (pages 2, 4, 37).

Crespi, A., Badertscher, A., Guignard, A. and Ijspeert, A. J. [2005], 'AmphiBot I: An amphibious snake-like robot', *Robotics and Autonomous Systems* **50**(4), 163–175. (page 9).

Cutkosky, M. R. and Ulmen, J. [2014], Dynamic Tactile Sensing, *in* 'Springer Tracts in Advanced Robotics', Vol. 95, Springer International Publishing, chapter 18, pp. 389–403. (pages 55, 78).

Dahiya, R. S. and Valle, M. [2008], Tactile sensing for robotic applications, *in* 'Sensors, Focus on Tactile, Force and Stress Sensors', number December, INTECH Open Access Publisher, chapter 15, pp. 289–304. (page 54).

Daley, M. A., Voloshina, A. and Biewener, A. A. [2009], 'The role of intrinsic muscle mechanics in the neuromuscular control of stable running in the guinea fowl.', *The Journal of physiology* **587**(Pt 11), 2693–707. (pages 8, 39).

Delattre, N. and Moretto, P. [2008], 'A new dimensionless number highlighted from mechanical energy exchange during running', *Journal of Biomechanics* **41**(13), 2895–2898. (page 52).

Dillet, R. [2016], 'Boston Dynamics CEO Marc Raibert demos the Spot at Disrupt | TechCrunch'. **URL:** *https://techcrunch.com/2016/09/14/boston-dynamics-ceo-marc-raibert-demos-the-spot-at-disrupt/* (page 3).

Eckert, P. and Ijspeert, A. [2016], Adaptive foot design for small quadruped robots, *in* 'Dynamic Walking 2016', pp. 4–5. (page 13).

Eckert, P., Spr, A., Witte, H. and Ijspeert, A. J. [2015], 'Comparing the effect of different spine and leg designs for a small bounding quadruped robot', *Icra* (D), 3128–3133. (pages 13, 81).

Eeonyx [2016], 'Eeonyx - Smart Conductive Fabrics'. **URL:** *http://eeonyx.com/about-our-products/* (page 54).

EtherCat Technology Group [2008], 'EtherCat Download Page'. **URL:** *https://www.ethercat.org/en/downloads.html* (page 100).

Farin, G. E. [1990], *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. (pages 39, 40).

Featherstone, R. [2008], *Rigid Body Dynamics Algorithms*, Springer. (pages 20, 21, 33).

Fraden, J. [2016], *Handbook of modern sensors: Physics, designs, and applications*. (page 54).

Gay, S. [2014], 'Visual Control of Legged Robots for Locomotion in Complex Environments using Central Pattern Generators', **6047**. (pages 13, 30, 104).

Gehring, C., Coros, S., Hutter, M., Bloesch, M., Hoepflinger, M. A. and Siegwart, R. [2013], Control of dynamic gaits for a quadrupedal robot, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', pp. 3287–3292. (pages 5, 7).

Geyer, H. and 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. (page 9).

Geyer, H., Seyfarth, A. and Blickhan, R. [2003], 'Positive force feedback in bouncing gaits?', *Proceedings. Biological sciences / The Royal Society* **270**(1529), 2173–83. (page 9).

Geyer, H., Seyfarth, A. and Blickhan, R. [2006], 'Compliant leg behaviour explains basic dynamics of walking and running.', *Proceedings. Biological sciences / The Royal Society* **273**(1603), 2861–2867. (pages 9, 39).

Gibson, J. J. [1958], 'Visually Controlled Locomotion and Visual Orientation in Animals', *British Journal of Psychology* **49**(3), 182–194. (page 10).

Gibson, J. J. [1979], *The ecological approach to visual perception*. (page 10).

Goebel, R., Sanfelice, R. G. and Teel, A. R. [2009], 'Hybrid dynamical systems', *IEEE Control Systems Magazine* **29**(2), 28–93. (page 2).

Göger, D., Gorges, N. and Worn, H. [2009], Tactile sensing for an anthropomorphic robotic hand: Hardware and signal processing, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', IEEE, pp. 895–901. (pages 56, 66).

Havoutis, I., Semini, C., Buchli, J. and Caldwell, D. G. [2013], Quadrupedal trotting with active compliance, *in* '2013 IEEE International Conference on Mechatronics, ICM 2013', pp. 610–616. (page 4).

Heim, S. W., Ajallooeian, M., Eckert, P., Vespignani, M. and Ijspeert, A. [2015], On designing an active tail for body-pitch control in legged robots via decoupling of control objectives, *in* 'Assistive Robotics: Proceedings of the 18th International Conference on Climbing and Walking Robots (CLAWAR)', Vol. 43, Emerald Group Publishing Limited, pp. 338–346. (page 13).

Herzog, A., Rotella, N., Mason, S., Grimminger, F., Schaal, S. and Righetti, L. [2016], 'Momentum Control with Hierarchical Inverse Dynamics on a Torque-Controlled Humanoid', *Autonomous Robots* **40**(3), 473–491. (page 7).

Hogy, S. M., Worley, D. R., Jarvis, S. L., Hill, A. E., Reiser, R. F. and Haussler, K. K. [2013], 'Kinematic and kinetic analysis of dogs during trotting after amputation of a pelvic limb', *American Journal of Veterinary Research* **74**(9), 1164–1171. (page 108).

Hutter, M., Gehring, C., Jud, D., Lauber, A. and Bellicoso, C. [2016], 'Anymal-a highly mobile and dynamic quadrupedal robot'. (page 5).

Ijspeert, A. J. [2008], 'Central pattern generators for locomotion control in animals and robots: A review', *Neural Networks* **21**(4), 642–653. (page 9).

Ijspeert, A. J., Crespi, A., Ryczko, D. and Cabelguen, J.-M. [2007], 'From swimming to walking with a salamander robot driven by a spinal cord model.', *Science (New York, N.Y.)* **315**(5817), 1416–1420. (page 9).

Interlink [2016], 'Interlink FSR'.
URL: *http://www.interlinkelectronics.com/datasheets/Datasheet_FSR.pdf* (page 54).

Jarvis, S. L., Worley, D. R., Hogy, S. M., Hill, A. E., Haussler, K. K. and Reiser, R. F. [2013], 'Kinematic and kinetic analysis of dogs during trotting after amputation of a thoracic limb', *American Journal of Veterinary Research* **74**(9), 1155–1163. (page 108).

Jatoth, R. and Kumar, T. [2009], 'Particle Swarm Optimization Based Tuning of Unscented Kalman Filter for Bearings Only Tracking', *2009 International Conference on Advances in Recent Technologies in Communication and Computing* **3**(3). (page 72).

Johnson, M., Shrewsbury, B., Bertrand, S., Wu, T., Duran, D., Floyd, M., Abeles, P., Stephen, D., Mertins, N., Lesman, A., Carff, J., Rifenburgh, W., Kaveti, P., Straatman, W., Smith, J., Griffioen, M., Layton, B., De Boer, T., Koolen, T., Neuhaus, P. and Pratt, J. [2015], 'Team IHMC's lessons learned from the DARPA robotics challenge trials', *Journal of Field Robotics* **32**(2), 192–208. (page 7).

Jones, S. J., Edgar, M. A. and Ransford, A. O. [1982], 'Sensory nerve conduction in the human spinal cord: epidural recordings made during scoliosis surgery.', *Journal of neurology, neurosurgery, and psychiatry* **45**(5), 446–51. (page 8).

Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M. and Schaal, S. [2011], 'Learning, planning, and control for quadruped locomotion over challenging terrain', *The International Journal of Robotics Research* **30**(2), 236–258. (pages 2, 37).

Kappassov, Z., Corrales, J. A. and Perdereau, V. [2015], 'Tactile sensing in dexterous robot hands - Review', *Robotics and Autonomous Systems* **74**, 195–220. (pages 54, 55, 107).

Kawamura, T., Inaguma, N., Nejigane, K., Tani, K. and Yamada, H. [2013], 'Measurement of slip, force and deformation using hybrid tactile sensor system for robot hand gripping an object', *International Journal of Advanced Robotic Systems* **10**. (page 56).

Kenndy, J., Eberhart, R. C., Kennedy, J. and Eberhart, R. C. [1995], Particle swarm optimization, *in* 'Proceedings of IEEE International Conference on Neural Networks', Vol. 4, pp. 1942–1948. (pages 43, 72).

Khoramshahi, M., Sprowitz, A., Tuleu, A., Ahmadabadi, M. N. and Ijspeert, A. [2013], Benefits of an active spine supported bounding locomotion with a small compliant quadruped robot, *in* 'Proceedings of 2013 IEEE International Conference on Robotics and Automation', number EPFL-CONF-186299, IEEE, pp. 3329–3334. (page 81).

Knüsel, J. [2013], 'Modeling a diversity of salamander motor behaviors with coupled abstract oscillators and a robot'. (page 10).

Korver, N. [2003], Adequacy of the Universal Serial Bus for real-time systems Individual Design Assignment ii Adequacy of the Universal Serial Bus for real-time systems, Technical report. (page 96).

Laamari, Y., Chafaa, K. and Athamena, B. [2015], 'Particle swarm optimization of an extended Kalman filter for speed and rotor flux estimation of an induction motor drive', *Electrical Engineering* **97**(2), 129–138. (page 72).

Liu, G. L., Habib, M. K., Watanabe, K. and Izumi, K. [2008], 'Central pattern generators based on Matsuoka oscillators for the locomotion of biped robots', *Artificial Life and Robotics* **12**(1-2), 264–269. (page 9).

Maiolino, P., Maggiali, M., Cannata, G., Metta, G. and Natale, L. [2013], 'A flexible and robust large scale capacitive tactile system for robots', *IEEE Sensors Journal* **13**(10), 3910–3917. (page 55).

Mangasarian, O. [1977], 'Solution of Symmetric Linear Complementarity Problems by lterative Methods', *Journal of Optimization Theory and Applications2* **22**(4). (page 28).

Metta, G., Fitzpatrick, P. and Natale, L. [2006], 'YARP: Yet another robot platform', *International Journal of Advanced Robotic Systems* **3**(1), 043–048. (page 89).

Moon, Y., Ko, N.-Y., Lee, K., Bae, Y. and Park, J.-K. [2009], 'Real-time EtherCAT Master Implementation on Xenomai for a Robot System', *International Journal of Fuzzy Logic and Intelligent Systems* **9**(3), 244–248. (page 101).

Moro, F. L., Spröwitz, A., Tuleu, A., Vespignani, M., Tsagarakis, N. G., Ijspeert, A. J. and Caldwell, D. G. [2013], 'Horse-like walking, trotting, and galloping derived from kinematic Motion Primitives (kMPs) and their application to walk/trot transitions in a compliant quadruped robot', *Biological Cybernetics* **107**(3), 309–320. (page 38).

Nordmann, A., Tuleu, A. and Wrede, S. [2013], A Domain-Specific Language and Simulation Architecture for the Oncilla Robot, *in* 'Workshop on Developments of Simulation Tools for Robotics & Biomechanics'. (page 91).

OptoForce [2016], 'OptoForce 3D Sensors'.
URL: *http://optoforce.com/3dsensor/* (page 56).

Or, Y. and Moravia, M. [2016], 'Analysis of Foot Slippage Effects on an Actuated Spring-mass Model of Dynamic Legged Locomotion', *International Journal of Advanced Robotic Systems* p. 1. (page 32).

Owaki, D., Kano, T., Nagasawa, K., Tero, A. and Ishiguro, A. [2012], 'Simple robot suggests physical interlimb communication is essential for quadruped walking.', *Journal of the Royal Society, Interface / the Royal Society* **10**(78), 20120669–. (pages 10, 38, 45, 46).

Owaki, D., Morikawa, L. and Ishiguro, A. [2013], 'Why do quadrupeds exhibit exclusively either trot or pace gaits ?', *Proc. of Dynamic Walking 2013* **11**(78), 20120669. (pages 10, 38, 46).

Playter, R. R., Buehler, M. and Raibert, M. H. [2006], 'BigDog', *Proceedings of SPIE* **6230**, 62302O–1–62302O–6. (page 3).

Pratt, G. A. and Williamson, M. M. [1995], 'Series elastic actuators', *IEEE/RSJ International Conference on Intelligent Robots and Systems. 'Human Robot Interaction and Cooperative Robots'* **1**(1524), 399–406. (page 5).

Pratt, J., Carff, J. and Drakunov, S. [2006], Capture Point : A Step toward Humanoid Push Recovery, pp. 200–207. (page 4).

Pratt, J., Chew, C.-M., Torres, A., Dilworth, P. and Pratt, G. [2001], 'Virtual Model Control: An Intuitive Approach for Bipedal Locomotion', *The International Journal of Robotics Research* **20**(2), 129–143. (page 5).

Prochazka, A., Gillard, D. and Bennett, D. J. [1997], 'Positive force feedback control of muscles.', *Journal of neurophysiology* **77**(6), 3226–3236. (page 9).

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R. and Mg, A. [2009], ROS: an open-source Robot Operating System, *in* '2009 IEEE International Conference on Robotics and Automation (ICRA)', Vol. 3, p. 5. (page 89).

Raibert, M. [2008], BigDog, the rough-terrain quadruped robot, *in* 'IFAC Proceedings Volumes (IFAC-PapersOnline)', Vol. 17. (page 3).

Raibert, M. H. [1990], 'Trotting , Pacing and Bounding By a Quadruped Robot', *Journal of biomechanics* **23**, 79–98. (page 4).

Raibert, M. H., Chepponis, M. and Brown, H. B. [1986], 'Running on Four Legs As Though They Were One', *IEEE Journal on Robotics and Automation* **2**(2), 70–82. (page 4).

Raibert, M. H. and Tello, E. R. [1986], 'Legged Robots That Balance', *IEEE Expert* **1**(4), 233. (page 4).

Remy, C. D., SIEGWART, R., GEHRING, C., BLOESCH, M., Hoepflinger, M. A., HUTTER, M., GEHRING, C., BLOESCH, M., Hoepflinger, M. A., Remy, C. D. and SIEGWART, R. [2012], StarlETH: A Compliant Quadrupedal Robot for Fast, Efficient, and Versatile Locomotion, *in* '15th International Conference on Climbing and Walking Robot - CLAWAR 2012', pp. 483–490. (page 5).

Righetti, L. and Ijspeert, A. J. [2008], Pattern generators with sensory feedback for the control of quadruped locomotion, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', pp. 819–824. (page 9).

Robotis [2016], 'ROBOTIS e-Manual v1.27.00 : Dynamixel Communication'.
URL: *http://support.robotis.com/en/product/dynamixel/dxl_communication.htm* (page 93).

Rossignol, S., Dubuc, R., Gossard, J.-P. and Dubuc, J. [2006], 'Dynamic Sensorimotor Interactions in Locomotion', *Physiological Reviews* **86**, 89–154. (pages 8, 9).

Rutishauser, S., Spröwitz, A., Righetti, L. and Ijspeert, A. [2008], Passive compliant quadruped robot using central pattern generators for locomotion control, *in* 'IEEE RAS & EMBS International Conference', pp. 710–715. (page 9).

Sadeghi, S. G., Chacron, M. J., Taylor, M. C. and Cullen, K. E. [2007], 'Neural variability, detection thresholds, and information transmission in the vestibular system.', *The Journal of neuroscience : the official journal of the Society for Neuroscience* **27**(4), 771–781. (page 7).

Seminara, L., Capurro, M., Cirillo, P., Cannata, G. and Valle, M. [2011], 'Electromechanical characterization of piezoelectric PVDF polymer films for tactile sensors in robotics applications', *Sensors and Actuators, A: Physical* **169**(1), 49–58. (page 55).

Semini, C., Buchli, J., Frigerio, M., Boaventura, T., Focchi, M., Guglielmino, E., Cannella, F., Tsagarakis, N. G. and Caldwell, D. G. [2011], HyQ–a dynamic locomotion research platform. (page 3).

Seok, S., Hyun, D. J., Park, S., Otten, D. and Kim, S. [2014], A highly parallelized control system platform architecture using multicore CPU and FPGA for multi-DoF robots, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', IEEE, pp. 5414–5419. (page 100).

Seok, S., Wang, A., Chuah, M. Y., Hyun, D. J., Lee, J., Otten, D. M., Lang, J. H. and Kim, S. [2014], 'Design Principles for Energy-Efficient Legged Locomotion and Implementation on the MIT Cheetah Robot', *IEEE/ASME Transactions on Mechatronics* **20**(3), 1117–1129. (pages 6, 12).

Seyfarth, A., Geyer, H. and Herr, H. [2003], 'Swing-leg retraction: a simple control model for stable running.', *The Journal of experimental biology* **206**(Pt 15), 2547–2555. (pages 9, 39).

Shahbazi, M. and Lopes, G. A. D. [2016], 'Coordination of Monopedal SLIP Models Towards Quadrupedal Running', *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* pp. 5440–5445. (page 9).

Siciliano and Khatib, O. [2008], *Springer Handbook of Robotics*. (page 20).

Soltoggio, A. and Steil, J. J. [2012], 'How Rich Motor Skills Empower Robots at Last: Insights and Progress of the AMARSi Project', *KI - Künstliche Intelligenz* **26**(4), 407–410. (page 11).

Sonar, H. A. and Paik, J. [2016], 'Soft Pneumatic Actuator Skin with Piezoelectric Sensors for Vibrotactile Feedback', *Frontiers in Robotics and AI* **2**, 38. (pages 55, 56, 57, 66, 67).

Spröwitz, A., Kuechler, L., Tuleu, A., Ajallooeian, M., D'Haene, M., Möckel, R. and Ijspeert, A. [2011], Oncilla robot: a light-weight bio-inspired quadruped robot for fast locomotion in rough terrain, *in* '5th International Symposium on Adaptive Motion of Animals and Machines', number EPFL-CONF-182313. (pages 11, 81).

Spröwitz, A., Moeckel, R., Maye, J. and Ijspeert, a. J. [2008], 'Learning to Move in Modular Robots using Central Pattern Generators and Online Optimization', *The International Journal of Robotics Research* **27**(3-4), 423–443. (page 9).

Spröwitz, A., Moeckel, R., Vespignani, M., Bonardi, S. and Ijspeert, A. J. [2014], 'Roombots: A hardware perspective on 3D self-reconfiguration and locomotion with a homogeneous modular robot', *Robotics and Autonomous Systems* **62**(7), 1016–1033. (page 91).

Spröwitz, A. T., Ajallooeian, M., Tuleu, A. and Ijspeert, A. J. [2014], 'Kinematic primitives for walking and trotting gaits of a quadruped robot with compliant legs.', *Frontiers in computational neuroscience* **8**(March), 27. (page 13).

Spröwitz, A., Tuleu, A., Vespignani, M., Ajallooeian, M., Badri, E. and Ijspeert, A. J. [2013], 'Towards dynamic trot gait locomotion: Design, control, and experiments with Cheetah-cub,

a compliant quadruped robot', *The International Journal of Robotics Research* **32**(8), 932–950. (pages 1, 9, 11, 12, 13, 14, 17, 18, 19, 20, 30, 31, 38, 42, 43, 46, 53, 66, 69, 73, 81, 103, 104, 107, 109).

Tekscan [2016], 'Flexiforce'.
URL: *http://www.tekscan.com/flexiforce.html* (pages 54, 58).

The Go Team [2015], 'build - The Go Programming Language'.
URL: *https://golang.org/pkg/go/build/* (page 84).

Thrun, S., Burgard, W. and Fox, D. [2005], 'Gaussian Filters', *Probalistic Robotics* pp. 39–84. (pages 67, 68).

Tuleu, A., Degrave, J., Wyffels, F., Gay, S. and Waegeman, T. [2013], 'sbcp-uc'.
URL: *https://redmine.amarsi-project.eu/projects/sbcp-uc* (page 95).

Tuleu, A., Nordmann, A., Degrave, J. and Ajallooeian, M. [2012], 'libsbcp'.
URL: *https://redmine.amarsi-project.eu/projects/libsbcp* (page 95).

Tuleu, A., Nordmann, A., Rückert, E., Ajallooeian, M. and Wrede, S. [2012], 'sbcpd'.
URL: *https://redmine.amarsi-project.eu/projects/sbcpd* (page 95).

USB 2.0 Promoters [2000], 'USB.org - USB 2.0 Documents'.
URL: *http://www.usb.org/developers/docs/usb20_docs/#usb20adopters* (page 96).

Van den Kieboom, J. [2014], On the dynamics of human locomotion and co-design of lower limb assistive devices, PhD thesis. (pages 43, 88).

Villeger, D., Costes, A., Watier, B. and Moretto, P. [2014], 'Modela-r as a Froude and Strouhal dimensionless numbers combination for dynamic similarity in running', *Journal of Biomechanics* **47**(16), 3862–3867. (page 52).

Warren Jr., W. H. [1998], 'Visually Controlled Locomotion: 40 years Later', *Ecological Psychology* **10**(3), 177–219. (page 10).

Weinmeister, K., Eckert, P., Witte, H. and Ijspeert, A. [2015], Cheetah-cub-S : Steering of a Quadruped Robot using Trunk Motion, *in* '7th Symposium on Adaptive Motion of Animals and Machine'. (pages 13, 81).

Weiss Robotics Gmbh [2016], 'Tactile Sensing – Weiss Robotics GmbH & Co. KG'.
URL: *https://www.weiss-robotics.com/en/product-category/tactile-sensing/* (page 54).

Wickler, S. J., Hoyt, D. F., Cogger, E. a. and Myers, G. [2003], 'The energetics of the trot-gallop transition.', *The Journal of experimental biology* **206**(Pt 9), 1557–1564. (pages 45, 108).

Witte, H., Hackert, R., Ilg, W., Biltzinger, J., Schillinger, N., Biedermann, F., Jergas, M., Preuschoft, H., Fischer, M. S., Schilling, N., Biedermann, F., Jergas, M., Preuschoft, H. and Fischer, M. S. [2000], Quadrupedal Mammals as Paragons for Walking Machines, *in* 'Proceedings of Adaptive Motion in Animals and Machines', p. 6 pp. (pages 11, 19).

Wooden, D., Malchano, M., Blankespoor, K., Howardy, A., Rizzi, A. A. and Raibert, M. [2010], Autonomous navigation for BigDog, *in* 'Proceedings - IEEE International Conference on Robotics and Automation', IEEE, pp. 4736–4741. (page 3).

Zarucco, L., Driessen, B., Scandella, M., Cozzi, F. and Cantile, C. [2010], 'Sensory nerve conduction and nociception in the equine lower forelimb during perineural bupivacaine infusion along the palmar nerves', *Canadian Journal of Veterinary Research* **74**(4), 305–313. (page 8).

Zhao, W., Yu, J., Fang, Y. and Wang, L. [2006], Development of multi-mode biomimetic robotic fish based on central pattern generator, *in* 'IEEE International Conference on Intelligent Robots and Systems', IEEE, pp. 3891–3896. (page 9).

Index

Alexandre Tuleu

alexandre@tuleu.science
EPFL - STI - IBI - BIOROB, Station 9, 1015 Lausanne SWITZERLAND
+41 21 693 26 02
Nationalité: Française

http://biorob.epfl.ch/people/tuleu
http://linkedin.com/in/alexandre-tuleu-5439a723

EDUCATION

PHD IN ROBOTICS CONTROL AND INTELLIGENT SYSTEMS

École Polytechnique Fédérale de Lausanne
Instituto Superior Técnico
September 2011 - June 2017 | Lausanne, CH & Lisbon, PT

MSC IN MICROENGINEERING

École Polytechnique Fédérale de Lausanne
September 2008 - March 2010 | Lausanne, CH

DIPLÔME D'INGÉNIEUR DE L'ÉCOLE POLYTECHNIQUE

École Polytechnique - Paristech
September 2005 - July 2008 | Palaiseau, FR
Specialization in Optimization & Signal Processing

EXPERIENCE

BIOROB | Research Assistant
May 2010 – September 2011 | Lausanne, CH

- Development of the simulator for the European FP7 AMARSi project quadruped robot and a suite of benchmark tools to compare different low-level control architecture for quadruped controller within the consortium.
- Development of Debian packages to help sharing the software toolsuite amongs partners.
- Development of the embedded software of the Cheetah-Cub robot.

PUBLICATIONS

JOURNALS

- 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. The International Journal of Robotics Research, 32(8), 932–950. article.
- Spröwitz, A. T., Ajallooeian, M., Tuleu, A., & Ijspeert, A. J. (2014). Kinematic primitives for walking and trotting gaits of a quadruped robot with compliant legs. Frontiers in Computational Neuroscience, 8(March), 27.
- Moro, F. L., Spröwitz, A., Tuleu, A., Vespignani, M., Tsagarakis, N. G., Ijspeert, A. J., & Caldwell, D. G. (2013). Horse-like walking, trotting, and galloping derived from kinematic Motion Primitives (kMPs) and their application to walk/trot transitions in a compliant quadruped robot. Biological Cybernetics, 107(3), 309–320.

CONFERENCE PAPERS

- 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).
- Nordmann, A., Tuleu, A., & Wrede, S. (2013). A Domain-Specific Language and Simulation Architecture for Motor Skill Exploration. 8th Workshop on Software Development and Integration in Robotics (SDIR) at 2013 IEEE International Conference on Robotics and Automation,
- Ajallooeian, M., Pouya, S., Tuleu, A., Sproewitz, A., & Ijspeert, A. J. (2013). Towards Modular Control for Moderately Fast Locomotion over Unperceived Rough Terrain. In Dynamic Walking (p. 2013).
- Ajallooeian, M., Gay, S., Tuleu, A., Sprowitz, A., & Ijspeert, A. J. (2013). Modular control of limit cycle locomotion over unperceived rough terrain. In IEEE International Conference on Intelligent Robots and Systems (pp. 3390–3397).

- Khoramshahi, M., Sprowitz, A., Tuleu, A., Ahmadabadi, M. N., & Ijspeert, A. (2013). Benefits of an active spine supported bounding locomotion with a small compliant quadruped robot. In Proceedings of 2013 IEEE International Conference on Robotics and Automation (pp. 3329–3334). inproceedings, IEEE.
- Hauser, S., Eckert, P., Tuleu, A., & Ijspeert, A. (2016). Friction and damping of a compliant foot based on granular jamming for legged robots. In Biomedical Robotics and Biomechatronics (BioRob), 2016 6th IEEE International Conference on (pp. 1160–1165). inproceedings.
- Spröwitz, A., Kuechler, L., Tuleu, A., Ajallooeian, M., D'Haene, M., Möckel, R., & Ijspeert, A. (2011). Oncilla robot: a light-weight bio-inspired quadruped robot for fast locomotion in rough terrain. In 5th International Symposium on Adaptive Motion of Animals and Machines. inproceedings.
- Spröwitz, A., Tuleu, A., Vespignani, M., Ajallooeian, M., & Badri, E. (2012). Robot Trotting with Segmented Legs in Simulation and Hardware. In Proceedings of the 7th Annual Dynamic Walking Conference (pp. 46–47). inproceedings.

LANGUAGES

PROGRAMMING

Over 5000 lines:
C++ • C • MatLab • Golang • LaTeX
Over 1000 lines:
Ruby • Node • AngularJS • Rails
Familiar:
Linux • Yocto • Debian • macOS

SPOKEN & WRITTEN

Native:
French **Fluent**:
English
Working Knowledge:
German